NON-ITERATIVE COMPUTATION OF GAUSS-JACOBI QUADRATURE *

AMPARO GIL[†], JAVIER SEGURA [‡], AND NICO M. TEMME[§]

Abstract. Asymptotic approximations to the zeros of Jacobi polynomials are given, with methods to obtain the coefficients in the expansions. These approximations can be used as standalone methods for the non-iterative computation of the nodes of Gauss–Jacobi quadratures of high degree ($n \geq 100$). We also provide asymptotic approximations for functions related to the first order derivative of Jacobi polynomials which are used for computing the weights of the Gauss–Jacobi quadrature. The performance of the asymptotic approximations is illustrated with numerical examples, and it is shown that nearly double precision relative accuracy is obtained both for the nodes and the weights when $n \geq 100$ and $-1 < \alpha, \beta \leq 5$. For smaller degrees the approximations are also useful as they provide 10^{-12} relative accuracy for the nodes when $n \geq 20$, and just one Newton step would be sufficient to guarantee double precision accuracy in that cases.

Key words. Jacobi polynomials; asymptotic expansions; Gaussian quadratures.

AMS subject classifications. 33C45, 41A60, 65D20, 65D32

1. Introduction. Given a weight function $w(x)$ in an interval $[a, b]$, the Gaussian quadrature formula with n nodes $Q_n(f) = \sum_{i=1}^{n} w_i f(x_i)$ for approximating the integral $I(f) = \int_a^b f(x)w(x)\, dx$ is the formula with the highest possible degree of accuracy, that is, the formula such that $I(f) = Q_n(f)$ if f is any polynomial of degree smaller than $2n$. Because of the optimality with respect to the degree of accuracy, Gaussian rules have fast convergence as the degree increases (especially for analytic functions), and they are one of the most popular methods of numerical integration, appearing in countless applications. A particularly important set of quadrature rules are the Gauss-Jacobi rules, with weight functions $w(x) = (1-x)^\alpha(1+x)^\beta$, $\alpha, \beta > -1$ in $[a, b] = [-1, 1]$ (Gauss-Legendre quadrature is the particular case $\alpha = \beta = 0$).

The Golub-Welsch algorithm [21] is an interesting method for computing quadrature rules of low degree. However, as the number of nodes n increases, the complexity scales as $\mathcal{O}(n^2)$. Recent papers describing fast methods of order $\mathcal{O}(n)$ for the computation of classical Gauss quadratures are [20, 5, 23, 4, 39, 7, 19]. A nice short review describing recent developments in the computation of high degree Gauss-Legendre quadrature can be found in [38].

In [23], the method for Gauss-Jacobi quadrature relies on simple asymptotic approximations for the nodes which are iteratively refined by Newton's method, with the Jacobi polynomial (whose roots are the nodes of the formula) and its derivative computed by two types of available asymptotic expansions; the same approach was used in [39] for Gauss-Hermite quadrature. A similar strategy was considered earlier for Gauss-Legendre quadrature in [33, 20, 5], but using different methods for computing the Legendre polynomials; methods for Gauss-Laguerre and Gauss-Hermite were also described in [20]. A more recent approach, based in the integration of ODEs using

† Departamento de Matemática Aplicada y CC. de la Computación. ETSI Caminos. Universidad de Cantabria. 39005-Santander, Spain. (amparo.gil@unican.es).

‡Departamento de Matemáticas, Estadística y Computación. Facultad de Ciencias. Universidad de Cantabria. 39005-Santander, Spain. (javier.segura@unican.es).

§IAA, 1825 BD 25, Alkmaar, The Netherlands. Former address: Centrum Wiskunde & Informatica (CWI), Science Park 123, 1098 XG Amsterdam, The Netherlands (nico.temme@cwi.nl).

non-oscillatory phase representations is that of [7], which appears to be competitive for Gauss rules of very high degree. All these approaches have in common the use of an iterative root-finding method (Newton's method in all cases) for polishing the initial estimations for the roots.

However, for the case of Gauss-Legendre quadrature, it was shown in [4] that the direct computation of the nodes and weights by explicit asymptotic expansions was possible, without recourse to iterative methods, and that this non-iterative approach is accurate for moderate orders ($n \geq 100$) and faster than iterative methods. In [19] it was shown that for the other two "classical" quadrature formulas (Gauss-Hermite and Gauss-Laguerre) a purely asymptotic approach was also possible. Finally, in the present paper we consider the asymptotic computation of the nodes and weights of Gauss-Jacobi quadrature, which completes the description of large-degree asymptotic methods for the classical Gauss quadrature rules (Hermite, Laguerre, Jacobi).

The present paper extends the results in [4], where the case of the Gauss-Legendre quadrature ($\alpha = \beta = 0$) was considered, as well as those in [23], which described an iterative method producing close to double precision absolute accuracy for the nodes. We obtain a fast non-iterative method based on asymptotic expansions for the Jacobi nodes and weights with nearly double precision relative accuracy for degrees $n \geq 100$ and $-1 < \alpha, \beta \leq 5$.

We consider asymptotic approximations of two types, one in terms of Bessel functions and a simpler one in terms of elementary functions. In [4], the computation was based on Bessel-type expansions, but we show that our simpler elementary approach is in fact able to compute accurately most of the nodes and weights (although the Bessel expansions should be considered for the extreme nodes). We discuss how to invert these asymptotic series in order to obtain asymptotic expansions for the nodes, and how to obtain the weights in a numerically stable way using asymptotic expansions related to the derivative of Jacobi polynomials.

We start the paper by summarizing some basic properties satisfied by Jacobi polynomials which are relevant to the paper. In §2 we derive two large-n asymptotic expansions in terms of elementary functions valid for $x \in [-1 + \delta, 1 - \delta]$, where δ is a small positive number. Always, α and β should be of order $o(1)$ for large values of n. The expansion in (2.8)–(2.9) is used in §2.2.2 for finding asymptotic expansions of the zeros. The expansion in terms of the J-Bessel function is considered in §3; see (3.4). This result is valid for $x \in [-1 + \delta, 1]$, where again δ is a small positive number, and it is used in §3.2 to find asymptotic expansions of the zeros. In §4 we give details of the numerical performance of these expansions. Finally, in §5 we summarize other applications of the methods described in this paper, in particular Gauss-Lobatto quadrature and barycentric interpolation.

1.1. Jacobi polynomials: first properties. The Jacobi polynomials $P_n^{(\alpha,\beta)}(x)$, with $\alpha, \beta > -1$, constitute an orthogonal set with weight function $w(x) = (1-x)^\alpha(1+x)^\beta$ on the interval $[-1, 1]$, and they satisfy the symmetry relation

$$P_n^{(\alpha,\beta)}(-x) = (-1)^n P_n^{(\beta,\alpha)}(x). \tag{1.1}$$

An explicit representation can be given in the form of a Gauss hypergeometric function

$$
\begin{aligned}
P_n^{(\alpha,\beta)}(x) &= \sum_{\ell=0}^n \frac{(n+\alpha+\beta+1)_\ell (\alpha+\ell+1)_{n-\ell}}{\ell!\,(n-\ell)!} \left(\frac{x-1}{2}\right)^\ell \\
&= \frac{(\alpha+1)_n}{n!}\, {}_2F_1\left(\begin{matrix} -n, n+\alpha+\beta+1 \\ \alpha+1 \end{matrix}; \tfrac{1}{2}(1-x)\right),
\end{aligned}
\tag{1.2}
$$

and the many transformation formulas of the Gauss function give many other representations.

The Rodrigues formula is

$$P_n^{(\alpha,\beta)}(x) = \frac{(-1)^n}{2^n n!\, w(x)} \frac{d^n}{dx^n}\left(w(x)(1-x^2)^n\right), \tag{1.3}$$

and this gives the integral representation

$$P_n^{(\alpha,\beta)}(x) = \frac{(-1)^n}{2^n\, w(x)} \frac{1}{2\pi i} \int_{\mathcal{C}} \frac{w(z)(1-z^2)^n}{(z-x)^{n+1}}\, dz, \quad x \in (-1,1), \tag{1.4}$$

where the contour $\mathcal{C}$ is a circle around the point $z = x$ with a radius small enough to have the points ± 1 outside the circle.

The recurrence relation is

$$P_{n+1}^{(\alpha,\beta)}(x) = (A_n x + B_n)P_n^{(\alpha,\beta)}(x) - C_n P_{n-1}^{(\alpha,\beta)}(x), \tag{1.5}$$

where

$$\begin{aligned}
A_n &= \frac{(2n+\alpha+\beta+1)(2n+\alpha+\beta+2)}{2(n+1)(n+\alpha+\beta+1)}, \\
B_n &= \frac{(\alpha^2-\beta^2)(2n+\alpha+\beta+1)}{2(n+1)(n+\alpha+\beta+1)(2n+\alpha+\beta)}, \\
C_n &= \frac{(n+\alpha)(n+\beta)(2n+\alpha+\beta+2)}{(n+1)(n+\alpha+\beta+1)(2n+\alpha+\beta)},
\end{aligned} \tag{1.6}$$

with initial values $P_0^{(\alpha,\beta)}(x) = 1$, $P_1^{(\alpha,\beta)}(x) = \frac{1}{2}(\alpha-\beta) + \frac{1}{2}(\alpha+\beta+2)x$.

2. Expansions in terms of elementary functions. We give details for an asymptotic expansion of the Jacobi polynomial $P_n^{(\alpha,\beta)}(x)$ that is valid inside the interval $[-1+\delta, 1-\delta]$, with δ a fixed small positive number. This expansion is best suited for the zeros of $P_n^{(\alpha,\beta)}(x)$ around the origin, but in fact it will provide double precision accuracy for most of the zeros, as we will discuss. Because of the relation $P_n^{(\alpha,\beta)}(x) = (-1)^n P_n^{(\beta,\alpha)}(-x)$ we can concentrate on the positive zeros.

2.1. An expansion derived by E. Hahn. In [26, §18.15(i)] an expansion is given derived in [22], which has the nice property that the coefficients are known in explicit form.

The expansion is described in terms of several formulas. We have for large values of n

$$\begin{aligned}
P_n^{(\alpha,\beta)}(\cos\theta) &= \frac{2^{2n+\alpha+\beta+1}\mathrm{B}\left(n+\alpha+1, n+\beta+1\right)}{\pi\,\sin^{\alpha+\frac{1}{2}}\frac{1}{2}\theta\,\cos^{\beta+\frac{1}{2}}\frac{1}{2}\theta} \times \\
&\quad \left(\sum_{m=0}^{M-1} \frac{f_m(\theta)}{2^m(2n+\alpha+\beta+2)_m} + \mathcal{O}\left(n^{-M}\right)\right),
\end{aligned} \tag{2.1}$$

where

$$\begin{aligned}
f_m(\theta) &= \sum_{\ell=0}^{m} \frac{C_{m,\ell}(\alpha,\beta)}{\ell!(m-\ell)!} \frac{\cos\theta_{n,m,\ell}}{\left(\sin\frac{1}{2}\theta\right)^\ell \left(\cos\frac{1}{2}\theta\right)^{m-\ell}}, \\
C_{m,\ell}(\alpha,\beta) &= \left(\tfrac{1}{2}+\alpha\right)_\ell \left(\tfrac{1}{2}-\alpha\right)_\ell \left(\tfrac{1}{2}+\beta\right)_{m-\ell} \left(\tfrac{1}{2}-\beta\right)_{m-\ell},
\end{aligned} \tag{2.2}$$

$$\theta_{n,m,\ell} = \tfrac{1}{2}(2n+\alpha+\beta+m+1)\theta - \tfrac{1}{2}(\alpha+\ell+\tfrac{1}{2})\pi.$$

This expansion has been used in a paper by Gatteschi and Pittaluga [14] to derive an approximation of the mid range zeros, and this approximation in terms of elementary functions is used in [23] as first estimates for these zeros. Additionally, in [23] the expansion (2.1) (and that for its derivative) is used for the iterative refinement of the first estimations of these zeros.

