Electronic Transactions on Numerical Analysis.
Volume 46, pp. 1–xx, 2017.

ISSN 1068–9613.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

# BINET-TYPE POLYNOMIALS AND THEIR ZEROS*

WALTER GAUTSCHI† AND GRADIMIR V. MILOVANOVIĆ‡

**Abstract.** Procedures based on moments are developed for computing the three-term recurrence relation for orthogonal polynomials relative to the Binet, generalized Binet, squared Binet, and related subrange weight functions. Monotonicity properties for the zeros of the respective orthogonal polynomials are also established.

**Key words.** Binet weight function, orthogonal polynomials, zeros, monotonicity

**AMS subject classifications.** 33C47, 65D20

**1. Introduction.** The *Binet weight function* may be defined by

$$w_1(x) = -\log(1 - \mathrm{e}^{-|x|}) \quad \text{on } [-\infty, \infty]. \tag{1.1}$$

It has been introduced, in connection with a number of summation formulas [2]–[4], in [4, Eq. (5.4)], where the Binet *distribution* is defined by $w^B(x) = w_1(2\pi x)/(2\pi)$ and used in Binet's summation formula, *ibid.*, Eq. (5.15). More generally,

$$w_1(x; \alpha) = -\log(1 - \alpha\mathrm{e}^{-|x|}) \quad \text{on } [-\infty, \infty], \;\; 0 < \alpha < 1, \tag{1.2}$$

is, what may be called, the *generalized Binet weight function*. We are interested in the polynomials orthogonal with respect to the weight functions (1.1) and (1.2), in particular in the recurrence formula

$$\pi_{k+1}(x) = (x - \alpha_k)\pi_k(x) - \beta_k\pi_{k-1}(x), \quad k = 0, 1, 2, \dots\,, \;\; \pi_{-1}(x) = 0, \tag{1.3}$$

satisfied by the respective monic polynomials. The coefficients $\alpha_k$, $\beta_k$ can be obtained by the classical Chebyshev algorithm, since the moments of both weight functions are known in terms of factorials and generalized polylogarithm functions. It is true that the classical Chebyshev algorithm is notoriously unstable, but we get around this problem by using sufficiently high precision. This is discussed for the Binet and generalized Binet weight functions in Section 2. The same can be done with the squares of the Binet and generalized Binet weight functions (Section 3), with the halfrange Binet and generalized Binet weight functions (Section 4), as well as with the squares of the halfrange weight functions (Section 5). Upper and lower subrange Binet weight functions are also considered in Section 6.

In the case of the generalized weight functions with parameter $\alpha$, we prove that all zeros, resp. positive zeros when the weight function is symmetric, are monotonically decreasing as functions of $\alpha$. They are shown to be monotonically increasing as functions of the upper or lower limit of the orthogonality interval. We do this by applying Markov's theorem and two variants thereof, and by a new related theorem of our own.

**2. Binet and generalized Binet weight functions.** Moment-related methods and their implementation, both in Matlab and Mathematica, are considered in Section 2.1 for the Binet weight function, and in Section 2.2.1 for generalized Binet weight functions. Section 2.2.2 is

*Received... Accepted... Published online on... Recommended by.... The research of G. V. Milovanović is supported in part by the Serbian Academy of Sciences and Arts (No. Φ-96).

†Department of Computer Science, Purdue University, West Lafayette, IN 47907-2066, USA (wgautschi@purdue.edu).

‡Serbian Academy of Sciences and Arts, Belgrade, Serbia & University of Niš, Faculty of Sciences and Mathematics, 18000 Niš, Serbia (gvm@mi.sanu.ac.rs).

**ETNA**
**Kent State University and**
**Johann Radon Institute (RICAM)**



devoted to a study of the zeros of orthogonal polynomials depending on a parameter and, in particular, to the monotone behavior of the zeros of generalized Binet polynomials $\pi_n^\alpha$ when considered functions of the parameter $\alpha$.

**2.1. Binet weight function.** Since the weight function in (1.1) is symmetric with respect to the origin, its moments are

$$\mu_k = \begin{cases} 0 & \text{if } k \text{ is odd,} \\ -2\displaystyle\int_0^\infty x^k \log(1-\mathrm{e}^{-x})\,\mathrm{d}x & \text{if } k \text{ is even.} \end{cases} \tag{2.1}$$

Substituting $\mathrm{e}^{-x} = t$ in the integral of (2.1), one gets

$$\mu_k = 2(-1)^{k+1}\int_0^1 \log^k t\, \log(1-t)\,\frac{\mathrm{d}t}{t}\,,$$

and thus

$$\mu_k = 2\,k!\,S_{k+1,1}(1) = 2\,k!\,\mathrm{Li}_{k+2}(1) = 2\,k!\,\zeta(k+2), \quad k \ \text{ even,} \tag{2.2}$$

where

$$S_{n,p}(x) = \frac{(-1)^{n+p-1}}{(n-1)!p!}\int_0^1 \log^{n-1}(t)\log^p(1-xt)\,\frac{\mathrm{d}t}{t} \tag{2.3}$$

is the Nielsen generalized polylogarithm [11, Eq. (1.1)] and $\mathrm{Li}_n(x)$ the ordinary polylogarithm ([10, Eq. (1.1)]). We can thus apply the classical Chebyshev algorithm (cf., e.g., [5, §2.1.7]) in sufficiently high precision to generate any number $N$ of recurrence coefficients $\alpha_k$, $\beta_k$, $k = 0, 1, \ldots, N-1$, to any desired accuracy.

To implement this in Matlab, one needs, foremost, the routine `smom_binet.m`* that generates in `dig`-digit arithmetic the $2\,\mathtt{N}\times 1$ array `mom` of the first $2\,\mathtt{N}$ moments (2.1),

$$\mathtt{mom} = \mathtt{smom\_binet}(\mathtt{dig}, \mathtt{N}).$$

In addition, the routine `dig_binet.m` is provided that, with the command

$$[\mathtt{ab}, \mathtt{dig}] = \mathtt{dig\_binet}(\mathtt{N}, \mathtt{dig0}, \mathtt{dd}, \mathtt{nofdig}), \tag{2.4}$$

helps to determine the number `dig` of digits needed to obtain the $\mathtt{N}\times 2$ array `ab` of the first `N` recurrence coefficients $\alpha_k$, $\beta_k$, $k = 0, 1, \ldots, \mathtt{N}-1$, to an accuracy of `nofdig` digits. The way this routine works is as follows: It first calculates the array `ab` with an estimated number `dig0` of digits (which is printed) and then successively increases (and prints) the number of digits in units of `dd` digits until the desired accuracy is achieved. If this happens after just one increment, the value of `dig0` must be lowered until at least two increments have occurred. The last value of `dig` printed can then be taken as the number of digits needed. A typical value of `dd` is 4. The command

$$\mathtt{ab} = \mathtt{sr\_binet}(\mathtt{dig}, \mathtt{nofdig}, \mathtt{N}), \tag{2.5}$$

finally, computes directly, in `dig`-digit arithmetic, the first `N` recurrence coefficients and places them to `nofdig` digits into the $\mathtt{N}\times 2$ array `ab`.

EXAMPLE 2.1. The first 100 recurrence coefficients to 32 digits of the Binet weight function.

---

*All Matlab routines and textfiles referenced in this paper can be accessed at `https://www.cs.purdue.edu/homes/wxg/archives/2002/codes/BINET.html`.

With $\mathtt{N} = 100$, $\mathtt{dig0} = 56$, $\mathtt{dd} = 4$, $\mathtt{nofdig} = 32$, the routine (2.4) yields $\mathtt{dig} = 64$ after two increments and also produces the $100{\times}2$ array `ab` of the first 100 recurrence coefficients to an accuracy of 32 digits. The $\alpha$- and $\beta$-coefficients are depicted in the second and third plot of Fig. 2.1, the first showing the Binet weight function. The recurrence coefficients are also made available in the textfile `coeff_binet.txt`, which can be loaded into the Matlab working window by the routine `loadvpa.m`. For the latter, see [6, p. ix]; see also [8, 2.3.8]. The array `ab` can also be obtained directly with the routine (2.5), using $\mathtt{dig} = 64$, $\mathtt{nofdig} = 32$, and $\mathtt{N} = 100$.

