A more accurate algorithm for computing the Geronimus transformation

María I. Bueno

Abstract

Given a family of polynomials orthogonal with respect to a measure \(d\mu\), it is often necessary to find another family of polynomials orthogonal with respect to the measure \(r(x) d\mu\) where \(r(x)\) is a rational function. The basic Geronimus transformation computes the polynomials orthogonal with respect to \(\frac{1}{x-\alpha} d\mu\), where \(\alpha\) is a real number, in terms of the polynomials orthogonal with respect to the measure \(d\mu\). In this work, we show that the standard algorithm to compute this transformation is not stable. We propose a new algorithm and prove that it is more accurate than the previous one. We also prove that this new algorithm is componentwise forward stable, which means that the obtained forward errors are of similar magnitude to those produced by a backward stable algorithm. Besides we provide a condition number that allows us to estimate forward errors in \(O(n)\) flops.

Key words: Geronimus transformation, Christoffel transformation, accuracy, condition number, forward stability, roundoff error analysis.
PACS: 65G50, 65F30, 65F35, 42C05

1 Introduction

Let \(d\mu\) be a real measure with finite moments and let \(\{P_n\}_{n=0}^\infty\) be a sequence of monic polynomials orthogonal with respect to \(d\mu\) [6]. Every sequence of monic orthogonal polynomials satisfies a three-term recurrence relation

\[ xP_n(x) = P_{n+1}(x) + B_{n+1}P_n(x) + G_nP_{n-1}(x), \]

This research has been partially supported by Dirección General de Investigación (Ministerio de Ciencia y Tecnología) of Spain through grants BFM2003-06335-C03-02 and MTM2006-06671 as well as by the Postdoctoral Fellowship EX2004-0658 provided by Ministerio de Educación y Ciencia of Spain.

Preprint submitted to Elsevier Science 8 September 2006
\[ P_{-1}(x) \equiv 0, \quad P_0(x) \equiv 1, \quad B_n, G_n \in \mathbb{R}, \quad G_n \neq 0 \quad \text{for all } n. \]

The previous set of equations can be written in matrix notation in the following way

\[ xp = Jp, \]

where \( p = [P_0(x), P_1(x), P_2(x), \ldots]^T \) and

\[
J = \begin{bmatrix}
B_1 & 1 & 0 & \cdots \\
G_1 & B_2 & 1 & \cdots \\
0 & G_2 & B_3 & \cdots \\
\vdots & \vdots & \vdots & \ddots
\end{bmatrix}.
\]

The semi-infinite tridiagonal matrix \( J \) is called the monic Jacobi matrix associated with \( \{P_n\} \). Although it is very unusual to denote the entries of a matrix by capital letters, since we will deal with an algorithm involving two monic Jacobi matrices, for the sake of clarity, we denote by capital letters the entries in the input matrix and we denote by the same lowercase letters the entries in the output matrix.

In the literature, numerous results studying the connection between the recurrence relations of polynomials orthogonal with respect to two allied measures can be found [4,28,9,16,1]. This relationship can be extended to the corresponding Jacobi matrices. Let us consider the linear functional \( L \) defined in the vector space of polynomials with real coefficients given by

\[ L(p) = \int_{\mathbb{R}} p(x) d\mu(x). \]

If \( \pi(x) \) denotes a polynomial, the new functional \( L_1 \) given by

\[ L_1(p) = \int_{\mathbb{R}} p(x) \pi(x) d\mu(x), \]

is called a polynomial perturbation of \( L \) (or equivalently, of \( d\mu(x) \)). Then, given a measure and a polynomial perturbation of this measure, the transformation that gives the monic Jacobi matrix associated with the perturbed measure in terms of the monic Jacobi matrix associated with the initial measure is called the Christoffel transformation or Darboux transformation. In particular, if \( \pi(x) \) is a polynomial of degree one, it is called basic Christoffel transformation.

Let us now consider the following rational transformation of the measure \( d\mu(x) \),

\[ L_2(p) = \int_{\mathbb{R}} p(x) \frac{1}{\pi(x)} d\mu(x) + \sum_{i=1}^{r} C_i p(\alpha_i), \quad (1) \]

where \( C_i \) denotes a nonzero constant for \( i = 1, \ldots, r \) and \( \{\alpha_i\}_{i=1}^{r} \) denotes the set of roots of the polynomial \( \pi(x) \). The new functional \( L_2 \) is a rational per-
turbation of $L$, and the transformation that gives the monic Jacobi matrix associated with the perturbed functional in terms of the monic Jacobi matrix associated with the initial one is called Geronimus transformation or Darboux transformation with parameter. This transformation can be considered as reciprocal of the Christoffel transformation in a very well defined sense as we will show in Section 2. Again, if $\pi(x)$ is a polynomial of degree one, the Geronimus transformation is called basic.

The Christoffel transformation was first studied by Christoffel in 1858 [7] and the Geronimus transformation was first studied by Geronimus in 1940. Among numerous papers by Geronimus on orthogonal polynomials there are two whose ideas anticipated many investigations in modern mathematical physics. The first nontrivial application of these transformations was proposed by Geronimus himself in [14]. This application is connected to the problem of classifying all sequences of orthogonal polynomials such that its derivatives form another set of orthogonal polynomials. In the last two decades, these transformations have attracted the interest of various specialists in different branches of mathematics and mathematical physics for its applications to different topics as discrete integrable systems [23,26,27], quantum mechanics, bispectral transformations in orthogonal polynomials [17–19], and, in the context of Numerical Analysis, to the computation of quadrature rules [9,10,13,16,21].