For deriving more accurate approximations of the mid-zeros we prefer a different expansion, which will be given in the next section. Differently to [23], iterative methods (Newton) will not be needed in order to improve the accuracy; after a node has been computed by means of the asymptotic expansion, we will only need to evaluate one additional asymptotic expansion in order to compute the corresponding weight. We therefore estimate that our computations will be around factor $2N$ faster that the iterative algorithm [23], with N the average number of iterations used in the algorithm of [23] ($2N$ and not N because the expansions both for the polynomial and the derivative are used in the iterative algorithm).

2.2. A compound Poincaré-type expansion. In this section we give an expansion which has the canonical form

$$P_n^{(\alpha,\beta)}(\cos\theta) = \frac{1}{\sqrt{\pi\kappa}} \frac{\cos\chi\, P(x) - \sin\chi\, Q(x)}{\sin^{\alpha+\frac{1}{2}}\frac{1}{2}\theta\,\cos^{\beta+\frac{1}{2}}\frac{1}{2}\theta}, \tag{2.3}$$

where

$$x = \cos\theta, \quad \chi = \kappa\theta - \left(\tfrac{1}{2}\alpha + \tfrac{1}{4}\right)\pi, \quad \kappa = n + \tfrac{1}{2}(\alpha+\beta+1), \tag{2.4}$$

with expansions

$$P(x) \sim \sum_{m=0}^{\infty} \frac{p_m(x)}{\kappa^m}, \quad Q(x) \sim \sum_{m=0}^{\infty} \frac{q_m(x)}{\kappa^m}, \quad \kappa \to \infty, \tag{2.5}$$

for $x \in [-1+\delta, 1-\delta]$, and α and β bounded. A few steps to obtain these expansions will be given in §2.2.4.

The first coefficients are

$$p_0(x) = 1, \quad q_0(x) = 0,$$

$$p_1(x) = -\tfrac{1}{2}\alpha\beta, \quad q_1(x) = \frac{2\alpha^2 - 2\beta^2 + (2\alpha^2 + 2\beta^2 - 1)x}{8\sin\theta},$$

$$p_2(x) = -\big(4\alpha^4 + 4\beta^4 - 16\beta^2 - 16\alpha^2 - 24\alpha^2\beta^2 + 8 +$$

$$4(2\alpha^2 + 2\beta^2 - 5)(\alpha^2 - \beta^2)x +$$

$$(4\alpha^4 + 4\beta^4 + 24\alpha^2\beta^2 - 4\alpha^2 - 4\beta^2 + 1)x^2\big)/(128\sin^2\theta),$$

$$q_2(x) = -\tfrac{1}{2}\alpha\beta\, q_1(x). \tag{2.6}$$

The expansions in (2.5) have negative powers of κ. We can modify the expansions by introducing the front factor[1]

$$G_\kappa(\alpha,\beta) = \frac{\Gamma(n+\alpha+1)}{n!\,\kappa^\alpha} = \frac{\Gamma\left(\kappa + \tfrac{1}{2}(\alpha-\beta+1)\right)}{\Gamma\left(\kappa - \tfrac{1}{2}(\alpha+\beta-1)\right)\kappa^\alpha}, \tag{2.7}$$

---

[1]This function is also present in the expansions in terms of Bessel functions, see §3.

and by multiplying the expansions in (2.5) by the asymptotic expansion of $1/G_\kappa(\alpha, \beta)$. The result is the representation

$$P_n^{(\alpha,\beta)}(\cos\theta) = \frac{G_\kappa(\alpha, \beta)}{\sqrt{\pi\kappa}} \frac{\cos\chi\, U(x) - \sin\chi\, V(x)}{\sin^{\alpha+\frac{1}{2}}\frac{1}{2}\theta\, \cos^{\beta+\frac{1}{2}}\frac{1}{2}\theta}, \tag{2.8}$$

with expansions

$$U(x) \sim \sum_{m=0}^{\infty} \frac{u_{2m}(x)}{\kappa^{2m}}, \quad V(x) \sim \sum_{m=0}^{\infty} \frac{v_{2m+1}(x)}{\kappa^{2m+1}}. \tag{2.9}$$

The first coefficients are

$$u_0(x) = 1, \quad v_1(x) = q_1(x),$$

$$\begin{aligned} u_2(x) = \frac{1}{384\sin^2\theta} \big(&12(5 - 2\alpha^2 - 2\beta^2)(\alpha^2 - \beta^2)x + \\ &4(-3(\alpha^2 - \beta^2)^2 + 3(\alpha^2 + \beta^2) - 6 + 4\alpha(\alpha^2 - 1 + 3\beta^2) + \\ &(-12(\alpha^2 + \beta^2)(\alpha^2 + \beta^2 - 1) - 16\alpha(\alpha^2 - 1 + 3\beta^2) - 3)x^2\big). \end{aligned} \tag{2.10}$$

We give the first terms of the expansion of $G_\kappa(\alpha, \beta)$:

$$\begin{aligned} G_\kappa(\alpha, \beta) \sim 1 &- \frac{\alpha\beta}{2\kappa} - \frac{\alpha(\alpha - 1)(1 + \alpha - 3\beta^2)}{24\kappa^2} + \\ &\frac{\alpha(\alpha - 1)(\alpha - 2)\beta(1 + \alpha - \beta^2)}{48\kappa^3} + \dots. \end{aligned} \tag{2.11}$$

A more efficient expansion in negative powers of $\left(\kappa - \frac{1}{2}\beta\right)^2$ reads

$$G_\kappa(\alpha, \beta) \sim (w/\kappa)^\alpha \sum_{m=0}^{\infty} \frac{C_m(\rho)(-\alpha)_{2m}}{w^{2m}}, \tag{2.12}$$

where

$$w = \kappa - \tfrac{1}{2}\beta, \quad \rho = \tfrac{1}{2}(\alpha + 1), \tag{2.13}$$

and the first $C_m(\rho)$ are

$$C_0(\rho) = 1, \quad C_1(\rho) = -\tfrac{1}{12}\rho, \quad C_2(\rho) = \tfrac{1}{1440}\left(5\rho + 1\right). \tag{2.14}$$

For more details we refer to [2, §5.11(iii)] and [37, §6.5.2-6.5.3].

REMARK 1. *When $\alpha = \beta = -\frac{1}{2}$ the Jacobi polynomial becomes a Chebyshev polynomial. We have*

$$P_n^{(-\frac{1}{2},-\frac{1}{2})}(\cos\theta) = \frac{2^{-2n}(2n)!}{(n!)^2}\cos(n\theta) = \frac{\Gamma\left(n + \frac{1}{2}\right)}{\sqrt{\pi}\,\Gamma(n + 1)}\cos(n\theta). \tag{2.15}$$

In this special case we have $\kappa = n$, $\chi = n\theta$. The $q_m(x)$ and $v_m(x)$ all vanish, $p_m(x)$ and $u_m(x)$ all become independent of x, and we obtain $P(x) = \sqrt{n}\,\Gamma\left(n + \frac{1}{2}\right)/\Gamma(n + 1)$.

2.2.1. Expansions of derivatives. A representation of the derivative can be written in the form

$$\frac{d}{d\theta} P_n^{(\alpha,\beta)}(\cos\theta) = \sqrt{\frac{\kappa}{\pi}} G_\kappa(\alpha,\beta) \frac{\sin\chi\, Y(x) + \cos\chi\, Z(x)}{\sin^{\alpha+\frac{1}{2}}\frac{1}{2}\theta\,\cos^{\beta+\frac{1}{2}}\frac{1}{2}\theta}, \qquad (2.16)$$

with expansions

$$Y(x) \sim \sum_{m=0}^{\infty} \frac{y_{2m}(x)}{\kappa^{2m}}, \quad Z(x) \sim \sum_{m=0}^{\infty} \frac{z_{2m+1}(x)}{\kappa^{2m+1}}, \qquad (2.17)$$

where $y_0(x) = 1$ and for $m = 0, 1, 2, \ldots$

$$\begin{aligned}
y_{2m}(x) &= u_{2m}(x) + \frac{d}{d\theta} v_{2m-1}(x) + \frac{\beta - \alpha - (\alpha+\beta+1)x}{2\sin\theta} v_{2m-1}(x), \\
z_{2m+1}(x) &= \frac{d}{d\theta} u_{2m}(x) - v_{2m+1}(x) + \frac{\beta - \alpha - (\alpha+\beta+1)x}{2\sin\theta} u_{2m}(x).
\end{aligned} \qquad (2.18)$$

For computing the Gauss weights it is convenient to have an expansion of the derivative of the function

$$W(\theta) = \cos\chi\, U(x) - \sin\chi\, V(x), \qquad (2.19)$$

which is the oscillatory part in the representation given in (2.8). We have

$$\frac{d}{d\theta} W(\theta) = -\kappa\big(\sin\chi M(x) + \cos\chi N(x)\big), \qquad (2.20)$$

with expansions

$$M(x) \sim \sum_{\ell=0}^{\infty} \frac{m_{2\ell}(x)}{\kappa^{2\ell}}, \quad N(x) \sim \sum_{\ell=0}^{\infty} \frac{n_{2\ell+1}(x)}{\kappa^{2\ell+1}}, \qquad (2.21)$$

where $m_0(x) = u_0(x) = 1$, $n_1(x) = v_1(x)$, and for $\ell = 1, 2, 3, \ldots$

$$\begin{aligned}
m_{2\ell}(x) &= u_{2\ell}(x) - \sin\theta \frac{d}{dx} v_{2\ell-1}(x), \\
n_{2\ell+1}(x) &= v_{2\ell+1}(x) + \sin\theta \frac{d}{dx} u_{2\ell}(x).
\end{aligned} \qquad (2.22)$$

2.2.2. An expansion of the zeros. We denote the zeros of $P_n^{(\alpha,\beta)}(\cos\theta)$ by $x_1, x_2, \cdots, x_n$.

As a first-order approximation in terms of the θ variable we take

$$\theta_0 = \frac{\left(n + 1 - k + \frac{1}{2}\alpha - \frac{1}{4}\right)\pi}{\kappa}, \quad k = 1, 2, \ldots, n. \qquad (2.23)$$

For this value of θ, χ defined in (2.4) becomes $\chi_0 = (n - k + \frac{1}{2})\pi$, which gives $\cos\chi_0 = 0$. This initial value θ_0 is given in [14], together with the expansion $\theta = \theta_0 + \theta_1/\kappa^2 + \mathcal{O}\left(\kappa^{-4}\right)$, with

$$\theta_1 = \left(\tfrac{1}{16} - \tfrac{1}{4}\alpha^2\right)\cot\left(\tfrac{1}{2}\theta_0\right) - \left(\tfrac{1}{16} - \tfrac{1}{4}\beta^2\right)\tan\left(\tfrac{1}{2}\theta_0\right). \qquad (2.24)$$

This is $-v_1(x) = -q_1(x)$, see (2.6).