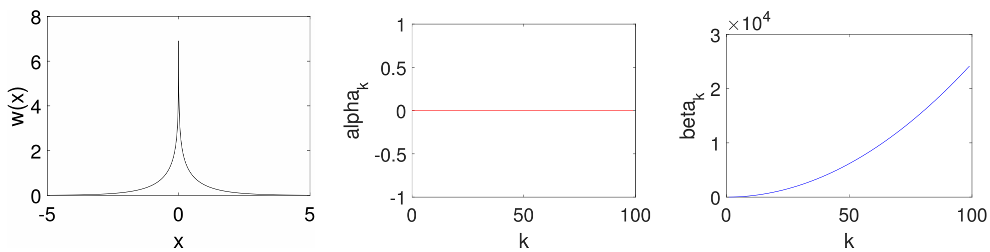


FIG. 2.1. *Binet weight function and its recurrence coefficients.*

The same 100 recurrence coefficients have been obtained in symbolic form by the Mathematica package `OrthogonalPolynomials` ([1], [13]), using the commands

```
<<orthogonalPolynomials`
momGB[k_,alpha_]:=If[OddQ[k],0,2 k!PolyLog[k+1,1,alpha]];
mom = Table[momGB[k,1], {k,0,199}];
{alBSym,beBSym}=aChebyshevAlgorithm[mom,Algorithm->Symbolic]
```

The $\beta_k$ for $0 \leq k \leq 11$, so obtained, are given on p. 457 of [12] in rational form, and those for $12 \leq k \leq 39$ to 60 decimals on p. 458 of the same reference. There is complete agreement to all 32 digits between the coefficients obtained in Matlab and those obtained in Mathematica rounded to 32 digits. The fact that the recurrence coefficients are rational numbers multiplied by $\pi^2$ makes the computation in symbolic Mathematica extremely fast. For example, on the MacBook Pro Retina OSX 10.12.6 laptop, the first 100 recurrence coefficients in symbolic form are obtained in 1.04 s. These symbolic formulae can then be used to compute the coefficients to an arbitrary precision. For example, `N[beBSym,32]` produces the $\beta$-coefficients to 32 digits in 1.1 ms, and `N[beBSym,500]` the same coefficients to 500 digits in 1.3 ms. If one uses the numerical calculation option in the Chebyshev algorithm,

```
{alB,beB}=aChebyshevAlgorithm[mom,WorkingPrecision->58]
```

with 58-digit working precision (`WP`), the first 100 recurrence coefficients are obtained to 32 digits in 77.3 ms. With 86-digit `WP`, they are obtained to 60 digits in 81.8 ms, and with 160-digit `WP` to 135 digits in 84.8 ms. In contrast, Matlab, on the Dell Optiplex 790 computer, takes 186 s to compute the same 100 coefficients to 32 digits.

**2.2. Generalized Binet weight function.**

**2.2.1. Recurrence coefficients.** The weight function (1.2), again being symmetric, has moments

$$\mu_k = \begin{cases} 0 & \text{if } k \text{ is odd,} \\ -2\displaystyle\int_0^\infty x^k \log(1 - \alpha \mathrm{e}^{-x})\mathrm{d}x & \text{if } k \text{ is even.} \end{cases} \tag{2.6}$$

**ETNA**
**Kent State University and**
**Johann Radon Institute (RICAM)**



Similarly as in Subsection 2.1, one finds

$$\mu_k = 2\,k!\,S_{k+1,1}(\alpha) = 2\,k!\,\mathrm{Li}_{k+2}(\alpha), \quad k \text{ even.} \tag{2.7}$$

The moments (2.6) are generated by the Matlab command `mom=smom_gbinet(dig, N,a)`, where `a` is the value of $\alpha$ and $0 < \alpha < 1$.

EXAMPLE 2.2. The first 100 recurrence coefficients to 32 digits of the generalized Binet weight function for $\alpha = 1/2$.

The Matlab command `[ab,dig]=dig_gbinet(N,a,dig0,dd,nofdig)`, when run with `N` $= 100$, `a` $= 1/2$, `dig0` $= 56$, `dd` $= 4$, `nofdig` $= 32$, yields `dig` $= 64$. The same command, or more directly, the command `ab=sr_gbinet(dig,nofdig,N,a)` with `dig` $= 64$, produces the $100\times 2$ array `ab` of the first 100 recurrence coefficients to an accuracy of 32 digits. They are depicted in the second and third plot of Fig. 2.2, the first showing the generalized Binet weight function for $\alpha = 1/2$. They are also made available in the textfile `coeff_gbinet.txt`.

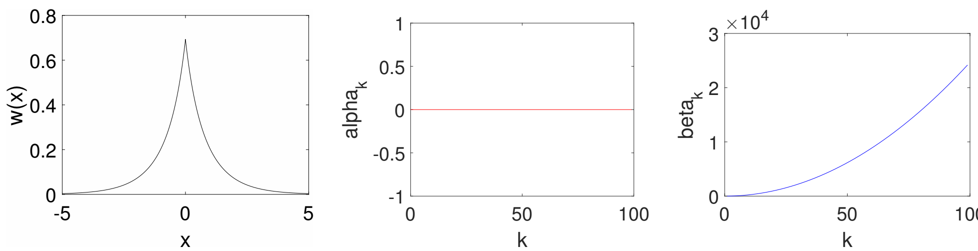


FIG. 2.2. *Generalized Binet weight function for $\alpha = 1/2$ and its recurrence coefficients.*

Unlike the case $\alpha = 1$, when the recurrence coefficients are simply rational numbers multiplied by $\pi^2$, in the general case $0 < \alpha < 1$, the symbolic expressions of the recurrence coefficients $\beta_k$ become rapidly more complicated with increasing $k$, and hence the runtime correspondingly larger. Yet, using the numerical calculation option in both, the computation of the moments and Chebyshev's algorithm, yields fast algorithms, similarly to those described in Example 2.1.

**2.2.2. Zeros of orthogonal polynomials.** The first objective of this subsection is to investigate the zeros of orthogonal polynomials depending on a parameter and to prove some monotonicity results. For this we use Markov's theorem and two simple corollaries thereof, as well as a new, but related, theorem. The second objective is to show appropriate plots.

We first recall Markov's theorem ([14, Theorem 6.12.1]).

THEOREM 2.3 (A. Markov). *Let $w(x;\alpha)$ be a positive weight function on $[a,b]$, $-\infty \le a < b \le \infty$, depending on a parameter $\alpha$, $\alpha_1 < \alpha < \alpha_2$. Assume that the first $2n$ moments of $w$ and of $\frac{\partial w}{\partial \alpha}$ exist, and let $\pi_n^\alpha$ denote the monic polynomial of degree $n$ orthogonal with respect to the weight function $w(\,\cdot\,;\alpha)$. Then each zero of $\pi_n^\alpha$ is an increasing (decreasing) function of $\alpha$ on $(\alpha_1,\alpha_2)$ provided that*

$$M(x;\alpha) := \frac{1}{w(x;\alpha)}\,\frac{\partial w(x;\alpha)}{\partial \alpha} \tag{2.8}$$

*is an increasing (decreasing) function of $x$ on $[a,b]$.*

Here are two simple corollaries to Markov's theorem.

COROLLARY 2.4. *Let $w(x;\alpha)$ be as in the theorem, and $w_r(x;\alpha) = [w(x;\alpha)]^r$, $r > 0$, have finite moments of order $\le 2n-1$. Then each zero of the $n$th-degree polynomial orthogonal with respect to the weight function $w_r$ is increasing* (*decreasing*) *on* $(\alpha_1, \alpha_2)$ *depending on whether* (2.8) *is increasing* (*decreasing*) *on* $[a, b]$.

*Proof.* We have

$$\frac{1}{w_r(x;\alpha)}\,\frac{\partial w_r(x;\alpha)}{\partial\alpha} = \frac{r\,[w(x;\alpha)]^{r-1}}{[w(x;\alpha)]^r}\,\frac{\partial w(x;\alpha)}{\partial\alpha} = \frac{r}{w(x;\alpha)}\,\frac{\partial w(x;\alpha)}{\partial\alpha}. \qquad \square$$

If $r < 0$, the type of monotonicity is reversed, from increasing to decreasing and vice versa.

COROLLARY 2.5. *Let $w(x;\alpha)$ be symmetric on $[-a, a]$, $0 < a \le \infty$, i.e., $w(-x;\alpha) = w(x;\alpha)$ for $0 \le x \le a$, but otherwise as in Theorem 2.3. Then each positive zero of $\pi_n^\alpha$ is increasing* (*decreasing*) *on* $(\alpha_1, \alpha_2)$ *depending on whether* (2.8) *is increasing* (*decreasing*) *on* $[0, a]$.

*Proof.* Suppose first that $n = 2k$ is even. Then, as is well known (see, e.g., [5, Theorem 1.18]),

$$\pi_{2k}^\alpha(x;\alpha) = \pi_k^+(x^2;\alpha),$$

where $\pi_k^+(\,\cdot\,;\alpha)$ is orthogonal on $[0, a^2]$ with respect to the weight function $w^+(t;\alpha) = t^{-1/2}w(t^{1/2};\alpha)$ on $[0, a^2]$. Now the positive zeros of $\pi_{2k}^\alpha(x;\alpha)$ are the positive square roots of the zeros of $\pi_k^+(\,\cdot\,;\alpha)$, hence increasing (decreasing) on $[\alpha_1, \alpha_2]$ if the same is true for the zeros of $\pi_k^+(\,\cdot\,;\alpha)$. But

$$\frac{1}{w^+(t;\alpha)}\,\frac{\partial w^+(t;\alpha)}{\partial\alpha} = \frac{t^{-1/2}}{t^{-1/2}w(t^{1/2};\alpha)}\,\frac{\partial w(t^{1/2};\alpha)}{\partial\alpha} = \frac{1}{w(t^{1/2};\alpha)}\,\frac{\partial w(t^{1/2};\alpha)}{\partial\alpha},$$

from which Corollary 2.5 follows.

For odd $n$, the proof is similar, using $\pi_{2k+1}^\alpha(x;\alpha) = x\pi_k^-(x^2;\alpha)$, where $\pi_k^-(\,\cdot\,;\alpha)$ is orthogonal on $[0, a^2]$ with respect to the weight function $w^-(t) = t^{1/2}w(t^{1/2};\alpha)$. $\square$

For later purposes, we consider the case where the parameter is not contained in the weight function, but is the upper limit of the interval of orthogonality, i.e., the (monic) polynomials $\{\pi_k\}$ are orthogonal on $[a, c]$, $-\infty \le a < c < \infty$, with respect to a weight function $w$,

$$\int_a^c \pi_k(x)\pi_\ell(x)w(x)\mathrm{d}x = 0, \quad k \ne \ell.$$

THEOREM 2.6. *Let $w(x)$ be a positive weight function on $[a, c]$, $-\infty \le a < c < \infty$, having finite moments $\mu_k$ for $0 \le k \le 2n-1$. Then each zero $x_\nu = x_\nu(c)$ of $\pi_n$ is monotonically increasing as a function of $c$.*

*Proof.* The proof follows the same line of arguments as the proof of Markov's theorem given in [14, Theorem 6.12.1], being based on the Gauss quadrature formula

$$\int_a^c p(x)w(x)\mathrm{d}x = \sum_{\mu=1}^n \lambda_\mu(c)\,p(x_\mu(c)), \quad p \in \mathbb{P}_{2n-1}. \tag{2.9}$$

Differentiating (2.9) with respect to $c$, we have

$$p(c)w(c) = \sum_{\mu=1}^n \lambda_\mu(c)\,p'(x_\mu(c))\,\frac{\mathrm{d}x_\mu}{\mathrm{d}c} + \sum_{\mu=1}^n \frac{\mathrm{d}\lambda_\mu}{\mathrm{d}c}\,p(x_\mu(c)). \tag{2.10}$$

Let

$$p(x) = \frac{\pi_n^2(x)}{x - x_\nu}\,, \quad p'(x_\nu) = [\pi_n'(x_\nu)]^2.$$

Then, since $p(x_\mu) = 0$ for all $\mu$ and $p'(x_\mu) = 0$ for $\mu \neq \nu$, we get from (2.10) that

$$\frac{\pi_n^2(c)}{c - x_\nu}\, w(c) = \lambda_\nu(c)\, [\pi_n'(x_\nu)]^2\, \frac{\mathrm{d}x_\nu}{\mathrm{d}c}\,. \tag{2.11}$$

Since on the right, both factors multiplying $\mathrm{d}x_\nu/\mathrm{d}c$ are positive, and on the left, $w(c) > 0$, $x_\nu < c$, it follows that $\mathrm{d}x_\nu/\mathrm{d}c > 0$. □

REMARK 2.7. Theorem 2.6 is valid also if $c$ is the lower limit of the orthogonality interval, by the same proof. Indeed, the left-hand side of Eq. (2.10) will then have a minus sign in front of it, and so does the left-hand side of Eq. (2.11). But now, $x_\nu > c$.

Returning to Theorem 2.3, the weight function is $w_1(x;\alpha)$ in (1.2), which is clearly symmetric on $(-\infty, \infty)$, so that according to Corollary 2.5 of Markov's theorem, the positive zeros of the generalized Binet polynomial $\pi_n^\alpha$ are increasing (decreasing) depending on whether

$$\frac{1}{w_1(x;\alpha)}\, \frac{\partial w_1(x;\alpha)}{\partial \alpha} = \frac{-1}{(\mathrm{e}^x - \alpha)\log(1 - \alpha \mathrm{e}^{-x})} \tag{2.12}$$

is increasing (decreasing) for $x$ in $(0, \infty)$.

Let the right-hand side of (2.12), as a function of $x$, be denoted by $f(x)$ and the denominator by $g(x)$. Then

$$f(x) = \frac{-1}{g(x)}\,, \quad f'(x) = \frac{g'(x)}{g^2(x)}\,.$$

So the matter depends on whether $g'(x)$ is positive (negative) on $[0, \infty)$.

Using the product rule of differentiation, we have

$$\begin{aligned} g'(x) &= (\mathrm{e}^x - \alpha)\, \frac{\alpha \mathrm{e}^{-x}}{1 - \alpha \mathrm{e}^{-x}} + \mathrm{e}^x \log(1 - \alpha \mathrm{e}^{-x}) \\ &= (\mathrm{e}^x - \alpha)\, \frac{\alpha}{\mathrm{e}^x - \alpha} + \mathrm{e}^x \log(1 - \alpha \mathrm{e}^{-x}) \\ &= \alpha + \mathrm{e}^x \log(1 - \alpha \mathrm{e}^{-x}) \\ &= \mathrm{e}^x [\alpha \mathrm{e}^{-x} + \log(1 - \alpha \mathrm{e}^{-x})]. \end{aligned}$$

Letting $t = \alpha \mathrm{e}^{-x}$, $0 < t < 1$, and $y(t) = t + \log(1 - t)$, we have $y(0) = 0$ and $y'(t) = -t/(1 - t) < 0$, so that $y(t) < 0$ on $(0, 1)$, i.e., the function in brackets is negative for $x$ in $(0, \infty)$, that is, $g'(x) < 0$. Thus we have

THEOREM 2.8. *All positive zeros of the generalized Binet polynomial $\pi_n^\alpha$ are monotonically decreasing as functions of $\alpha$.*

In order to plot the zeros, we first use the Matlab routine `dig_gbinet.m` to determine the number `dig` of digits needed to obtain the first 30 recurrence coefficients to an accuracy of 6 digits (more than enough for plotting purposes). The result, for any $\alpha$ in $(0, 1]$, is `dig` $= 16$. Once the respective variable-precision array `ab` has been obtained, one can revert to double precision for the rest of the computations.

**ETNA**
**Kent State University and**
**Johann Radon Institute (RICAM)**



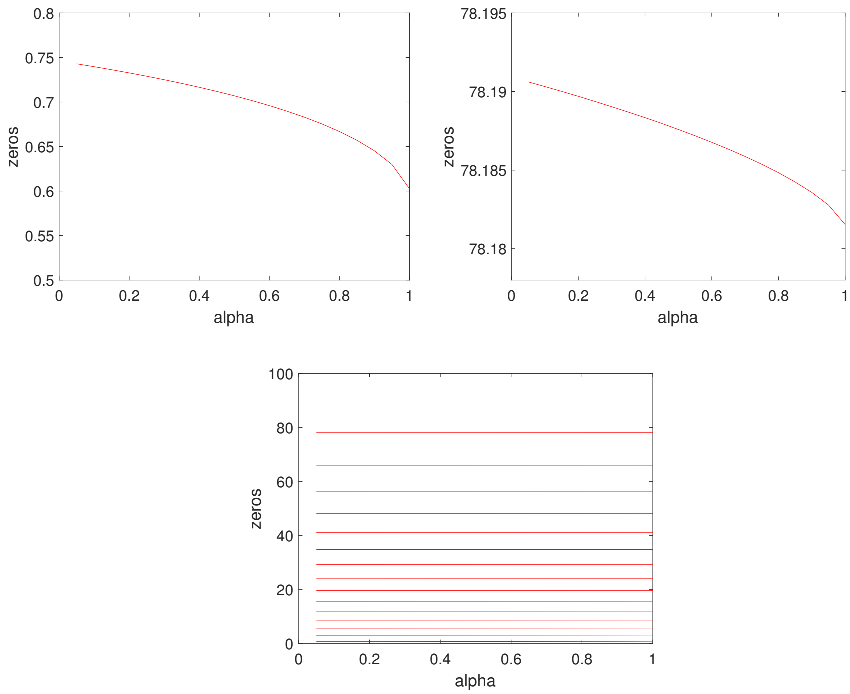


FIG. 2.3. *The positive zeros for $n = 30$ of the generalized Binet polynomial $\pi_n^\alpha$ in dependence of the parameter $\alpha$, $0 < \alpha \le 1$; the smallest and largest positive zero (top), all positive zeros (bottom).*

The Matlab script `plot_zeros_gbinet.m` with $\mathrm{N} = 30$ plots the 15 positive zeros of $\pi_n^\alpha$, and at the same time verifies their monotonic descent as functions of $\alpha$. That descent is relatively slow, almost imperceptible; see the third plot in Fig. 2.3. The first two plots show the smallest and largest positive zero, plotted in a scale that makes their monotone descent visible. The plots, indeed, suggest not only monotonicity, but also concavity, and perhaps even complete monotonicity. In general, monotonicity was found to be consistently weaker the larger the zero. For example, when $n = 30$, the relative decrement of the smallest zero varies in absolute value between $4.41\times10^{-3}$ and $4.52\times10^{-1}$, whereas the one for the largest zero varies between $3.83\times10^{-6}$ and $1.59\times10^{-5}$.

**3. Squared Binet and squared generalized Binet weight functions.** Moment-related methods and their implementation, both in Matlab and Mathematica, are considered in Section 3.1 for the squared Binet weight function, and in Section 3.2.1 for squared generalized Binet weight functions. Section 3.2.2 deals with zeros of squared generalized Binet polynomials.