To find a few more details, we use $W(\theta)$ defined in (2.19), and expand this function at θ_0 by writing $\theta = \theta_0 + \varepsilon$, which gives for a zero θ

$$W(\theta) = W(\theta_0 + \varepsilon) = W(\theta_0) + \frac{\varepsilon}{1!} W'(\theta_0) + \frac{\varepsilon^2}{2!} W''(\theta_0) + \ldots = 0, \qquad (2.25)$$

where the derivatives are with respect to θ. We assume for ε an expansion in the form

$$\varepsilon \sim \frac{\theta_1}{\kappa^2} + \frac{\theta_2}{\kappa^4} + \frac{\theta_3}{\kappa^6} + \ldots . \qquad (2.26)$$

Using this expansion and those of $U(x)$ and $V(x)$ in (2.9), and comparing equal powers of κ, we find

$$
\begin{aligned}
\theta_1 &= -v_1, \\
\theta_2 &= u_2 v_1 + v_1' v_1 + \tfrac{1}{3} v_1^3 - v_3, \\
\theta_3 &= -\tfrac{4}{3} v_1' v_1^3 - \tfrac{1}{5} v_1^5 - v_5 + v_3 v_1^2 - \tfrac{1}{2} v_1'' v_1^2 - 2 v_1' u_2 v_1 - (v_1')^2 v_1 + \\
&\quad v_1' v_3 - u_2^2 v_1 - u_2' v_1^2 + u_4 v_1 - u_2 v_1^3 + u_2 v_3 + v_3' v_1,
\end{aligned}
\qquad (2.27)
$$

where u_m, v_m have argument x, and $x = \cos(\theta_0)$. The derivatives are with respect to θ. In terms of the original variables:

$$
\begin{aligned}
\theta_1 &= -\frac{1}{8 \sin \theta} \left(2\beta^2 x + 2\alpha^2 x - x - 2\beta^2 + 2\alpha^2 \right), \\
\theta_2 &= \frac{1}{384 \sin^3 \theta} \big(-33x - 36\beta^2 x^2 + 36\alpha^2 x^2 + 24\beta^4 x^2 - 24\alpha^4 x^2 + 2x^3 + \\
&\quad 84\beta^2 x - 60\alpha^4 x - 60\beta^4 x + 84\alpha^2 x + 4\beta^4 x^3 + 4\alpha^4 x^3 - 8\beta^2 x^3 + \\
&\quad 40\alpha^2 - 8\alpha^2 x^3 - 40\beta^2 + 32\beta^4 - 32\alpha^4 + 24\alpha^2 \beta^2 x^3 - 24\alpha^2 \beta^2 x \big).
\end{aligned}
\qquad (2.28)
$$

We summarize the steps for finding a zero x_k, $1 \le k \le n$, given n, α, and β.
1. Compute θ_0 defined in (2.23).
2. With this θ_0, compute the coefficients given in (2.27) or (2.28), with $x = \cos \theta_0$. The coefficients $u_{2m}(x)$ and $v_{2m+1}(x)$ are those in the expansions in (2.9).
3. Compute ε from (2.26) and next $\theta = \theta_0 + \varepsilon$.
4. Compute $x_k \sim \cos \theta$.

2.2.3. Details on computing $\cos \chi$. When we evaluate the functions $W(\theta)$ or $W'(\theta)$ (see (2.19) and (2.20)), with θ in the form $\theta = \theta_0 + \varepsilon$, see (2.23), we can write χ defined in (2.4) as

$$\chi = \kappa \varepsilon + \left(n - k + \tfrac{1}{2} \right) \pi, \qquad (2.29)$$

and, hence,

$$\cos \chi = (-1)^{n-k+1} \sin(\kappa \varepsilon), \quad \sin \chi = (-1)^{n-k} \cos(\kappa \varepsilon). \qquad (2.30)$$

In the original forms, especially for the zeros in the middle of the interval, the argument χ may be of order κ, in the new forms the arguments are of order $1/\kappa$. As a

consequence, the evaluation of these functions can be done with better accuracy when we use (2.30).

For example, when we take $n = 100$, $\alpha = \frac{1}{3}$, $\beta = \frac{1}{5}$ and the expansion in (2.26) with the given three terms in (2.27), we obtain for the middle zero x_{50}, using Maple with Digits=16,

$$\begin{aligned}
\cos\chi &= 0.0001908363241002135, \\
(-1)^{n-k+1}\sin(\kappa\varepsilon) &= 0.0001908363242842724,
\end{aligned} \tag{2.31}$$

with a relative error 9.64×10^{-10}. Corresponding values of $W(\theta)$ are, when using both forms,

$$-0.1837567 \times 10^{-12} \quad \text{and} \quad 0.3013 \times 10^{-15}. \tag{2.32}$$

This feature is more relevant for the middle zeros than for the other ones. However, the middle zeros are more interesting than those near the endpoints ± 1, because the expansions are not valid there.

A similar problem may occur when evaluating $x = \cos\theta$ when we have found the approximation of a zero θ near $\frac{1}{2}\pi$, that is, near $x = 0$. Assume that n is even, and write $n = 2m$. Then $m = \frac{1}{2}\kappa - \frac{1}{4}(\alpha + \beta + 1)$ and for a zero x_k near the origin, we write $k = m + j$. Then, $\theta_0 = \frac{1}{2}\pi + \tau$, where $\tau = (\alpha - \beta + 2 - 4j)/(4\kappa)$, and $x = \cos(\theta_0 + \varepsilon) = -\sin(\varepsilon + \tau)$. Similar for odd $n = 2m + 1$ and the zero x_k with $k = m + j$. Then we have $\tau = (\alpha - \beta + 4 - 4j)/(4\kappa)$. When $\alpha = \beta$ and $j = 1$ (the zero at the origin in this case), we have $\tau = 0$.

2.2.4. The saddle point method for obtaining the expansion. To obtain the representation in (2.3) and the expansions in (2.5) we use the integral representation given in (1.4) where $w(z) = (1 - z)^\alpha (1 + z)^\beta$. We write the integral in the form

$$P_n^{(\alpha,\beta)}(x) = \frac{(-1)^n}{2^n n!\, w(x)} \frac{1}{2\pi i} \int_{\mathcal{C}} e^{-\kappa\phi(z)} \frac{w(z)(z - x)^{\gamma-1}}{(1 - z^2)^\gamma}\, dz, \tag{2.33}$$

where κ is given in (2.4) and

$$\gamma = \tfrac{1}{2}(\alpha + \beta + 1), \quad \phi(z) = \ln(z - x) - \ln(1 - z^2). \tag{2.34}$$

The saddle points $z_\pm$ follow from the zeros of

$$\phi'(z) = \frac{z^2 - 2xz + 1}{(z - x)(1 - z^2)}, \quad \Longrightarrow \quad z_\pm = e^{\pm i\theta}, \quad x = \cos\theta. \tag{2.35}$$

The saddle point contour runs through z_- from $z = -1$ to $z = 1$ (below the real z-axis), and then through z_+ from $z = 1$ to $z = -1$ (above the real axis).

The contribution P^+ from z_+ follows from the substitution

$$\phi(z) - \phi(z_+) = \tfrac{1}{2}s^2, \tag{2.36}$$

with corresponding points $z = \pm 1$ and $s = \pm\infty$. This gives

$$P^+ = -\frac{(-1)^n}{2^n\, w(x)} \frac{e^{-\kappa\phi(z_+)}}{2\pi i} \int_{-\infty}^{\infty} e^{-\frac{1}{2}\kappa s^2} f^+(s)\, ds, \tag{2.37}$$

where

$$f^+(s) = \frac{w(z)}{(z-x)} \left(\frac{z-x}{1-z^2} \right)^\gamma \frac{dz}{ds}. \tag{2.38}$$

By expanding $f^+(s) = f^+(0) \sum_{k=0}^\infty c_k^+ s^k$, the following expansion is obtained:

$$P^+ \sim A^+ \sum_{m=0}^\infty \frac{2^m \left(\frac{1}{2}\right)_m c_{2m}^+}{\kappa^m}, \quad A^+ = -\frac{(-1)^n}{2^n} \frac{e^{-\kappa\phi(z_+)}}{w(x)} \frac{1}{2\pi i} f^+(0), \tag{2.39}$$

For A^+ we evaluate

$$\frac{w(z_+)}{w(x)} = \frac{e^{\frac{1}{2}i(\alpha\theta + \beta\theta - \pi\alpha)}}{\sin^\alpha \frac{1}{2}\theta \cos^\beta \frac{1}{2}\theta}, \qquad \frac{1-z_+^2}{z_+ - x} = -2z_+,$$
$$\left. \frac{dz}{ds} \right|_{s=0} = \sqrt{\sin\theta}\, e^{\frac{1}{2}i\theta - \frac{1}{4}\pi i}, \tag{2.40}$$

and this gives

$$A^+ = \frac{e^{i\chi}}{2\sqrt{\pi\kappa}\, \sin^{\alpha+\frac{1}{2}} \frac{1}{2}\theta \cos^{\beta+\frac{1}{2}} \frac{1}{2}\theta}, \tag{2.41}$$

with χ defined in (2.4).

The contribution from the saddle point z_- is the complex conjugate of P^+ (assuming that α and β are real), and by splitting up the coefficients of the expansion in (2.39) in real and imaginary parts, we obtain (2.3) and (2.5).

REMARK 2. *The starting integrand in (1.4) has a pole at $z = x$, while the one of (2.33) shows an algebraic singularity at $z = x$ and $\phi(z)$ defined in (2.34) has a logarithmic singularity at this point. To handle this from the viewpoint of multivalued functions, we can introduce a branch cut for the functions involved from $z = x$ to the left, assuming that the phase of $z - x$ is zero when $z > x$, equals $-\pi$ when z approaches -1 on the lower part of the saddle point contour in (2.33), and $+\pi$ on the upper side. Because the saddle points $e^{\pm i\theta}$ stay off the interval $(-1, 1)$, we do not need to consider function values on the branch cuts for the asymptotic analysis.*

3. An expansion in terms of Bessel functions. The second expansion we consider is expressed in terms of Bessel functions. The expansion is derived in [12] and in §3.4 we give more details on the coefficients of this expansion. This expansion will be used for the extreme zeros and, combined with the elementary expansion for the inner nodes, it will lead to a non-iterative double precision algorithm for Gauss-Jacobi quadrature. The expansion reads

$$P_n^{(\alpha,\beta)}(\cos\theta) \sim \frac{G_\kappa(\alpha,\beta)}{\sin^\alpha \frac{1}{2}\theta \cos^\beta \frac{1}{2}\theta} \sqrt{\frac{\theta}{\sin\theta}} \sum_{m=0}^\infty A_m(\theta) \frac{J_{\alpha+m}(\kappa\theta)}{\kappa^m}, \tag{3.1}$$

where $G_\kappa(\alpha,\beta)$ is defined in (2.7), and

$$\kappa = n + \tfrac{1}{2}(\alpha + \beta + 1). \tag{3.2}$$

This holds uniformly valid with respect to $\theta \in [0, \pi - \delta]$, with δ an arbitrarily small positive number. The coefficients $A_m(\theta)$ are analytic functions for $0 \le \theta < \pi$, and the first ones are $A_0(\theta) = 1$ and

$$A_1(\theta) = \frac{(4\alpha^2 - 1)(\sin\theta - \theta\cos\theta) + 2\theta(\alpha^2 - \beta^2)(\cos\theta - 1)}{8\theta\sin\theta}. \tag{3.3}$$

We convert the expansion into the representation (see also [3] and [26, Eqn. 18.15.6])

$$P_n^{(\alpha,\beta)}(\cos\theta) = \frac{G_\kappa(\alpha,\beta)}{\sin^\alpha \frac{1}{2}\theta \cos^\beta \frac{1}{2}\theta} \sqrt{\frac{\theta}{\sin\theta}} W(\theta),$$

$$W(\theta) = J_\alpha(\kappa\theta) S(\theta) + \frac{1}{\kappa} J_{\alpha+1}(\kappa\theta) T(\theta), \tag{3.4}$$

where $S(\theta)$ and $T(\theta)$ have the expansions

$$S(\theta) \sim \sum_{m=0}^{\infty} \frac{S_m(\theta)}{\kappa^{2m}}, \quad T(\theta) \sim \sum_{m=0}^{\infty} \frac{T_m(\theta)}{\kappa^{2m}}, \quad \kappa \to \infty, \tag{3.5}$$

with $S_0(\theta) = A_0(\theta) = 1$, $T_0(\theta) = A_1(\theta)$, and for $m = 1, 2, 3, \ldots$

$$S_m(\theta) = -\frac{1}{\theta^{m-1}} \sum_{j=0}^{m-1} \binom{m-1}{j} A_{j+m+1}(\theta)(-\theta)^j 2^{m-1-j}(\alpha+2+j)_{m-j-1},$$

$$T_m(\theta) = \frac{1}{\theta^m} \sum_{j=0}^{m} \binom{m}{j} A_{j+m+1}(\theta)(-\theta)^j 2^{m-j}(\alpha+1+j)_{m-j}. \tag{3.6}$$

The first few terms are also given in [3], where the expansion is derived by using the differential equation of the Jacobi polynomials.