**3.1. Squared Binet weight function.** The *squared Binet weight function*,

$$w_2(x) = \log^2(1 - \mathrm{e}^{-|x|}) \quad \text{on } [-\infty, \infty], \tag{3.1}$$

being symmetric, has the moments

$$\mu_k = \begin{cases} 0 & \text{if } k \text{ is odd,} \\ 2\displaystyle\int_0^\infty x^k \log^2(1 - \mathrm{e}^{-x})\mathrm{d}x & \text{if } k \text{ is even.} \end{cases} \tag{3.2}$$

**ETNA**
**Kent State University and**
**Johann Radon Institute (RICAM)**



Putting $e^{-x} = t$ in the integral of (3.2), we get

$$\mu_k = 2(-1)^k \int_0^1 \log^k t \, \log^2(1-t)\, \frac{\mathrm{d}t}{t}\,.$$

For $k = 0$ we have $\mu_0 = 4\,\zeta(3)$ ([9, Eq. 4.261.12 for $n = 0$]) while for $k$ (even) $> 0$

$$\mu_k = 4\,k!\, S_{k+1,2}(1),$$

with $S_{n,p}$ as defined in (2.3). We have ([11, pp. 39, 41] or [10, p. 1236])

$$s_n = S_{n-1,1}(1) = \sum_{j=1}^{\infty} \frac{1}{j^n} = \zeta(n)$$

and [11, Eq. (4.16)]

$$S_{n-1,2}(1) = \tfrac{1}{2}\, n s_{n+1} - \tfrac{1}{2}\,(s_2 s_{n-1} + s_3 s_{n-2} + \cdots + s_{n-1} s_2),$$

so that

$$\mu_k = 2k!\left[(k+2)\zeta(k+3) - \sum_{\nu=2}^{k+1} \zeta(\nu)\zeta(k+3-\nu)\right], \quad k(\text{even}) \;> 0. \tag{3.3}$$

The first N moments (3.3) are generated in `dig`-digit arithmetic by the Matlab command `mom=smom_sqbinet(dig,N)`.

EXAMPLE 3.1. The first 100 recurrence coefficients to 32 digits of the squared Binet weight function.

The Matlab command `[ab,dig]=dig_sqbinet(N,dig0,dd,nofdig)`, when run with `N`$=$ 100, `dig0` $=$ 108, `dd` $=$ 4, `nofdig` $=$ 32, yields `dig` $=$ 116. The same command, or more directly, the command `ab=sr_sqbinet(dig,nofdig,N)` with `dig` $= 116$, produces the $100{\times}2$ array `ab` of the first 100 recurrence coefficients to 32 digits. They are depicted in the second and third plot of Fig. 3.1, the first showing the squared Binet

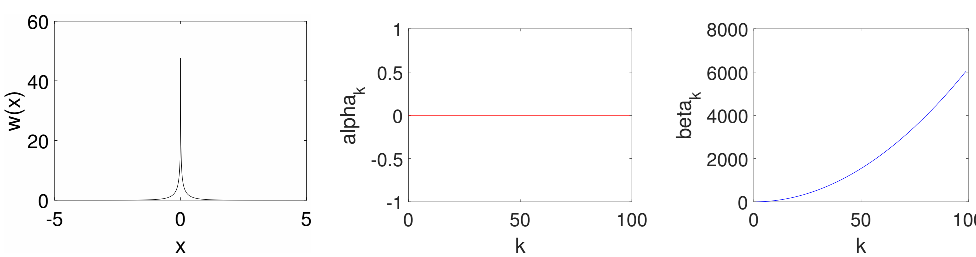


FIG. 3.1. *Squared Binet weight function and its recurrence coefficients.*

weight function. They are also made available in the textfile `coeff_sqbinet.txt`; see also [8, 2.3.9].

Symbolic computation in Mathematica of the first 200 moments is accomplished by the command

```
momSGB = Table[If[k == 0, 4 Zeta[3], If[OddQ[k], 0, 2k!
              ((k+2)Zeta[k+3]-Sum[Zeta[v]Zeta[k+3-v],
              {v,2,k+1}])]], {k,0,199}];
```

taking 170.1 ms to run. With the numerical calculation option in Chebyshev's algorithm, using working precision `WP` $= 108$, yields the first 100 recurrence coefficients to 32 digits in 136.2 ms, with `WP` $= 136$ to 60 digits in 147.7 ms, and with `WP` $= 196$ to 120 digits in 168.4 ms.

### 3.2. Squared generalized Binet weight function.

**3.2.1. Recurrence coefficients.** The *squared generalized Binet weight function*,

$$w_2(x;\alpha) = \log^2(1-\alpha \mathrm{e}^{-|x|}) \quad \text{on } [-\infty,\infty],\ 0<\alpha<1, \tag{3.4}$$

has the moments

$$\mu_k = \begin{cases} 0 & \text{if } k \text{ is odd,} \\ 2\displaystyle\int_0^\infty x^k \log^2(1-\alpha \mathrm{e}^{-x})\mathrm{d}x & \text{if } k \text{ is even.} \end{cases} \tag{3.5}$$

Similarly as in Subsection 3.1, one finds

$$\mu_k = 4\,k!\,S_{k+1,2}(\alpha), \tag{3.6}$$

where $S_{n,p}(x)$ is the Nielsen generalized polylogarithm function (2.3). We have [10, Eq. (2.9)]

$$S_{k+1,2}(\alpha) = \sum_{\nu=1}^{\infty}\left(\sum_{\mu=1}^{\nu}\frac{1}{\mu}\right)\frac{\alpha^{\nu+1}}{(\nu+1)^{k+2}}\,, \quad 0<\alpha<1. \tag{3.7}$$

The series converges fairly rapidly for all $k \geq 0$ provided $\alpha$ is not too close to 1.

The moments (3.5) are generated by the Matlab command `mom=smom_sqg_binet (dig,N,a)`.

EXAMPLE 3.2. The first 100 recurrence coefficients to 32 digits of the squared generalized Binet weight function for $\alpha = 1/2$.

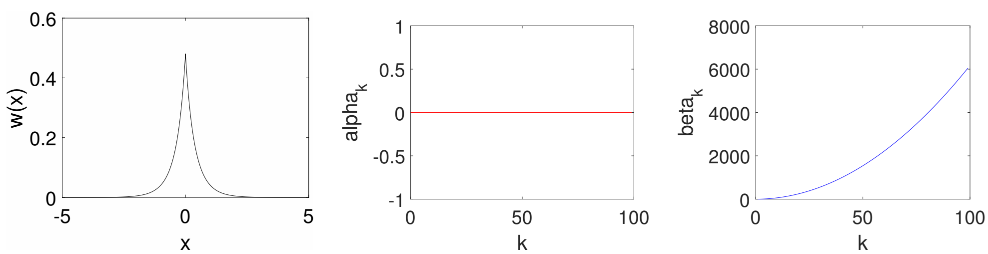


FIG. 3.2. *Squared generalized Binet weight function for $\alpha = 1/2$ and its recurrence coefficients.*

The Matlab command `[ab,dig]=dig_sqgbinet(N,a,dig0,dd,nofdig)`, when run with `N` $= 100$, `a` $= 1/2$, `dig0` $= 56$, `dd` $= 4$, `nofdig` $= 32$, yields `dig` $= 64$. The same command, or more directly, the command `ab=sr_sqgbinet(dig,nofdig,N,a)` with `dig` $= 64$, produces the $100\times 2$ array `ab` of the first 100 recurrence coefficients to 32 digts. They are depicted in the second and third plot of Fig. 3.2, the first showing the squared generalized Binet weight function for $\alpha = 1/2$. They are also made available in the textfile `coeff_sqgbinet.txt`; see also [8, 2.3.11].

Symbolic computation in Mathematica of the first 200 moments is accomplished by the command

**ETNA**
Kent State University and
Johann Radon Institute (RICAM)



```
momSGB=Table[If[OddQ[k],0,4k!PolyLog[k+1,2,1/2]],{k,0,199}];
```

taking 3.3 ms to run (equally fast for every $\alpha < 1$). With the numerical calculation option in Chebyshev's algorithm, using working precision `WP` $= 64$, the first 100 recurrence coefficients to 32 digits are obtained in 2.12 s, with `WP` $= 92$ to 60 digits in 2.90 s, and with `WP` $= 152$ to 120 digits in 5.55 s.

**3.2.2. Zeros of the orthogonal polynomials.** By virtue of Corollary 2.4 to Markov's theorem and of what was proved in Subsection 2.2.2, we have

THEOREM 3.3. *All positive zeros of the squared generalized Binet polynomial $\pi_n^\alpha$ are monotonically decreasing as functions of $\alpha$.*

The zeros behave similarly as those for the generalized Binet polynomials, but are only about half as large. For $n = 30$, the smallest and largest positive zero are shown in the first two plots of Fig. 3.3, and all 15 positive zeros in the third plot; cf. `plot_zeros_sqgbinet.m`.