To compute the coefficients $A_m(\theta)$ for small values of θ in a stable way we need expansions. We can write

$$A_m(\theta) = \chi^m \theta^m \sum_{j=0}^{\infty} A_{jm}\theta^{2j}, \quad \chi = \frac{\theta}{\sin\theta}, \tag{3.7}$$

where the series represent entire functions of θ. The first few A_{jm} are

$$A_{0,1} = \tfrac{1}{24}\left(\alpha^2 + 3\beta^2 - 1\right),$$

$$A_{1,1} = \tfrac{1}{480}\left(-3\alpha^2 - 5\beta^2 + 2\right),$$

$$A_{0,2} = \tfrac{1}{5760}\left(-16\alpha - 14\alpha^2 - 90\beta^2 + 5\alpha^4 + 4\alpha^3 + 45\beta^4 + 30\beta^2\alpha^2 + 60\beta^2\alpha + 21\right). \tag{3.8}$$

The coefficients $S_m(\theta)$ and $T_m(\theta)$ can be expanded in the form

$$S_m(\theta) = \theta^2 \chi^{2m} \sum_{j=0}^{\infty} S_{jm}\theta^{2j}, \quad m \ge 1,$$

$$T_m(\theta) = \theta \chi^{2m+1} \sum_{j=0}^{\infty} T_{jm}\theta^{2j}, \quad m \ge 0, \tag{3.9}$$

in which the series represent entire functions of θ.

REMARK 3. *Asymptotic expansions of Jacobi polynomials in terms of modified Bessel functions $I_\nu(z)$ have been derived in [11], where $P_n^{(\alpha,\beta)}(z)$ and an associated function (with expansion in terms of $K_\nu(z)$) are considered for $z \notin [-1,1]$. The approach is based on Frank Olver's methods by using the differential equation. However, Elliott's expansion of $P_n^{(\alpha,\beta)}(z)$ is also valid for z in a neighborhood of $z = 1$, and it coincides with our expansion when performing the proper transformations.*

3.1. Expansions of a derivative. For computing the Gauss weights it is convenient to have an expansion of the derivative of the function $U(\theta) = \sqrt{\theta}\, W(\theta)$, with $W(\theta)$ defined in (3.4). We have

$$\frac{d}{d\theta}U(\theta) = -\kappa\sqrt{\theta}\left(J_{\alpha+1}(\kappa\theta)\,Y(\theta) - \frac{1}{2\theta\kappa}J_\alpha(\kappa\theta)\,Z(\theta)\right), \qquad (3.10)$$

with expansions

$$Y(\theta) \sim \sum_{m=0}^{\infty}\frac{Y_m(\theta)}{\kappa^{2m}}, \quad Z(\theta) \sim \sum_{m=0}^{\infty}\frac{Z_m(\theta)}{\kappa^{2m}}, \quad \kappa \to \infty, \qquad (3.11)$$

where $Y_0(\theta) = S_0(\theta) = A_0(\theta) = 1$, $Z_0(\theta) = (2\alpha+1) + 2\theta A_1(\theta)$, and for $m = 1, 2, 3, \ldots$

$$
\begin{aligned}
Y_m(\theta) &= S_m(\theta) + \frac{2\alpha+1}{2\theta}T_{m-1}(\theta) - \frac{d}{d\theta}T_{m-1}(\theta), \\
Z_m(\theta) &= (2\alpha+1)S_m(\theta) + 2\theta T_m(\theta) + 2\theta\frac{d}{d\theta}S_m(\theta).
\end{aligned}
\qquad (3.12)
$$

For small values of θ we need expansions, similar as in (3.7). We have

$$
\begin{aligned}
Y_m(\theta) &= \chi^{2m}\sum_{j=0}^{\infty}Y_{jm}\theta^{2j}, \quad m \geq 1, \\
Z_m(\theta) &= \theta^2\chi^{2m+1}\sum_{j=0}^{\infty}Z_{jm}\theta^{2j}, \quad m \geq 0,
\end{aligned}
\qquad (3.13)
$$

in which the series represent entire functions of θ.

3.2. Expansions of the zeros. For obtaining accurate approximations of the zeros for large degree Jacobi polynomials we can use the Bessel-type expansion given earlier. We give asymptotic expansions that can be used for all positive zeros. However, the most interesting region of application of Bessel-type expansions is close to the endpoints of the interval $[-1,1]$, because for the rest of the interval the previous elementary expansions are accurate enough and they are simpler to handle (and therefore more efficient). We explain how to compute explicitly enough coefficients of the expansions so that the zeros are computed accurately, without the need to use iterative methods to refine the results.

We write the zeros x_{n+1-m} (with $x_1 < x_2 < \cdots < x_n$) of $P_n^{(\alpha,\beta)}(x)$ in terms of the zeros j_m of the Bessel function $J_\alpha(x)$. The zero x_n corresponds to j_1, x_{n-1} to j_2, and so on. Because the representation in (3.4) cannot be used as $x \to -1$, we consider only the positive zeros. For the other zeros we can use the symmetry relation (1.1).

A zero of $P_n^{(\alpha,\beta)}(x)$ is a zero of $W(\theta)$ defined in (3.4), and we approximate a zero x_{n+1-m} with corresponding θ value following from $x_{n+1-m} = \cos\theta$. We write θ in

the form

$$\theta = \theta_0 + \varepsilon, \quad \theta_0 = j_m/\kappa. \tag{3.14}$$

We assume for ε an expansion in the form

$$\varepsilon \sim \frac{\theta_1}{\kappa^2} + \frac{\theta_2}{\kappa^4} + \frac{\theta_3}{\kappa^6} + \dots \tag{3.15}$$

By expanding $W(\theta)$ at θ_0 we have upon expanding

$$W(\theta) = W(\theta_0 + \varepsilon) = W(\theta_0) + \frac{\varepsilon}{1!}W'(\theta_0) + \frac{\varepsilon^2}{2!}W''(\theta_0) + \dots = 0. \tag{3.16}$$

Using the representation of $W(\theta)$ given in (3.4), substituting the expansion of ε and those of $S(\theta)$ and $T(\theta)$ given in (3.5), and comparing equal powers of κ, we can obtain the coefficients θ_j of (3.15).

The first coefficient is $\theta_1 = T_0(\theta_0) = A_1(\theta_0)$ (see (3.9)), and the second one follows from

$$6\theta\theta_2 = 6\theta T_1(\theta) + 6\theta(T_0'(\theta) - S_1(\theta))T_0(\theta) - 3(1 + 2\alpha)T_0(\theta)^2 - 2\theta T_0(\theta)^3, \tag{3.17}$$

evaluated at $\theta = \theta_0$.

The coefficients can be expanded in the form

$$\theta_m = \theta\chi^{2m-1}\sum_{j=0}^{\infty} t_{jm}\theta^{2j}, \quad m \geq 1, \tag{3.18}$$

evaluated at θ_0, where the series represent entire functions of θ. The first few t_{jm} are

$$t_{0,2} = \tfrac{1}{2}(2T_{0,1} + T_{0,0}^2 - 2\alpha T_{0,0}^2),$$

$$t_{1,2} = \tfrac{1}{12}(-4T_{0,0}^3 + 12T_{1,1} - 12T_{0,0}S_{0,1} - 24\alpha T_{0,0}T_{1,0} +$$

$$36T_{0,0}T_{1,0} + 3T_{0,0}^2 + 2\alpha T_{0,0}^2), \tag{3.19}$$

$$t_{0,3} = \tfrac{1}{12}(12T_{0,2} + 12T_{0,0}T_{0,1} + 2T_{0,0}^3 +$$

$$16\alpha^2 T_{0,0}^3 - 12\alpha T_{0,0}^3 - 24\alpha T_{0,0}T_{0,1}).$$

The numerical steps to obtain the zeros are as explained at the end of §2.2.2. For example, with $n = 100$, $\alpha = \tfrac{1}{3}$, $\beta = \tfrac{1}{4}$, we have for the largest zero $x_{100} = 0.9995853721163790$ (computed by Maple), $\theta_0 = 0.02879787927325625$, and an approximation of x_{100} given by $\cos\theta_0 = 0.9995853697308934$, with relative error 2.39×10^{-9}. Compute $\theta_1 = -0.8416536425087086 \times 10^{-3}$, and we have $\theta \sim \theta_0 + \theta_1/\kappa^2$, which gives $x_{100} \sim \cos\theta = 0.9995853721164185$, with relative error 3.96×10^{-14}. These computations were done in Maple with Digits=16.

3.3. Details on computing Bessel functions near a zero. The numerical evaluation of the Bessel function $J_\alpha(\kappa\theta)$ that occurs in $W(\theta)$ defined in (3.4) and in the derivative of $U(\theta)$ in (3.10), may become unstable when $\kappa\theta$ is near a zero of $J_\alpha(z)$. This problem shows up when computing the Gauss weights and when we use the standard software for the Bessel functions. A related problem is discussed in §2.2.3 for the elementary case.

To handle this, we can use the following expansion (see [29, Eqn. 10.23.1])

$$J_\alpha(\lambda z) = \lambda^\alpha \sum_{m=0}^\infty \frac{w^m}{m!} J_{\alpha+m}(z), \quad w = \tfrac{1}{2} z \left(1 - \lambda^2\right), \tag{3.20}$$

with $z = u$ and $\lambda = 1 + h/u$. This gives

$$J_\alpha(u + h) = \lambda^\alpha \sum_{m=0}^\infty \frac{w^m}{m!} J_{\alpha+m}(u), \quad w = -\frac{h(2u + h)}{2u}. \tag{3.21}$$

When h is small the series works as a Taylor expansion, because $w = \mathcal{O}(h)$. When u is a zero of $J_\alpha(u)$, the first term of the series vanishes.

This happens when we take $u + h = \kappa\theta$ with $u = \kappa\theta_0 = j_k$ (a zero of $J_\alpha(u)$, see (3.14)) and $h = \kappa\varepsilon$. The mentioned computational problems arise for largest zeros x_k (with small k). In that case, $u = \mathcal{O}(1)$ and $h = \mathcal{O}(1/\kappa)$, and the series in (3.21) converges quite fast.

The terms of the series can be generated by using the recurrence relation of the Bessel function

$$J_{\alpha+1}(u) = \frac{2\alpha}{u} J_\alpha(u) - J_{\alpha-1}(u), \tag{3.22}$$

which gives for $m = 1, 2, 3, \ldots$

$$m(m + 1) f_{m+1} = \frac{2m(\alpha + m)w}{u} f_m - w^2 f_{m-1}, \quad f_m = \frac{w^m}{m!} J_{\alpha+m}(u), \tag{3.23}$$

with starting value $f_0 = 0$ (when u is a zero of $J_\alpha(u)$) and $f_1 = w J_{\alpha+1}(u)$.

In Table 3.1 we show the relative errors in the computation of the Bessel function $J_\alpha(u + h)$ for $\alpha = \tfrac{1}{4}$, $u = j_5 = 15.32\cdots$ and several small values of h by using the standard Maple code for this Bessel function and the series expansion in (3.21), both with Digits $= 16$. The value m indicates the number of terms used in the expansion. The errors follow from comparing the results with Bessel function values obtained in Maple with Digits $= 64$. We observe a good performance of the series expansions, the standard algorithm gives poorer results as h becomes smaller.

REMARK 4. *We know that the forward recursion of the Bessel functions in (3.23) may be unstable. However, because we start in the domain where the zeros are ($u > \alpha$), the recursion of the first terms will be stable, and for the later terms we can accept less accurate values. Because of the fast convergence of the series for the values of u and h to compute the Gauss weights, it is not needed to use a backward recursion scheme for the Bessel functions.*

3.4. Details on the coefficients $A_m(\theta)$. We verify the expansion in (3.1) and give details on the coefficients $A_m(\theta)$. We introduce

$$b_j(\theta) = \frac{(-1)^j \left(\tfrac{1}{2}\alpha + \tfrac{1}{2}\beta\right)_j \left(\tfrac{1}{2}\alpha - \tfrac{1}{2}\beta\right)_j}{j! \left(\alpha + \tfrac{1}{2}\right)_j (1 + \cos\theta)^j}, \quad j = 0, 1, 2, \ldots, \tag{3.24}$$

and for $m = 0, 1, 2, \ldots$ functions $\psi_{j,m}(\theta)$ defined by the generating functions

$$\left(\frac{2\theta(\cos\phi - \cos\theta)}{\sin\theta\,(\theta^2 - \phi^2)}\right)^{m+\alpha-\frac{1}{2}} = \sum_{j=0}^\infty \psi_{j,m}(\theta)\left(\theta^2 - \phi^2\right)^j. \tag{3.25}$$

TABLE 3.1

Relative errors in the computation of the Bessel function $J_\alpha(u+h)$ for $\alpha = \frac{1}{4}$, $u = j_5 = 15.32\cdots$ and several small values of h by using the standard Maple code for this Bessel function with Digits = 16 and the series expansion in (3.21); m indicates the number of terms used in the expansion.

h	m	standard algorithm	series expansion (3.21)
10^{-1}	9	0.26×10^{-13}	0.26×10^{-15}
10^{-2}	5	0.26×10^{-12}	0.68×10^{-16}
10^{-3}	4	0.26×10^{-11}	0.17×10^{-15}
10^{-4}	3	0.26×10^{-10}	0.55×10^{-15}
10^{-5}	2	0.26×10^{-09}	0.13×10^{-15}

Then, the coefficients $A_m(\theta)$ are given by

$$A_m(\theta) = (2\theta)^m \left(\alpha + \tfrac{1}{2}\right)_m \sum_{j=0}^{m} b_j(\theta) \left(\frac{\sin\theta}{2\theta}\right)^j \psi_{m-j,j}(\theta). \qquad (3.26)$$

The starting point in [12] is the integral representation (due to George Gasper [13]) in terms of the Gauss hypergeometric function

$$P_n^{(\alpha,\beta)}(\cos\theta) = Q_n^{(\alpha,\beta)}(\theta) \times$$
$$\int_0^\theta \frac{\cos(\kappa\phi)}{(\cos\phi - \cos\theta)^{\frac{1}{2}-\alpha}} {}_2F_1\left(\begin{matrix} \tfrac{1}{2}(\alpha+\beta), \tfrac{1}{2}(\alpha-\beta) \\ \alpha + \tfrac{1}{2} \end{matrix} ; \frac{\cos\theta - \cos\phi}{1 + \cos\theta}\right) d\phi, \qquad (3.27)$$

where κ is defined in (3.2) and

$$Q_n^{(\alpha,\beta)}(\theta) = \frac{2^{\frac{1}{2}-\alpha}\Gamma(n+\alpha+1)}{\sqrt{\pi}\, n!\, \Gamma\left(\alpha + \tfrac{1}{2}\right) \sin^{2\alpha}\left(\tfrac{1}{2}\theta\right) \cos^{\alpha+\beta}\left(\tfrac{1}{2}\theta\right)}. \qquad (3.28)$$

By expanding the ${}_2F_1$-function, it follows that

$$P_n^{(\alpha,\beta)}(\cos\theta) = Q_n^{(\alpha,\beta)}(\theta) \times$$
$$\sum_{m=0}^{\infty} b_m \int_0^\theta \cos(\kappa\phi)(\cos\phi - \cos\theta)^{m+\alpha-\frac{1}{2}}\, d\phi, \qquad (3.29)$$

where $b_m(\theta)$ is defined in (3.24)

Next, the expansion in (3.25) is used and the integral representation[2]

$$\int_0^\theta \left(\theta^2 - \phi^2\right)^{\sigma-\frac{1}{2}} \cos(\kappa\phi)\, d\phi = 2^{\sigma-1}\sqrt{\pi}\,\Gamma\left(\sigma + \tfrac{1}{2}\right)\theta^\sigma \kappa^{-\sigma} J_\sigma(\kappa\theta), \qquad (3.30)$$

---

[2] A proof easily follows by expanding the cosine function in its power series.

with $\sigma = j + m + \alpha$. This gives the double sum

$$
\begin{aligned}
P_n^{(\alpha,\beta)}(\cos\theta) &= Q_n^{(\alpha,\beta)}(\theta)\sqrt{\frac{\pi}{2}}\left(\frac{\sin\theta}{\theta}\right)^{\alpha-\frac{1}{2}}\left(\frac{\theta}{\kappa}\right)^{\alpha} \times \\
&\quad \sum_{m=0}^{\infty} p_m \sum_{j=0}^{\infty} q_{j+m} \psi_{j,m}(\theta) \\
&= Q_n^{(\alpha,\beta)}(\theta)\sqrt{\frac{\pi}{2}}\left(\frac{\sin\theta}{\theta}\right)^{\alpha-\frac{1}{2}}\left(\frac{\theta}{\kappa}\right)^{\alpha} \times \\
&\quad \sum_{m=0}^{\infty} q_m \sum_{j=0}^{m} p_j \psi_{m-j,j}(\theta),
\end{aligned}
\tag{3.31}
$$

where $\psi_{j,m}(\theta)$ are the coefficients in (3.25), and

$$
p_m = b_m(\theta)\left(\frac{\sin\theta}{2\theta}\right)^m, \quad q_m = \left(\frac{2\theta}{\kappa}\right)^m \Gamma\left(m+\alpha+\tfrac{1}{2}\right) J_{m+\alpha}(\kappa\theta).
\tag{3.32}
$$

This verifies the expansion in (3.1) with the $A_m(\theta)$ defined in (3.26).

We give a few details from [12] on evaluating the functions $\psi_{j,m}(\theta)$. First consider the expansion (see [41, p. 140 (3)])

$$
\frac{2\theta(\cos\phi - \cos\theta)}{\sin\theta\,(\theta^2 - \phi^2)} = \sum_{j=0}^{\infty} \chi_j(\theta)\left(\theta^2 - \phi^2\right)^j,
\tag{3.33}
$$

where the $\chi_j(\theta)$ are available in the form of Bessel functions of fractional order:

$$
\chi_j(\theta) = \frac{1}{(j+1)!\,(2\theta)^j}\frac{J_{j+\frac{1}{2}}(\theta)}{J_{\frac{1}{2}}(\theta)}, \quad j = 0, 1, 2,, \ldots.
\tag{3.34}
$$

These functions follow from a simple recursion, and then the $\psi_{j,m}(\theta)$ follow from

$$
\left(\sum_{j=0}^{\infty}\chi_j(\theta)w^j\right)^{\mu} = \sum_{j=0}^{\infty}\psi_{j,m}(\theta)w^j, \quad w = \theta^2 - \phi^2, \quad \mu = m + \alpha - \tfrac{1}{2}.
\tag{3.35}
$$

The first two are

$$
\psi_{0,m}(\theta) = 1, \quad \psi_{1,m}(\theta) = \tfrac{1}{4}\mu\frac{1 - \theta\cot\theta}{\sin^2\theta}.
\tag{3.36}
$$

4. Numerical performance of the expansions for computing the nodes and weights of the G-J quadrature. Next, we discuss in more detail the performance of the expansions given in §2.2 and §3 for computing the nodes and weights of the Gauss-Jacobi quadrature.

In terms of the derivatives of the orthogonal polynomials, the weights of the Gauss-Jacobi quadrature are given by

$$
w_i = \frac{M_{n,\alpha,\beta}}{(1 - x_i^2)[P_n^{(\alpha,\beta)\prime}(x_i)]^2} = \frac{M_{n,\alpha,\beta}}{\left[\dfrac{d}{d\theta}P_n^{(\alpha,\beta)}(\cos\theta_i)\right]^2},
\tag{4.1}
$$

$$
M_{n,\alpha,\beta} = 2^{\alpha+\beta+1}\frac{\Gamma(n+\alpha+1)\Gamma(n+\beta+1)}{n!\Gamma(n+\alpha+\beta+1)},
$$

where $x_i = \cos\theta_i$.

As we did in [19] for the Gauss–Hermite and Gauss–Laguerre quadratures, it is convenient to introduce scaled weights:

$$\omega_i = \dot{u}(\theta_i)^{-2} \qquad (4.2)$$

where the dot indicates the derivative with respect to θ and

$$u(\theta) = M_{n,\alpha,\beta}^{-\frac{1}{2}} \left(\sin\frac{1}{2}\theta\right)^{\alpha+\frac{1}{2}} \left(\cos\frac{1}{2}\theta\right)^{\beta+\frac{1}{2}} P_n^{(\alpha,\beta)}(\cos\theta). \qquad (4.3)$$

The weights are related with the scaled weights by

$$w_i = \left(\sin\frac{1}{2}\theta\right)^{2\alpha+1} \left(\cos\frac{1}{2}\theta\right)^{2\beta+1} \omega_i. \qquad (4.4)$$

The advantage of computing scaled weights is that, similarly as described in [19], scaled weights do not underflow/overflow for large parameters and, additionally, they are well-conditioned as a function of the roots θ_i. Indeed, the scaled weights are $\omega_i = W(\theta_i)$ with $W(\theta) = \dot{u}(\theta)^{-2}$, but $\dot{W}(\theta_i) = 0$ because $\ddot{u}(\theta_i) = 0$. In addition, as $n \to +\infty$, the scaled weights are essentially constant and the main dependence on the nodes of the unscaled weights w_i is given by the elementary sine and cosine factor of (4.4). This is related to the circle theorem for Gaussian quadratures in $[-1, 1]$ (see [8, 16]), which states that, as $n \to +\infty$,

$$\frac{nw_i}{\pi w(x_i)} \sim \sqrt{1 - x_i^2},$$

where in the Gauss-Jacobi case $w(x) = (1-x)^\alpha (1+x)^\beta$.

When considering the asymptotic expansion for Jacobi polynomials in terms of elementary functions, the function $u(\theta)$ can be written in terms of the function $W(\theta)$ given in (2.19)

$$u(\theta) = \frac{M_{n,\alpha,\beta}^{-\frac{1}{2}} G(\alpha,\ \beta)}{\sqrt{\pi\kappa}} W(\theta), \qquad (4.5)$$

and for the computation of $\dot{u}(\theta)$ we use the expansion (2.20).

When considering the asymptotic expansion in terms of Bessel functions, the function $u(\theta)$ is given by

$$u(\theta) = \frac{M_{n,\alpha,\beta}^{-\frac{1}{2}} \Gamma(n+\alpha+1)}{n!\kappa^\alpha \sqrt{2}} U(\theta), \qquad (4.6)$$

where $U(\theta)$ is given in Section 3.1. In this case, the expansion (3.10) is used for computing the derivative of $u(\theta)$.

Examples of the performance of the asymptotic expansions for the evaluation of the nodes and scaled weights of the Gauss-Jacobi quadrature are given in Figures 4.1, 4.2, 4.3 and 4.4. We concentrate on the positive zeros; for the negative zeros

we can use the relation (1.1). In our tests, we use finite precision implementations (double precision accuracy) of the expansions [3]. We compare the approximations to the nodes obtained with the asymptotic expansions against the results of a Maple implementation (with a large number of digits) of an iterative algorithm which uses the global fixed point method of [31]. The Jacobi polynomials used in this algorithm are computed using the intrinsic Maple function. The scaled weights for testing have also been computed using Maple.