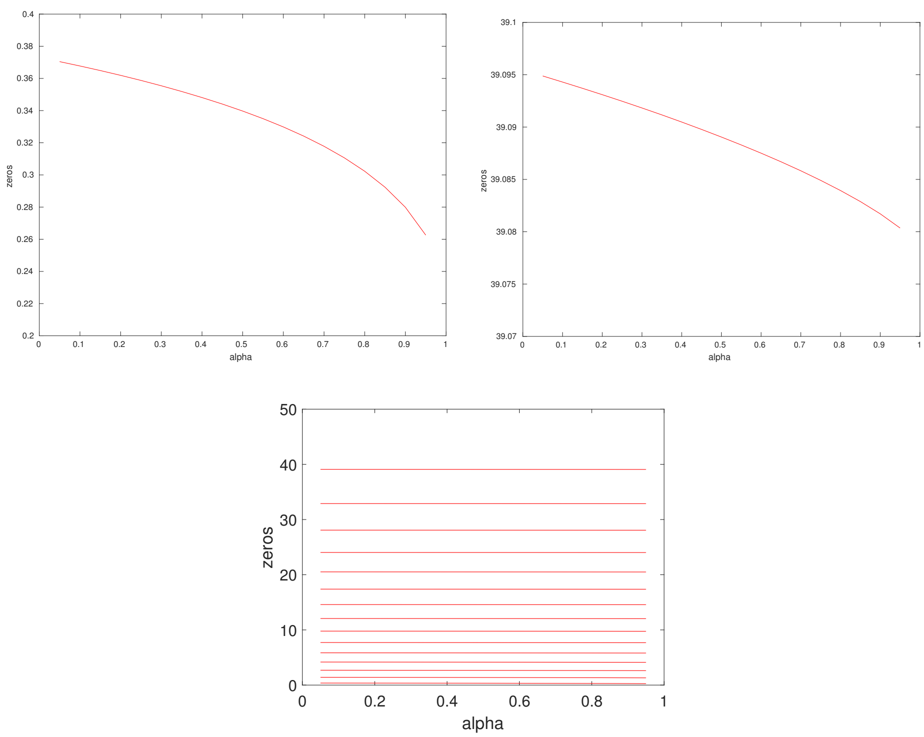


FIG. 3.3. *The positive zeros for $n = 30$ of the squared generalized Binet polynomials in dependence of the parameter $\alpha$, $0 < \alpha < 1$; the smallest and largest positive zero (top), all positive zeros (bottom).*

**4. Halfrange Binet and halfrange generalized Binet weight functions.** Moment-related methods and their implementation in Matlab are considered in Section 4.1 for the halfrange Binet weight function and in Section 4.2.1 for halfrange generalized Binet weight functions. Section 4.2.2 deals with zeros of halfrange generalized Binet polynomials.

**4.1. Halfrange Binet weight function.** The halfrange Binet weight function is the weight function (1.1) supported on $[0, \infty]$. Its moments are (cf. Eqs. (2.1), (2.2))

$$\mu_k = k!\,\zeta(k+2), \quad k = 0, 1, 2, \ldots\ . \tag{4.1}$$

They are generated by the Matlab command `mom=smom_hrbinet(dig,N)`.

EXAMPLE 4.1. The first 100 recurrence coefficients to 32 digits of the halfrange Binet weight function.

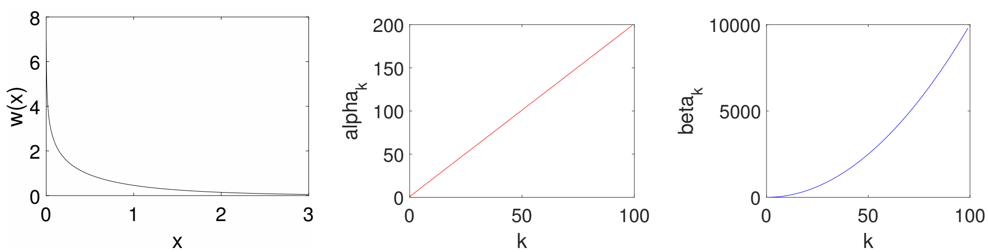


FIG. 4.1. *Halfrange Binet weight function and its recurrence coefficients.*

The Matlab command `[ab,dig]=dig_hrbinet(N,dig0,dd,nofdig)`, when run with `N` $= 100$, `dig0` $= 116$, `dd` $= 4$, `nofdig` $= 32$, yields `dig` $= 124$. The same command, or more directly, the command `ab=sr_hrbinet(dig, nofdig,N)` with `dig` $= 124$, produces the $100{\times}2$ array `ab` of the first 100 recurrence coefficients to 32 digits. They are depicted in the second and third plot of Fig. 4.1, the first showing the halfrange Binet weight function. They are also made available in the textfile `coeff_hrbinet.txt`; see also [8, 2.9.26].

**4.2. Halfrange generalized Binet weight function.**

**4.2.1. Recurrence coefficients.** The halfrange generalized Binet weight function is the weight function (1.2) supported on $[0, \infty]$. Its moments are (cf. Eqs. (2.6), (2.7))

$$\mu_k = k!\,S_{k+1,1}(\alpha) = k!\,\mathrm{Li}_{k+2}(\alpha), \quad k = 0, 1, 2, \ldots\ . \tag{4.2}$$

They are generated by the Matlab command `mom=smom_hrgbinet(dig,N,a)`, `a` $= \alpha$.

EXAMPLE 4.2. The first 100 recurrence coefficients to 32 digits of the halfrange generalized Binet weight function for $\alpha = 1/2$.

The Matlab command `[ab,dig]=dig_hrgbinet(N,a,dig0,dd,nofdig)`, when run with `N` $= 100$, `a` $= 1/2$, `dig0` $= 120$, `dd` $= 4$, `nofdig` $= 32$, yields `dig` $= 128$. The same command, or more directly, the command `ab=sr_hrgbinet(dig,nofdig, N,a)` with `dig` $= 128$, produces the $100{\times}2$ array `ab` of the first 100 recurrence coefficients to 32 digits. They are depicted in the second and third plot of Fig. 4.2, the first showing the halfrange generalized Binet weight function for $\alpha = 1/2$. They are also made available in the textfile `coeff_hrgbinet.txt`; see also [8, 2.9.30].

**4.2.2. Zeros of the orthogonal polynomials.** By what was proved in Subsection 2.2.2, we have

THEOREM 4.3. *All zeros of the halfrange generalized Binet polynomial $\pi_n^\alpha$ are monotonically decreasing as functions of $\alpha$.*

For $n = 15$, the smallest and largest zero are shown in the first two plots of Fig. 4.3, and all zeros in the third plot; cf. `plot_zeros_hrgbinet.m`.

**ETNA**
**Kent State University and**
**Johann Radon Institute (RICAM)**



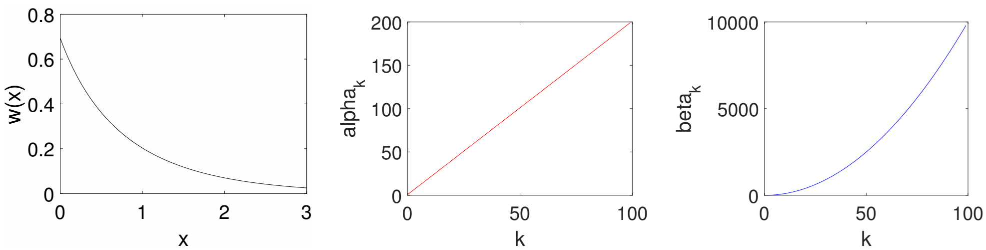


FIG. 4.2. *Halfrange generalized Binet weight function for $\alpha = 1/2$ and its recurrence coefficients.*

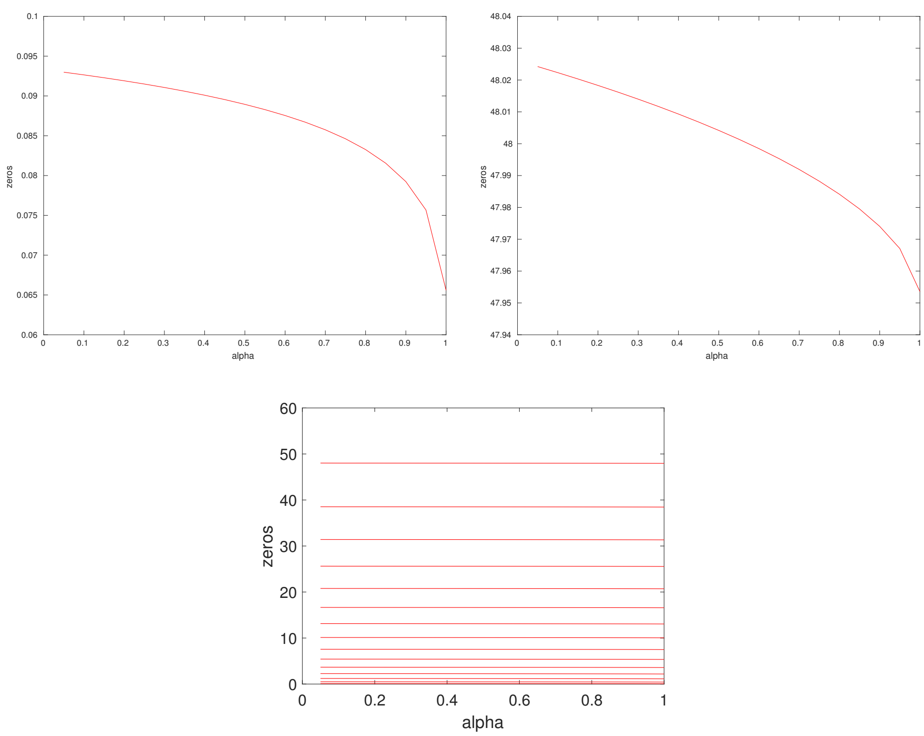


FIG. 4.3. *The zeros for $n = 15$ of the halfrange generalized Binet polynomials in dependence of the parameter $\alpha$, $0 < \alpha \le 1$; the smallest and largest zero (top), all zeros (bottom).*

**5. Halfrange squared Binet and halfrange squared generalized Binet weight functions.** Moment-related methods and their implementation in Matlab are considered in Section 5.1 for the halfrange squared Binet weight function and in Section 5.2.1 for halfrange squared generalized Binet weight functions. Section 5.2.2 deals with zeros of halfrange squared generalized Binet polynomials.