Figure 4.1 shows the performance of the expansion in terms of elementary functions for $P_n^{(\alpha,\beta)}(x)$ for $\alpha = 0.1$, $\beta = -0.3$ and $n = 100, 1000$. For computing the scaled weights (SW in the figure) for $n = 100$, five terms have been considered in the expansions of the coefficients $M(x)$ and $N(x)$ in (2.21). For the nodes, we have used six terms in the expansion (2.26). For $n = 1000$, fewer terms were needed to obtain the accuracy shown in the figure. The label i in the abscissa represents the order of the zero (starting from $i = 1$ for the smallest positive zero). The points not shown in the plot correspond to values with all digits correct in double precision accuracy. As can be seen, for $n = 100$ the use of the elementary expansion allows the computation of more than $3/4$ of the first zeros with 15-16 digits correct and it fails for the largest zeros, as expected. When $n = 1000$, only the last few nodes are not computed with double precision accuracy. It is important to note that, using the idea given §2.2.3 for the computation of $\cos\theta$, there is no loss of accuracy for the first nodes and then double precision accuracy is also possible in the relative error of these nodes, which is an advantage over the algorithms of [23].

The asymptotic expansion for the scaled weights performs also well for the first half of the nodes ($n = 100$) with the number of terms used in the computation. For $n = 1000$, only the last 15 scaled weights are computed with an accuracy worse than 10^{-15}.

The elementary expansions are fast and for $n = 100$ the average CPU time per node (running gfortran on a Laptop with Intel Core i54310U 2.6 GHz processor under Windows) was $\approx 0.7\mu s$, and the additional computation of weights increased the total CPU time by a factor 3. As the degree becomes larger, the time per node can be smaller because less terms are required, but only with this (pessimistic) estimation and taking into account we observe a clear speed-up with respect to the methods in [23]. This, is as expected, because for computing the weights we only need to evaluate one additional asymptotic, while the iterative methods must refine the nodes and two expansions (one for the polynomial and another one for the derivative) must be computed. We therefore expect that our computations will be around factor $2N$ faster, with N the average number of iterations used in [23].

For the computation of the nodes using the Bessel expansion, an algorithm for computing the zeros of Bessel functions is needed. In the finite precision implementation of the expansion we use the algorithm given in [17] for the first few zeros. For larger zeros, a very efficient method of computation is the use of the MacMahon's expansion (see [29, §10.21(vi)])

$$j_{\nu,m} \sim a - \frac{\mu - 1}{8a} - \frac{4(\mu - 1)(7\mu - 31)}{3(8a)^3} - \frac{32(\mu - 1)(83\mu^2 - 982\mu + 3779)}{15(8a)^5} - \cdots , \quad (4.7)$$

---

[3]A file with explicit expressions of some of the coefficients of the expansions used in our calculations is available at http://personales.unican.es/gila/CoefsExpansions.txt; see also http://personales.unican.es/segurajj/gaussian.html
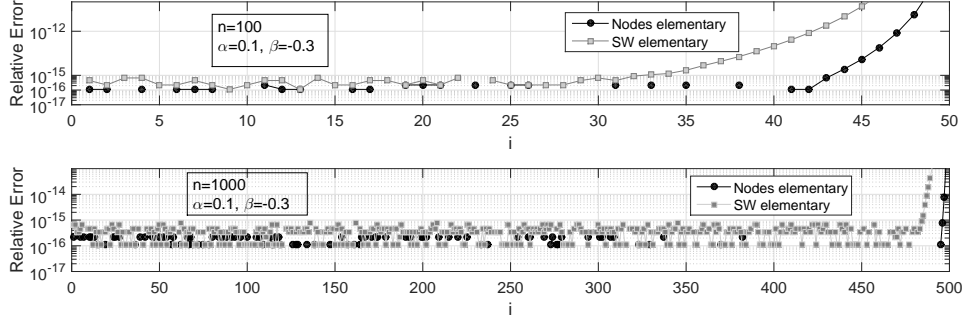
FIG. 4.1.   *Performance of the asymptotic expansions in terms of elementary functions for computing the nodes and scaled weights of $P_n^{(\alpha,\beta)}(x)$ for $\alpha = 0.1$, $\beta = -0.3$ and $n = 100, 1000$.*

where $\mu = 4\nu^2$, $a = (m + \nu/2 - 1/4)\pi$.

On the other hand, the J-Bessel functions needed to compute the weights, are evaluated using our own algorithm for Bessel functions. Two first examples of the performance of the Bessel expansion are shown in Figure 4.2. For comparison, the choice of parameters is the same as in Figure 4.1. For computing the scaled weights for $n = 100$, four terms have been considered in the expansions of the coefficients $Y(x)$ and $Z(x)$ in (3.11). For the nodes, we have used three terms in the expansion (3.15). As can be seen, the expansion performs well even for the smallest nodes (and their corresponding scaled weights), although higher accuracy is obtained for larger nodes as expected. Regarding the computation of scaled weights, it is important to mention that the accurate evaluation of the two Bessel functions appearing in the expansion, seems to be crucial in order to obtain more than 14 digits of accuracy for a few of the points closest to $x = 1$ [4]. For computing the Bessel function $J_\alpha(\kappa\theta)$ the scheme given in §3.3 is used in our calculations for the last points.

We have checked that the range validity of the two different expansions depend very little on the values of α and β ($-1 < \alpha, \beta \leq 5$) and that the main dependence is on the degree n. Because the elementary expansion is more efficient, it should be preferred over the Bessel expansion when it gives sufficiently accurate results. This is, as mentioned, the case of $3/4$ of all the nodes when $n = 100$ and decreasing as n increases. Once the nodes are computed, the weights are obtained, as explained before, by using the asymptotic expansions (2.20) and (3.10); for this computation, it seems convenient to use the elementary expansion (2.20) for less nodes (half of the nodes for $n = 100$), and more weights can be computed with this elementary expansion as n becomes larger.

Two other examples of the computation of nodes and scaled weights with the expansions considered in this paper are shown in Figures 4.3 and 4.4. In Figure 4.3, the parameter α is chosen now larger than before ($\alpha = 5$) in order to show the performance of the expansions for moderate values of the parameters. On the other hand, Figure 4.4 illustrates the performance when both parameters (α and β) are negative. As can be seen, the performance of the expansions is very similar in all cases.

---

[4]In [23] it was reported that a loss of accuracy in the weights near the boundary took place, and that it appeared to be directly related to the error in evaluating the Bessel functions at these points.
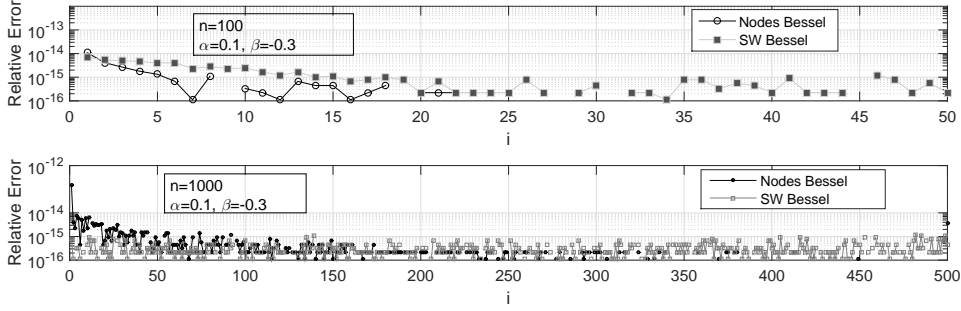
FIG. 4.2. *Performance of the asymptotic expansions in terms of Bessel functions for computing the nodes and scaled weights of $P_n^{(\alpha,\beta)}(x)$ for $\alpha = 0.1$, $\beta = -0.3$ and $n = 100, 1000$.*
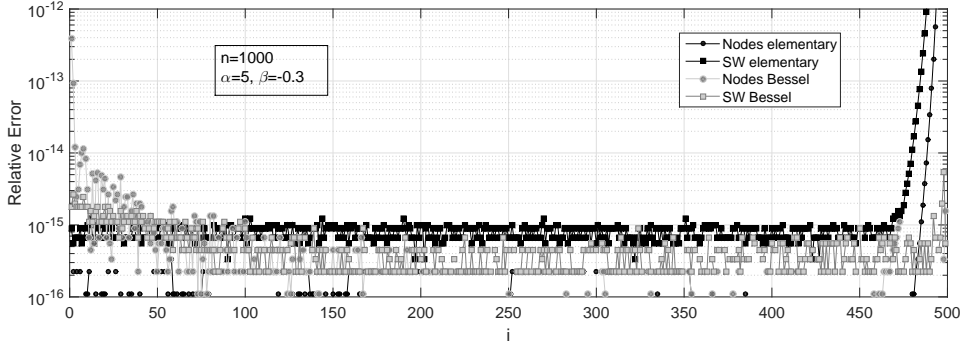

FIG. 4.3.   *Performance of the asymptotic expansions for computing the nodes and scaled weights of $P_n^{(\alpha,\beta)}(x)$ for $\alpha = 5$, $\beta = -0.3$ and $n = 1000$.*

Regarding CPU times, the Bessel expansions, as expected, are slower than the elementary expansions. For $n = 100$ the average CPU time per node (running gfortran on an Intel Core i54310U 2.6 GHz processor under Windows) was $2\,\mu s$. The additional computation of weights increased the total CPU time by a factor 5. This expansion is, as expected, slower than the elementary expansion. However, as mentioned earlier, the Bessel expansions can be avoided for most of the nodes and the execution time of the algorithm combining both expansions is closer to the elementary expansion timing than to the Bessel case. The actual speed-up of the combined algorithm in the least favorable case ($n = 100$) is larger than a factor 20 with respect to the CPU time declared in Table 4.1 for the Matlab algorithm of [23]; however this comparison should be handled with care because it is always difficult to compare algorithms which are executed in different computers, using different programming languages, and probably with different optimization levels. A detailed discussion of the comparison of CPU times will be only possible when the algorithms are made available in a same platform. In any case, taking into account the complexities of the two methods, and as commented earlier, we can safely expect that our methods will be at least $2N$ times faster, with N the average number of iterations used by the algorithm in [23].

Our algorithms are able to produce close to double precision accuracy for $n \geq 100$ and $-1 < \alpha, \beta \leq 5$. The accuracy worsens as large parameters α and β or smaller
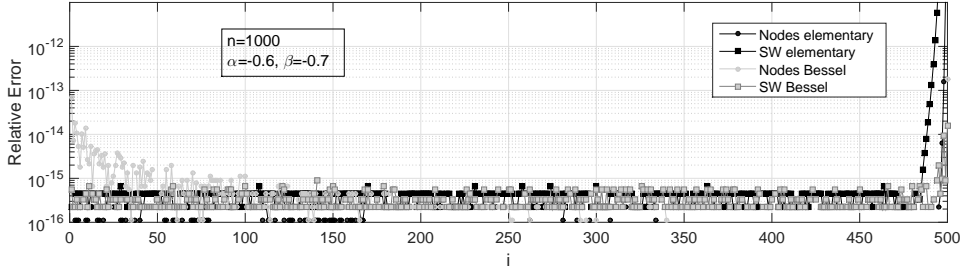
FIG. 4.4.   *Performance of the asymptotic expansions for computing the nodes and scaled weights of $P_n^{(\alpha,\beta)}(x)$ for $\alpha = -0.6$, $\beta = -0.7$ and $n = 1000$.*
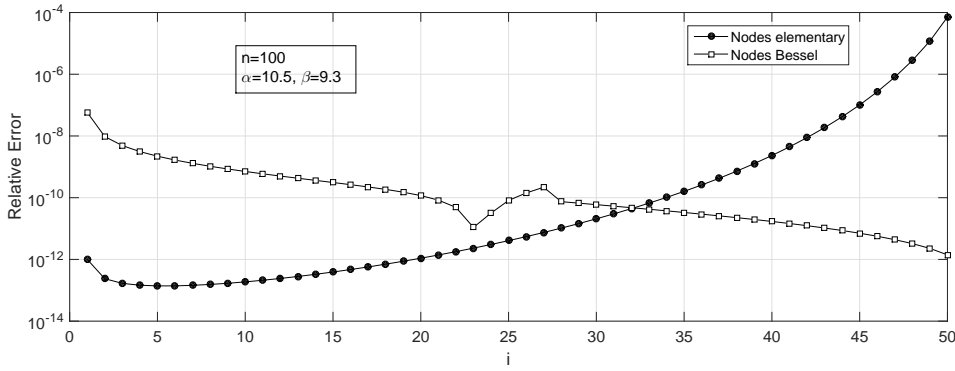

FIG. 4.5.   *Performance of the asymptotic expansions for computing the nodes and scaled weights of $P_n^{(\alpha,\beta)}(x)$ for $\alpha = 10.5$, $\beta = 9.3$ and $n = 1000$.*

n is considered. Larger values of the parameters α and β are not very common in applications (in particular those related to barycentric interpolation and spectral methods which are discussed in the next section) because the Lebesgue constants becomes large, as discussed in [40] (for more details [34, sect. 14.4]). As an illustration of a case with larger parameters, we shown in Fig. 4.5 the accuracy in the computation of the nodes for $n = 100$, $\alpha = 10.5$ and $\beta = 9.3$; we observe that some accuracy is lost with respect to the case $-1 < \alpha, \beta \leq 5$, but still the nodes can be computed with 10^{-10} relative accuracy.

In the same way, our algorithms can also be used for smaller n and they are quite accurate when $n \geq 20$ as long as the parameters are not large. For instance, as shown in Fig. 4.6 it is possible to obtain 10^{-12} relative accuracy for the nodes when $n = 20$. This means that only one Newton step would be enough required to reach double precision accuracy in that cases, thus improving the speed of iterative methods for small degrees.

5. Extensions and further applications. In this paper, we have discussed the non-iterative computation of Gauss-Jacobi quadrature. As we next explain, the methods and techniques considered in this paper can also be applied in other contexts.

5.1. Computation of Jacobi polynomials. First, we mention that the expansions we have described for Jacobi polynomials, together with the described methods
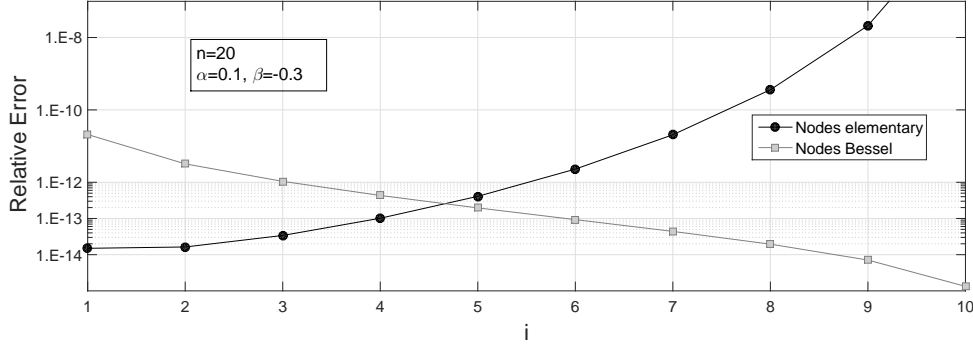
FIG. 4.6. *Performance of the asymptotic expansions for computing the nodes of $P_n^{(\alpha,\beta)}(x)$ for $\alpha = 0.1$, $\beta = 0.3$ and $n = 20$.*

for obtaining the coefficients of the expansions, provide accurate methods for computing the Jacobi polynomials for moderately large orders ($n \geq 100$) and $-1 < \alpha, \beta \leq 5$. We expect that the combination of these methods with the use of the three-term recurrence relation for $n < 100$, will lead to fast and accurate methods for computing Jacobi polynomials, similarly as happened for the computation of Laguerre polynomials in [18].

5.2. Jacobi-Gauss-Radau and Jacobi-Gauss-Lobatto quadratures. In some applications (for example for solving boundary value problems by spectral methods [32]), it can be interesting to include the boundaries of the integration interval as nodes, maybe even with degree of multiplicity greater than one. The generalized Gauss-Lobatto rule in $[a, b]$ for a weight function $w(x)$ supported in $[a, b]$ can be defined as the rule

$$\int_a^b f(x)w(x)dx \approx \sum_{j=0}^{j_a-1} v_0^{[j]} f^{(j)}(a) + \sum_{i=1}^{n} v_i f(x_i) + \sum_{j=0}^{j_b-1} v_{n+1}^{[j]} f^{(j)}(b),$$

with the highest degree of accuracy, which is $2n + j_a + j_b - 1$; here $f^{(j)}$ stands for the j-th derivative of f, and if a sum $\sum_{j=0}^{-1}$ appears we consider it an empty sum. Generalized Jacobi-Gauss-Lobatto quadrature corresponds to $a = -1$, $b = +1$ and $w(x) = (1-x)^\alpha(1+x)^\beta$. The most common cases in applications are when $j_a, j_b = 1$ (Gauss-Lobatto) and $j_a = 1$, $j_b = 0$ or $j_a = 0$, $j_b = 1$ (Gauss-Radau).

For obvious reasons, the weights $v_0^{[j]}$ and $v_{n+1}^{[j]}$ are called boundary weights, while the values x_i and v_i are the internal nodes and weights. This problem is closely related to the problem of Gaussian quadrature studied so far, and in particular the internal nodes and weights can be computed using exactly the same techniques, because they are directly related to Gaussian quadrature. Indeed, it is easy to show that the nodes x_i are the Gaussian nodes in $[a, b]$ associated to the weight function $w(x)(x-a)^{j_a}(b-x)^{j_b}$, while the corresponding Gaussian weights w_i for this modified weight function are related with the internal weights of the generalized Gauss-Lobatto rule by

$$v_i = \frac{w_i}{(x_i - a)^{j_a}(b - x_i)^{j_b}}.$$

The computation of the boundary weights is a different problem, but for the problem we are considering in this paper (the Jacobi case) there exist iterative formulas for

computing them [30], and for the simple cases when $j_a, j_b \leq 1$ (Gauss-Radau or Gauss-Lobatto), the boundary weights have simple and explicit expressions.

For instance, for the Jacobi-Gauss-Lobatto case we have

$$\int_a^b (1-x)^\alpha (1+x)^\beta f(x) dx \approx v_0 f(a) + \sum_{i=1}^n v_i f(x_i) + v_{n+1} f(b),$$

where x_i are the zeros of $P_n^{(\alpha+1,\beta+1)}(x)$ and $v_i = w_i/(1-x_i^2)$, with w_i the weights for the n-degree Gauss-Jacobi quadrature with parameters $\alpha+1$ and $\beta+1$. Therefore, the internal nodes and weights can be computed with the expressions given in this paper. For the boundary weights, the following formula are available (see, for instance, [15]):

$$v_0 = 2^{\alpha+\beta+1} \frac{(\beta+1)\Gamma(\beta+1)^2 \Gamma(n+1)\Gamma(n+\alpha+2)}{\Gamma(n+\beta+2)\Gamma(n+\alpha+\beta+3)},$$

and v_{n+1} has the same expression as v_0 but with α interchanged with β.

5.3. Barycentric interpolation. Let $S = \{(x_i, f_i), i = 1, \dots n\}$ be a set of points of $\mathbb{R}^2$, with $x_i \neq x_j$ if $i \neq j$. Then the polynomial of the smallest degree $Q(x)$ such that $Q(x_i) = f_i$, $i = 1, \dots n$ (the Lagrange interpolation polynomial for the set S) can be written as

$$Q(x) = \frac{\displaystyle\sum_{i=1}^n \frac{u_i}{x-x_i} f_i}{\displaystyle\sum_{i=1}^n \frac{u_i}{x-x_i}}, \quad u_i = k \left(\prod_{j\neq i}^n (x_i - x_j) \right)^{-1}, \tag{5.1}$$

with k an arbitrary constant; the quantities u_i are called barycentric weights. This is called the barycentric formula of the Lagrange interpolation polynomial. Among the different formulas for computing Lagrange interpolants, this is one of the most efficient interpolation methods and it is a stable method [24].

For similar reasons that make the Gaussian quadrature optimal in the sense of the degree accuracy, it can be shown that the interpolation at the orthogonal polynomial nodes have particularly good convergence rates (see for instance [42]). A well-known example appearing in all textbooks is that of interpolation at the Chebyshev nodes, which smoothens the Runge phenomenon appearing when the nodes $\{x_i\}$ are equally spaced. We refer to [40] for additional examples of interpolation using Jacobi nodes and weights (some of them requiring few hundreds of nodes for double precision accuracy). In that paper the barycentric weights where computed with a Matlab version of the algorithm [20], and it is mentioned that the inclusion of the methods of [23] should improve the performance.

As discussed in [40], and as it will be easy to understand for the reasons we will now explain, the barycentric weights at the nodes of orthogonal polynomials can be easily related to the Gauss weights for this set of nodes; the same is true for the Gauss-Radau and Gauss-Lobatto variants. To see this, first we observe that the barycentric weights of Eq. (5.1) can be written as (taking $k = 1$):

$$u_i(x) = 1/\phi'(x_i), \quad \phi(x) = \prod_{j=1}^n (x - x_j).$$

Therefore, when the nodes are the roots of a polynomial (with only simple roots), the barycentric weights are inversely proportional to the derivative of the polynomial at the roots (with an arbitrary proportionality constant). Then, the Gaussian weights are proportional to the square of the barycentric weights up to, possibly, an elementary function of the nodes (see Eq. (4.1)). If we write the barycentric weights taking the square root, the barycentric weights for interpolation at the zeros of $P_n^{(\alpha,\beta)}(x)$ can be written as

$$u_i = (-1)^i \sin\theta_i \sqrt{w_i},$$

where $\sin\theta_i = \sin(\arccos x_i) = \sqrt{1 - x_i^2}$ and w_i are the corresponding Gauss-Jacobi weights. The alternating signs $(-1)^i$ are obvious because the derivative of the orthogonal polynomial takes different signs at consecutive zeros. Similar relations exist for the internal weights of Jacobi-Gauss-Radau ($x = -1$ or $x = +1$ are also nodes) and Jacobi-Gauss-Lobatto ($x = -1$ or $x = +1$ are also nodes) interpolations, as can be understood from the previous discussion on the associated quadrature rules.

Examples of applications employing barycentric interpolation at Jacobi points (or variants like Jacobi-Lobatto points) are the earlier mentioned references [40, 42]. The case of for Hermite-Fejér at Jacobi points was considered in [43].

5.4. Other applications and some advantages with respect to previous methods. Due to the many applications of Gauss quadrature and its variants (Gauss-Radau and Gauss-Lobatto) and barycentric interpolation, it is not surprising that Gauss-Jacobi nodes and weights pop up in a many applications, for many of which a moderately large number of nodes may appear.

For the particular case of Legendre quadratures, and as commented in [33], atmospheric models use quadratures with a large number of nodes (as an example of this see for instance [27], where 200 nodes Gauss-Legendre quadrature is employed). As mentioned in [25] (where quadratures with hundreds and even thousands of nodes are considered), one disadvantage of the Gaussian rules is that explicit formulas do not exists for the nodes and weights and that iterative methods are usually needed; however, similarly as in [4], we do provide explicit formulas which are iteration free, and that are fast enough to open the possibility of fast computations with varying numbers of nodes.