**5.1. Halfrange squared Binet weight function.** The halfrange squared Binet weight function is the weight function (3.1) supported on $[0, \infty]$. Its moments are (cf. Eqs. (3.2), (3.3))

$$\begin{aligned} &\mu_0 = 2\,\zeta(3), \\ &\mu_k = k!\left[(k+2)\zeta(k+3) - \sum_{\nu=2}^{k+1} \zeta(\nu)\zeta(k+3-\nu)\right], \quad k = 1, 2, 3, \ldots\,. \end{aligned} \tag{5.1}$$

**ETNA**
Kent State University and
Johann Radon Institute (RICAM)



The first `N` of them are generated in `dig`-digit arithmetic by the Matlab command `mom=smom_hrsqbinet(dig,N)`.

EXAMPLE 5.1. The first 100 recurrence coefficients to 32 digits of the halfrange squared Binet weight function.

The Matlab command `[ab,dig]=dig_hrsqbinet(N,dig0,dd,nofdig)`, when run with `N` $=$ 100, `dig0` $=$ 160, `dd` $=$ 4, `nofdig` $=$ 32, yields `dig` $=$ 168. The same command, or more directly, the command `ab=sr_hrsqbinet(dig,nofdig,N)` with `dig` $=$ 168, produces the $100\times 2$ array `ab` of the first 100 recurrence coefficients to 32 digits. They are depicted in the second and third plot of Fig. 5.1, the first showing the halfrange squared Binet weight function. They are also made available in the textfile `coeff_hrsqbinet.txt`; see also [8, 2.9.31].

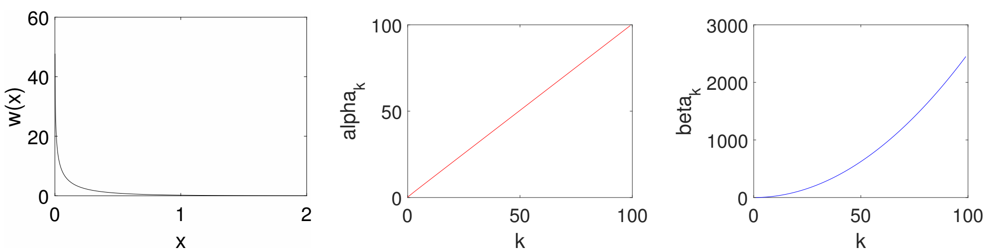


FIG. 5.1. *Halfrange squared Binet weight function and its recurrence coefficients.*

### 5.2. Halfrange squared generalized Binet weight function.

**5.2.1. Recurrence coefficients.** The halfrange squared generalized Binet weight function is the weight function (3.4) supported on $[0,\infty]$. Its moments are (cf. Eqs. (3.5), (3.6))

$$\mu_k = 2\,k!S_{k+1,2}(\alpha), \quad k=0,1,2,\dots\,. \tag{5.2}$$

They are generated by the Matlab command `mom=smom_hrsqgbinet(dig,N,a)`.

EXAMPLE 5.2. The first 100 recurrence coefficients to 32 digits of the halfrange squared generalized Binet weight function for $\alpha=1/2$.

The Matlab command `[ab,dig]=dig_hrsqgbinet(N,a,dig0,dd,nofdig)`, when run with `N` $=100$, `a` $=1/2$, `dig0` $=116$, `dd` $=4$, `nofdig` $=32$, yields `dig` $=124$. The same command, or more directly, the command `ab=sr_hrsqgbinet(dig,nofdig,N,a)` with `dig` $=124$, produces the $100\times 2$ array `ab` of the first 100 recurrence coefficients to 32 digits. They are depicted in the second and third plot of Fig. 5.2, the first showing the halfrange squared generalized Binet weight function for $\alpha=1/2$. They are also made available in the textfile `coeff_hrsqgbinet.txt`; see also [8, 2.9.32].

**5.2.2. Zeros of the orthogonal polynomials.** Since, according to Subsection 4.2.2, all zeros of the halfrange generalized Binet polynomial are monotonically decreasing, the same is true, by Corollary 2.4 of Markov's theorem (cf. Subsection 2.2.2), for the square of the weight function. Thus we have

THEOREM 5.3. *All zeros of the halfrange squared generalized Binet polynomial $\pi_n^\alpha$ are monotonically decreasing as functions of $\alpha$.*

For $n=15$, the smallest and largest zero are shown in the first two plots of Fig. 5.3, and all zeros in the third plot; cf. `plot_zeros_hrsqgbinet.m`.

**ETNA**
**Kent State University and**
**Johann Radon Institute (RICAM)**



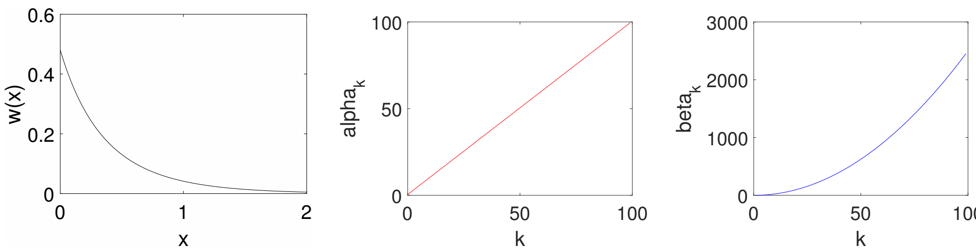


FIG. 5.2. *Halfrange squared generalized Binet weight function for $\alpha = 1/2$ and its recurrence coefficients*

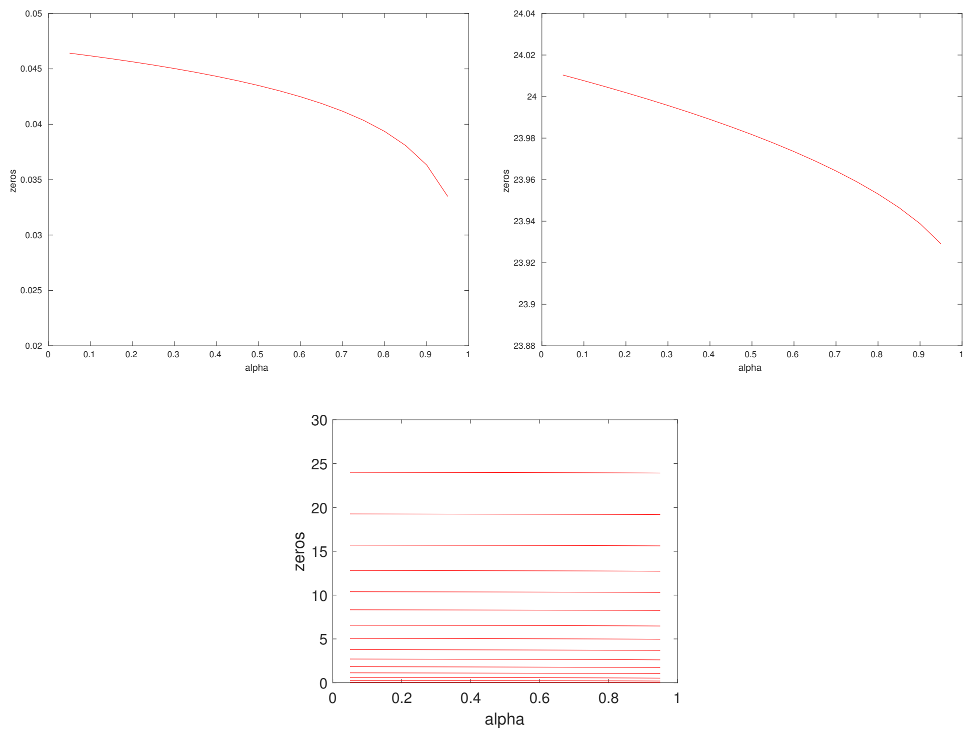


FIG. 5.3. *The zeros for $n = 15$ of the halfrange squared generalized Binet polynomials in dependence of the parameter $\alpha$, $0 < \alpha < 1$; the smallest and largest zero (top), all zeros (bottom).*

**6. Subrange Binet weight functions.** Moment-related methods and their implementation in Matlab are considered in Section 6.1.1 for an upper subrange Binet weight function, in Section 6.2.1 for a lower subrange Binet weight function, and in Section 6.3.1 for a lower symmetric subrange Binet weight function. Sections 6.1.2, 6.2.2, and 6.3.2 deal with the zeros of the respective subrange Binet polynomials.