But, of course, not only the Legendre case is important and also the Gauss-Jacobi rules appear in a vast number of applications, particularly for integrals with end-point singularities. Just to mention a few particular recent examples of the many which can be found in the literature see [44] (mechanics), [1] (nuclear engineering) and [9] (approximation theory). On the other hand, the approximations in terms of Jacobi polynomials for different values of the parameters (Legendre, Gegenbauer, Chebyshev or general Jacobi) have, as discussed in [6], different advantages depending on the criterion of merit, and different applications use some of these features. In most of these applications, the barycentric interpolation at the polynomial zeros is at the heart of the proposed methods. Without aiming at being exhaustive, apart from the applications mentioned in [4] for the particular case of Gauss-Legendre, we add the spectral Gegenbauer method of [10] (where hundreds of nodes and weights are used), the Jacobi pseudospectral methods applied to optimal control problems of reference [35] and fractional differential equations of [36], for which the use of asymptotic methods allows for an application of the method for high accuracies (see Remark 14 of [35] and Remark 4.5 of [36]) or the Jacobi spectral methods for the Black-Scholes model of option pricing [28] which uses hundreds of nodes. Finally we recall

that for the computation of the barycentric interpolation formulas, it is important to rely on methods which are efficient for large or moderately large degrees, both for Lagrange [40] as well as for Hermite-Fejér [43] interpolation.

As we see, in a good number applications, and in particular for barycentric interpolation and (pseudo-)spectral methods, a moderately large or even a vary large number of nodes and weights is needed. It would be completely inefficient (as illustrated in [23]) as well as unreliable to compute these nodes and weights by the Golub-Welsch algorithm; instead, one should consider the more modern asymptotic methods, starting with [5, 23, 4, 39], continuing with our work on Gauss-Laguerre and Gauss-Hermite [19] and ending in the present paper, which extends the asymptotic methods of Bogaert to the more general and important case of Jacobi, and has advantages both in terms of accuracy and speed with respect to [23].

The present paper offers direct approximations for the Gauss-Jacobi nodes and weights in the same ranges as [23] but in a faster non-iterative way, and with double precision relative accuracy both for the nodes and the weights. We expect that these approximations will become part of forthcoming numerical software for computing Gauss-Jacobi nodes and weights and Jacobi barycentric weights. It is also expected that the fast and direct computation of Gauss-Jacobi nodes will have an impact on the implementation of spectral methods for solving a huge variety of problems. The algorithms are easily parallelizable, similarly as discussed in [5, 4] for the particular case of Gauss-Legendre, and differently from iterative algorithms like [23] (which can not be parallelized so efficiently because each node requires several repeated iterations). Furthermore, in applications where just the nodes are needed the advantage over [23] will be even larger in terms of speed, because for that algorithm computing the nodes was almost as expensive as computing the nodes and weights (as they require iterations), while our approximations for the nodes are much cheaper. When both the nodes and weights are computed, the speed-up will be in any case always larger than $2N$, with N the number of iterations required by the iterative method; our numerical experiments suggest that, in fact, the speed-up factor could be considerably larger. Also for smaller degrees n ($n \leq 20$) the approximations for the nodes are useful because they are more accurate than previous approximations and they can be used to speed-up iterative methods (just one Newton iteration would give more than double precision accuracy).

REFERENCES

[1] A. A. Heimlich, F. C. Silva, and A. S. Martinez, *Parallel GPU implementation of pwr reactor burnup*, IJMS, 91 (2016), pp. 135–141.

[2] R. A. Askey and R. Roy, *Chapter 5, Gamma function*, in NIST Handbook of Mathematical Functions, U.S. Dept. Commerce, Washington, DC, 2010, pp. 135–147. http://dlmf.nist.gov/5.

[3] P. Baratella and L. Gatteschi, *The bounds for the error term of an asymptotic approximation of Jacobi polynomials*, in Orthogonal polynomials and their applications (Segovia, 1986), vol. 1329 of Lecture Notes in Math., Springer, Berlin, 1988, pp. 203–221.

[4] I. Bogaert, *Iteration-free computation of Gauss-Legendre quadrature nodes and weights*, SIAM J. Sci. Comput., 36 (2014), pp. A1008–A1026.

[5] I. Bogaert, B. Michiels, and J. Fostier, *O(1) computation of Legendre polynomials and Gauss-Legendre nodes and weights for parallel computing*, SIAM J. Sci. Comput., 34

(2012), pp. C83–C101.

[6] J. P. Boyd and R. Petschek, *The relationships between Chebyshev, Legendre and Jacobi polynomials: the generic superiority of Chebyshev polynomials and three important exceptions*, J. Sci. Comput., 59 (2014), pp. 1–27.

[7] J. Bremer, *On the numerical calculation of the roots of special functions satisfying second order ordinary differential equations*, SIAM J. Sci. Comput., 39 (2017), pp. A55–A82.

[8] P. J. Davis and P. Rabinowitz, *Some geometrical theorems for abscissas and weights of Gauss type*, J. Math. Anal. Appl., 2 (1961), pp. 428–437.

[9] E. De Micheli and G. A. Viano, *The expansion in Gegenbauer polynomials: a simple method for the fast computation of the Gegenbauer coefficients*, J. Comput. Phys., 239 (2013), pp. 112–122.

[10] K. T. Elgindy, *High-order, stable, and efficient pseudospectral method using barycentric Gegenbauer quadratures*, App. Num. Math., 113 (2017), pp. 1–25.

[11] D. Elliott, *Uniform asymptotic expansions of the Jacobi polynomials and an associated function*, Math. Comp., 25 (1971), pp. 309–315.

[12] C. L. Frenzen and R. Wong, *A uniform asymptotic expansion of the Jacobi polynomials with error bounds*, Canad. J. Math., 37 (1985), pp. 979–1007.

[13] G. Gasper, *Formulas of the Dirichlet-Mehler type*, (1975), pp. 207–215. Lecture Notes in Math., Vol. 457.

[14] L. Gatteschi and G. Pittaluga, *An asymptotic expansion for the zeros of Jacobi polynomials*, in Mathematical Analysis, vol. 79 of Teubner-Texte Math., Teubner, Leipzig, 1985, pp. 70–86.

[15] W. Gautschi, *High-order Gauss-Lobatto formulae*, Numer. Algorithms, 25 (2000), pp. 213–222. Mathematical journey through analysis, matrix theory and scientific computation (Kent, OH, 1999).

[16] ———, *The circle theorem and related theorems for Gauss-type quadrature rules*, Electron. Trans. Numer. Anal., 25 (2006), pp. 129–137 (electronic).

[17] A. Gil and J. Segura, *Computing the real zeros of cylinder functions and the roots of the equation $x\mathcal{C}'_\nu + \gamma\mathcal{C}_\nu(x) = 0$*, Comput. Math. Appl., 64 (2012), pp. 11–21.

[18] A. Gil, J. Segura, and N. M. Temme, *Efficient computation of Laguerre polynomials*, Comput. Phys. Commun., 210 (2017), pp. 124–131.

[19] A. Gil, J. Segura, and N. M. Temme, *Asymptotic approximations to the nodes and weights of Gauss–Hermite and Gauss–Laguerre quadratures*, Stud. Appl. Math., 140 (2018).

[20] A. Glaser, X. Liu, and V. Rokhlin, *A fast algorithm for the calculation of the roots of special functions*, SIAM J. Sci. Comput., 29 (2007), pp. 1420–1438.

[21] G. H. Golub and J. H. Welsch, *Calculation of Gauss quadrature rules*, Math. Comp. 23 (1969), 221-230; addendum, ibid., 23 (1969), pp. A1–A10.

[22] E. Hahn, *Asymptotik bei Jacobi-Polynomen und Jacobi-Funktionen*, Math. Z., 171 (1980), pp. 201–226.

[23] N. Hale and A. Townsend, *Fast and accurate computation of Gauss-Legendre and Gauss-Jacobi quadrature nodes and weights*, SIAM J. Sci. Comput., 35 (2013), pp. A652–A674.

[24] N. J. Higham, *The numerical stability of barycentric Lagrange interpolation*, IMA J. Numer. Anal., 24 (2004), pp. 547–556.

[25] D. Hotta and M. Ujiie, *A nestable, multigrid-friendly grid on a sphere for global spectral models based on Clenshaw-Curtis quadrature*, Quaterly Journal of the Royal Meteorological Society, 144 (2018), p. 13821397.

[26] T. H. Koornwinder, R. Wong, R. Koekoek, and R. F. Swarttouw, *Chapter 18, Orthogonal polynomials*, in NIST Handbook of Mathematical Functions, U.S. Dept. Commerce, Washington, DC, 2010, pp. 435–484. http://dlmf.nist.gov/18.

[27] N. G. Loeb, S. Kato, and B. A. Wielicki, *Defining top-of-the-atmosphere flux reference level for earth radiation budget studies*, Journal of Climate, 15 (2002), pp. 3301–3309.

[28] B. F. Nteumagné, E. Pindza, and E. Maré, *Applying the barycentric Jacobi spectral method to price options with transaction costs in a fractional Black-Scholes framework*, Journal of Mathematical Finance, 4 (2014), pp. 35–46.

[29] F. W. J. Olver and L. C. Maximon, *Chapter 10, Bessel functions*, in NIST Handbook of Mathematical Functions, U.S. Dept. Commerce, Washington, DC, 2010, pp. 215–286. http://dlmf.nist.gov/10.

[30] G. Petrova, *Generalized Gauss-Radau and Gauss-Lobatto formulas with Jacobi weight functions*, BIT, 57 (2017), pp. 191–206.

[31] J. Segura, *Reliable computation of the zeros of solutions of second order linear ODEs using a fourth order method*, SIAM J. Numer. Anal., 48 (2010), pp. 452–469.

[32] J. Shen, T. Tang, and L.-L. Wang, *Spectral methods*, vol. 41 of Springer Series in Computa-

tional Mathematics, Springer, Heidelberg, 2011. Algorithms, analysis and applications.

[33] P. N. SWARZTRAUBER, *On computing the points and weights for Gauss-Legendre quadrature*, SIAM J. Sci. Comput., 24 (2002), pp. 945–954.

[34] G. SZEGŐ, *Orthogonal polynomials*, American Mathematical Society, Providence, R.I., fourth ed., 1975. American Mathematical Society, Colloquium Publications, Vol. XXIII.

[35] X. TANG, Z. LIU, AND Z. WANG, *Integral fractional pseudospectral methods for solving fractional optimal control problems*, Automatica, 62 (2015), pp. 304–311.

[36] X. TANG, Y. SHI, AND H. XU, *Fractional pseudospectral schemes with equivalence for fractional differential equations*, SIAM J. Sci. Comput., 39 (2017), pp. A966–A982.

[37] N. M. TEMME, *Asymptotic methods for integrals*, vol. 6 of Series in Analysis, World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, 2015.

[38] A. TOWNSEND, *The race for high order Gauss-Legendre quadrature*, SIAM News, March (2015).

[39] A. TOWNSEND, T. TROGDON, AND S. OLVER, *Fast computation of Gauss quadrature nodes and weights on the whole real line*, IMA J. Numer. Anal., 36 (2016), pp. 337–358.

[40] H. WANG, D. HUYBRECHS, AND S. VANDEWALLE, *Explicit barycentric weights for polynomial interpolation in the roots or extrema of classical orthogonal polynomials*, Math. Comp., 83 (2014), pp. 2893–2914.

[41] G. N. WATSON, *A treatise on the theory of Bessel functions*, Cambridge Mathematical Library, Cambridge University Press, Cambridge, 1944. Second edition.

[42] S. XIANG, *On interpolation approximation: convergence rates for polynomial interpolation for functions of limited regularity*, SIAM J. Numer. Anal., 54 (2016), pp. 2081–2113.

[43] S. XIANG AND G. HE, *The fast implementation of higher order Hermite-Fejér interpolation*, SIAM J. Sci. Comput., 37 (2015), pp. A1727–A1751.

[44] K. B. YILMAZ, I. COMEZ, Y. B., M. GÜLER, AND S. EL-BORGI, *Frictional receding contact problem for a graded bilayer system indented by a rigid punch*, Int. J. Mech. Sci., 41 (2018), pp. 127–142.