### 6.1. An upper subrange Binet weight function.

**6.1.1. Recurrence coefficients.** The weight function (1.1) is now assumed to be supported on the interval $[c, \infty]$, $0 < c < \infty$. The approach via moments,

$$\mu_k = -\int_c^\infty x^k \log(1 - e^{-x})\mathrm{d}x,$$

is still a valid option, giving, with the substitution of variables $t = e^{c-x}$,

$$\mu_k = \sum_{\nu=0}^{k} k^{(\nu)} c^{k-\nu} \mathrm{Li}_{\nu+2}(e^{-c}), \quad k = 0, 1, 2, \dots , \tag{6.1}$$

where

$$k^{(\nu)} = \begin{cases} 1 & \text{if } \nu = 0, \\ k(k-1)\cdots(k-\nu+1) & \text{if } \nu > 0, \end{cases}$$

is the descending factorial power and $\mathrm{Li}_n(x)$ the polylogarithm (cf. Subsection 2.1). The moments (6.1) are generated by the Matlab routine `smom_usrbinet.m`.

It is, however, considerably simpler, and hence faster, to make use of a linear translation of the upper subrange Binet weight function on $[c, \infty]$ to the halfrange generalized Binet weight function with parameter $\alpha = \mathrm{e}^{-c}$ (cf. Subsection 4.2). Denoting the recurrence coefficients of the latter by $a_k(\alpha)$, $b_k(\alpha)$, $k = 0, 1, 2, \dots$ , it is easy to see that

$$\alpha_k = a_k(\alpha) + c, \ \ \beta_k = b_k(\alpha), \quad k = 0, 1, 2, \dots, \ \ \alpha = \mathrm{e}^{-c}. \tag{6.2}$$

The moments needed to generate the $a_k(\alpha)$, $b_k(\alpha)$ are then those in (4.2), which are definitely simpler than those in (6.1). They are produced by the Matlab command `mom=smom_usrbinet_alt(dig,N,c)`.

EXAMPLE 6.1. The first 100 recurrence coefficients to 32 digits of the upper subrange Binet weight function for $c = 1$.

The Matlab command

```
[ab,dig]=dig_usrbinet_alt(N,c,dig0,dd,nofdig),
```

when run with `N` $= 100$, `c` $= 1$, `dig0` $= 120$, `dd` $= 4$, `nofdig` $= 32$, yields `dig` $= 128$. The same command, or more directly, the command

```
ab=sr_usrbinet_alt(dig,nofdig,N,c)
```

with `dig` $= 128$, `c` $= 1$, produces the $100{\times}2$ array `ab` of the first 100 recurrence coefficients to 32 digits. They are depicted in the second and third plot of Fig. 6.1, the first showing the upper subrange Binet weight function on $[1, \infty]$. They are also made available in the textfile `coeff_usrbinet.txt`; see also [8, 2.9.28].

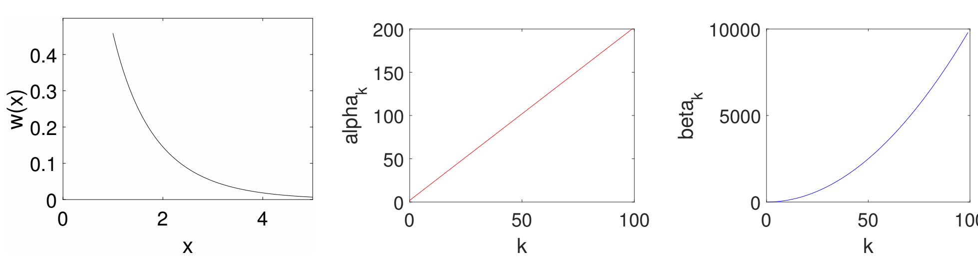


FIG. 6.1. *Upper subrange Binet weight function and its recurrence coefficients.*

**ETNA**
Kent State University and
Johann Radon Institute (RICAM)



**6.1.2. Zeros of the orthogonal polynomials.** Our interest is now in the behavior of the zeros of the upper subrange Binet polynomials as functions of $c$. Here, Remark 2.7 to Theorem 2.6 of Subsection 2.2.2 applies to give

THEOREM 6.2. *All zeros of the upper subrange Binet polynomial $\pi_n$, orthogonal on $[c,\infty]$, are monotonically increasing as functions of $c$.*

The zeros for $n = 15$ are shown in Fig. 6.2; cf. `plot_zeros_usrbinet.m`.

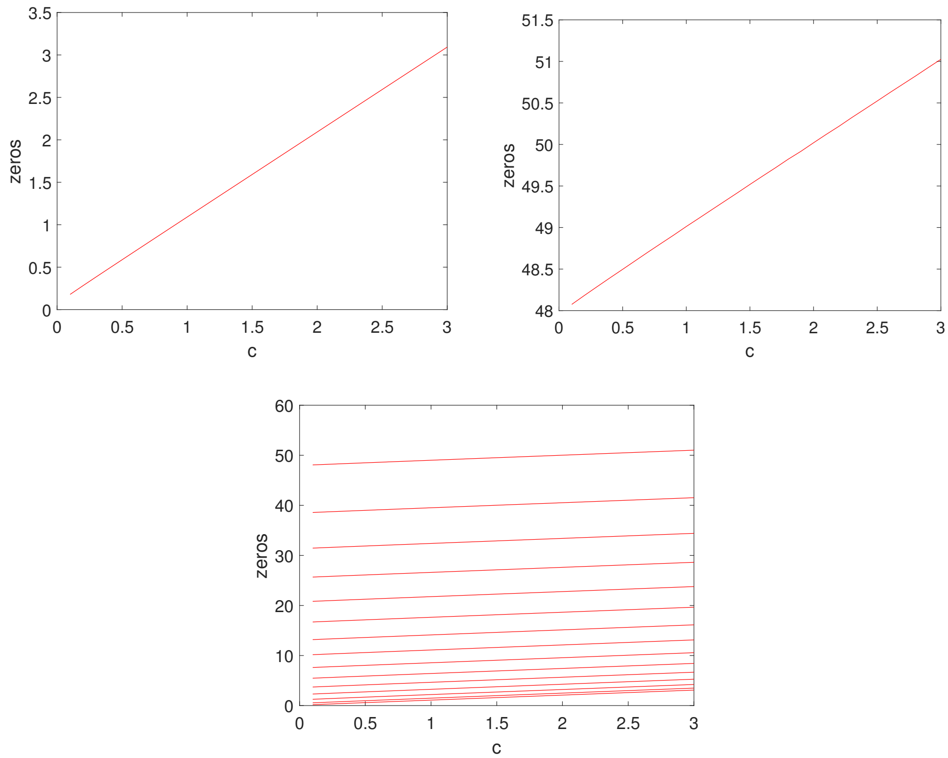


FIG. 6.2. *The zeros for $n = 15$ of the upper subrange Binet polynomials in dependence of the parameter $c$, $0 < c \le 3$; the smallest and largest zero (top), all zeros (bottom).*

### 6.2. A lower subrange Binet weight function.

**6.2.1. Recurrence coefficients.** We consider here the weight function (1.1) supported on the interval $[0, c]$, $0 < c < \infty$. We take the simple approach of computing the respective moments as the difference between the halfrange and upper subrange moments,

$$\mu_k = \mu_k^{\mathrm{hr}} - \mu_k^{\mathrm{usr}}(c), \quad k = 0, 1, 2, \dots \, , \tag{6.3}$$

where $\mu_k^{\mathrm{hr}}$ are the moments in (4.1), and $\mu_k^{\mathrm{usr}}(c)$ those in (6.1), although (6.3) may be subject to severe cancellation, especially if $c$ is small. This must be compensated by an increase of the precision used to compute the moments.

The moments (6.3) are generated in `dig`-digit arithmetic by the Matlab command `mom=smom_lsrbinet(dig,N,c)`.

EXAMPLE 6.3. The first 100 recurrence coefficients to 32 digits of the lower subrange Binet weight function on $[0, c]$ for $c = 1$.

**ETNA**
Kent State University and
Johann Radon Institute (RICAM)



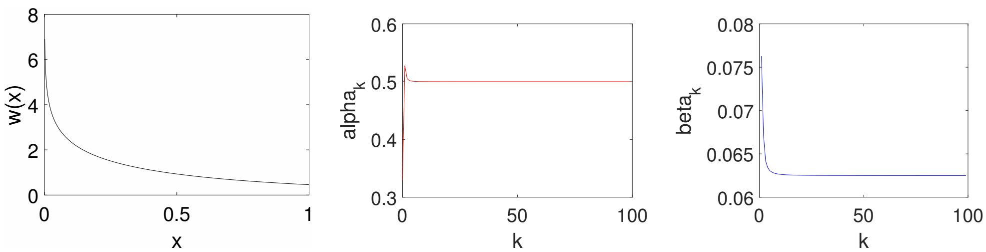


FIG. 6.3. *Lower subrange Binet weight function with $c = 1$ and its recurrence coefficients.*

The Matlab command `[ab,dig]=dig_lsrbinet(N,c,dig0,dd,nofdig)`, when run with $\texttt{N} = 100$, $\texttt{c} = 1$, $\texttt{dig0} = 520$, $\texttt{dd} = 4$, $\texttt{nofdig} = 32$, yields $\texttt{dig} = 528$. This large number of `dig` is due to extremely severe cancellation in (6.3), causing a loss of as many as 375 digits! The same command, or more directly, the command `ab=sr_lsrbinet (dig,nofdig,N,c)` with $\texttt{dig} = 528$, $\texttt{c} = 1$, produces the $100 \times 2$ array `ab` of the first 100 recurrence coefficients to 32 digits. They are depicted in the second and third plot of Fig. 6.3, the first showing the lower subrange Binet weight function on $[0, 1]$. They are also made available in the textfile `coeff_lsrbinet.txt`; see also [8, 2.9.27].

**6.2.2. Zeros of the orthogonal polynomials.** By Theorem 2.6 of Subsection 2.2.2, we have

THEOREM 6.4. *All zeros of the lower subrange Binet polynomial $\pi_n$, orthogonal on $[0, c]$, are monotonically increasing as functions of $c$.*

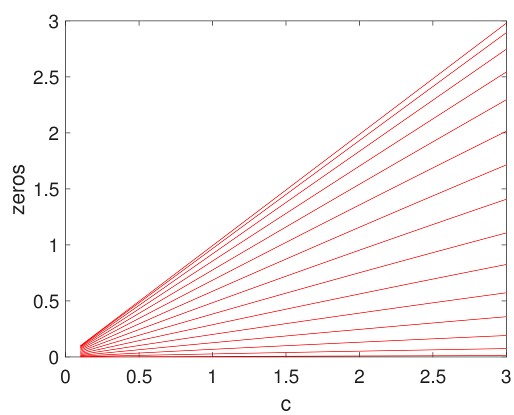


FIG. 6.4. *The zeros for $n = 15$ of the lower subrange Binet polynomial $\pi_n$, orthogonal on $[0, c]$, in dependence of the parameter $c$, $0 < c \leq 3$.*

Using the routine `dig_lsrbinet.m` with $\texttt{N} = 15$, it was found that $\texttt{dig} = 90$ digits are required to obtain the first 15 recurrence coefficients to an accuracy of 6 digits whenever $c \geq 1/10$. The zeros obtained are shown in Fig. 6.4; cf. `plot_zeros_lsrbinet.m`.

**ETNA**
Kent State University and
Johann Radon Institute (RICAM)



### 6.3. A lower symmetric subrange Binet weight function.

**6.3.1. Recurrence coefficients.** Here, the weight function (1.1) is supported on the interval $[-c, c]$, $0 < c < \infty$. The moments $\mu_k$, therefore, are 0 if $k$ is odd, and twice those in (6.3) if $k$ is even, and are generated by the routine `smom_lssrbinet.m`.

EXAMPLE 6.5. The first 100 recurrence coefficients to 32 digits of the lower symmetric subrange Binet weight function on $[-c, c]$ for $c = 1$.

The Matlab routine `dig_lssrbinet(N,c,dig0,dd,nofdig)`, run with `N` $= 100$, `c` $= 1$, `dig0` $= 460$, `dd` $= 4$, `nofdig` $= 32$, yields `dig` $= 468$ as the number of digits needeed to obtain the $100{\times}2$ array `ab` of the desired recurrence coefficients to 32 digits. The same array, more directly, can be obtained by the command `ab=sr_lssrbinet(dig, nofdig,N,c)` with `dig` $= 468$, `c` $= 1$. The recurrence coefficients are depicted in the second and third plot of Fig. 6.5, the first showing the lower symmetric subrange Binet weight function on $[-1, 1]$, and to 32 digits are made available in the textfile `coeff_lssrbinet.txt`; see also [8, 2.9.29].

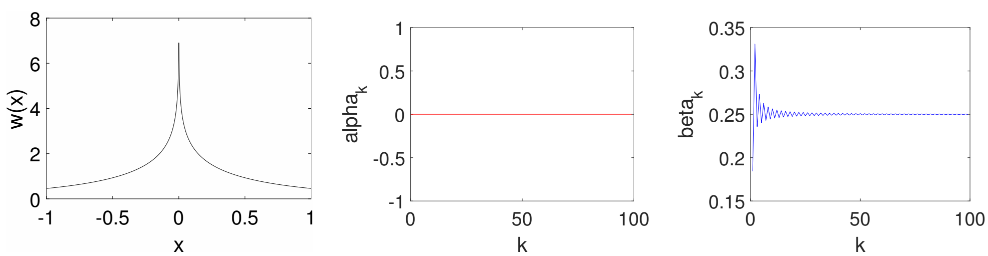


FIG. 6.5. *Lower symmetric subrange Binet weight function with $c = 1$ and its recurrence coefficients.*

**6.3.2. Zeros of the orthogonal polynomials.** Since the lower symmetric subrange Binet weight function is symmetric on $[-c, c]$, we can apply the Remark to Theorem 2.3 in [7] to obtain

THEOREM 6.6. *All positive zeros of the lower symmetric subrange Binet polynomial $\pi_n$, orthogonal on $[-c, c]$, $0 < c < \infty$, are monotonically increasing as functions of $c$.*

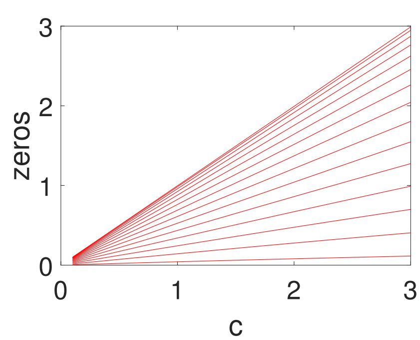


FIG. 6.6. *The positive zeros for $n = 30$ of the lower symmetric subrange Binet polynomial $\pi_n$, orthogonal on $[-c, c]$, in dependence of the parameter $c$, $0 < c \leq 3$.*

Plots of the positive zeros for $n = 30$ are shown in Fig. 6.6; cf. `plot_zeros_lssrbinet.m`.

## REFERENCES

[1] A. CVETKOVIĆ AND G. V. MILOVANOVIĆ, *The Mathematica package "OrthogonalPolynomials"*, Facta Univ. Ser. Math. Inform. 19 (2004), pp. 17–36.
[2] G. DAHLQUIST, *On summation formulas due to Plana, Lindelöf and Abel, and related Gauss–Christoffel rules, I.* BIT 37 (1997), pp. 256–295.
[3] G. DAHLQUIST, *On summation formulas due to Plana, Lindelöf and Abel, and related Gauss–Christoffel rules, II.* BIT 37 (1997), pp. 804–832.
[4] G. DAHLQUIST, *On summation formulas due to Plana, Lindelöf and Abel, and related Gauss–Christoffel rules, III.* 39 (1999), pp. 51–78.
[5] W. GAUTSCHI, *Orthogonal Polynomials: Computation and Approximation*, Numerical Mathematics and Scientific Computation, Oxford University Press, Oxford, 2004.
[6] W. GAUTSCHI, *Orthogonal Polynomials in MATLAB: Exercises and Solutions*, Software, Environment, Tools, SIAM, Society of Industrial and Applied Mathematics, Philadelphia, PA, 2016.
[7] W. GAUTSCHI, *On the zeros of subrange Jacobi polynomials*, Numer. Algorithms, to appear.
[8] W. GAUTSCHI, *A Software Repository for Orthogonal Polynomials*, e-book, SIAM, Society of Industrial and Applied Mathematics, Philadelphia, PA, 2018.
[9] I. S. GRADSHTEYN AND I. M. RYZHIK. *Table of Integrals, Series, and Products*, 5th ed., Academic Press, New York, 1994.
[10] K. S. KÖLBIG, *Nielsen's generalized polylogarithms*, SIAM J. Math. Anal. 17 (1986), pp. 1232–1258.
[11] K. S. KÖLBIG, J. A. MIGNACO, AND E. REMIDDI, *On Nielsen's generalized polylogarithms and their numerical calculation*, BIT 10 (1970), pp. 38–73.
[12] G. V. MILOVANOVIĆ, *Summation formulas of Euler–Maclaurin and Abel-Plana: Old and new results and applications*, in Progress in Approximation Theory and Applicable Complex Analysis , N. K. Govil et al. eds., Springer Optimization and Its Applications, v. 117, Springer International Publishing AG, 2017, pp. 429–461.
[13] G. V. MILOVANOVIĆ AND A. S. CVETKOVIĆ, *Special classes of orthogonal polynomials and corresponding quadratures of Gaussian type*, Math. Balkanica 26 (2013), pp. 169–184.
[14] G. SZEGÖ, *Orthogonal Polynomials*, 4th ed., Colloquium Publications, v. 23, American Mathematical Society, Providence, RI, 1975.