In a previous paper [3], it is shown that the standard algorithm to compute the basic Christoffel transformation is not stable. In that paper, a more accurate algorithm to compute that transformation is presented. It was also proven that this new algorithm is componentwise forward stable. We say that an algorithm is “forward stable” if the forward errors are of similar magnitude to those produced by a backward stable algorithm [20].

In the literature, we have not found any formal analysis of the stability and sensitivity of the standard algorithm to compute the basic Geronimus transformation. However, in [12] the following comment can be read: “This algorithm is more useful, and more effective, the closer $\alpha$ is to the endpoints of the support of the measure. As $\alpha$ moves away from the support, the algorithm quickly becomes unstable.” Numerical experiments support this assessment.

In this paper, we show that the standard algorithm for computing the basic Geronimus transformation can be expressed as a modified Christoffel transformation depending on two parameters $\alpha$ and $s$. We also prove that the numerical properties of the algorithm for computing the Geronimus transformation can be derived from the properties of the algorithm for computing the Christoffel transformation. It is remarkable that the value of the parameter $s$ has no effect on the stability and sensibility of the algorithm to compute the Geronimus transformation. Based on the results in [3], we propose a new algorithm for computing the basic Geronimus transformation which is more
accurate than the standard one. Moreover, we show that for large enough values of the shift $\alpha$ the algorithm becomes accurate, which means that it produces outputs with small componentwise forward errors. Moreover, if the parameter $s$ satisfies $|s| = O(|B_1|)$, for large enough values of $\alpha$ the algorithm becomes stable.

Finally, we prove that the new algorithm for computing the basic Geronimus transformation is componentwise forward stable. It should be remarked that the new algorithm is not componentwise backward stable. It is not either componentwise stable in the mixed forward-backward sense [20]. No need to say that the forward stability does not imply that the forward errors are small. For that reason we also provide a condition number that allows us to estimate the forward errors produced by the algorithm in $O(n)$ flops.

2 Matrix version of Christoffel and Geronimus transformation

In this section we present the matrix version of the basic Christoffel transformation and the basic Geronimus transformation. The standard algorithms to compute these transformations can be derived from them in a straightforward way.

Consider a monic Jacobi matrix $J$ associated with a real measure $d\mu$ and let $\alpha$ be a real number such that $J - \alpha I$ has a unique LU factorization. If $J - \alpha I = LU$ denotes the LU factorization without pivoting of $J - \alpha I$, where $L$ is unit lower triangular, then the matrix version of the basic Christoffel or Darboux transformation with shift $\alpha$ [9,13,16,21] is given by

$$J - \alpha I = LU, \quad J_1 = UL + \alpha I,$$

where $J_1$ is the monic Jacobi matrix associated with the measure $(x-\alpha)d\mu(x)$, and the factors $L$ and $U$ have the following structure

$$
L = \begin{bmatrix}
1 & 0 & 0 & \ldots \\
\ell_1 & 1 & 0 & \ldots \\
0 & \ell_2 & 1 & \ldots \\
0 & 0 & \ell_3 & \ldots \\
\vdots & \vdots & \vdots & \ddots 
\end{bmatrix}, \quad
U = \begin{bmatrix}
u_1 & 1 & 0 & \ldots \\
0 & \nu_2 & 1 & \ldots \\
0 & 0 & \nu_3 & \ldots \\
0 & 0 & 0 & \nu_4 \\
\vdots & \vdots & \vdots & \ddots 
\end{bmatrix}.
$$

From now on, we refer to the transformation given in (2) as Christoffel transformation with shift $\alpha$. A direct application of (2) leads to the standard algorithm to compute Christoffel transformation.
The finite version of the transformation given in (2) is

\[ J(B, G) - \alpha I = LU, \quad J(b, g) = (UL + \alpha I)_{n-1}, \]

where \( J(B, G) \) is the \( n \times n \) leading principal submatrix of \( J \), with \( B = [B_1, ..., B_n]^T, G = [G_1, ..., G_{n-1}]^T \). Moreover, \((M)_{n-1}\) denotes the leading principal submatrix of order \( n - 1 \) for any matrix \( M \), and \( J(b, g) \) is the \( n - 1 \) leading principal submatrix of \( J_1 \), being \( b = [b_1, ..., b_{n-1}]^T \) the elements on the main diagonal of \( J(b, g) \), and \( g = [g_1, ..., g_{n-2}]^T \) the elements on the first lower subdiagonal, i.e., the entries in the positions \((i + 1, i)\), \( 1 \leq i \leq n - 2 \). The \( n \times n \) matrix equation \( J(B, G) - \alpha I = LU \) denotes the LU factorization of \( J(B, G) - \alpha I \), and, here, \( L \) and \( U \) are the \( n \times n \) leading principal submatrices of the semi-infinite \( L \) and \( U \) matrices appearing in (2). We have not changed the notation for \( L \) and \( U \) for the sake of simplicity.

Let us now consider the basic Geronimus transformation, that is, the rational modification of a measure \( d\mu \) by a polynomial of degree one. This transformation depends on a second parameter \( s \) as we will explain below. Let \( J \) be a monic Jacobi matrix, let \( s \) be any nonzero real number, and let \( \alpha \) be a real number such that \( J - \alpha I \) has an UL factorization. The matrix version of the Geronimus transformation with parameter \( s \) and shift \( \alpha \) \([9,13,16,21]\) is given by

\[ J - \alpha I = U_s L_s, \quad J_s = L_s U_s + \alpha I, \quad (4) \]

where \( J_s \) is the monic Jacobi matrix associated with the functional given in (1) where \( \pi(x) = x - \alpha \), and \( C \) is a constant depending on \( s \). The factors \( L_s \) and \( U_s \) are as in (3) with \( u_1 = s \). From now on, we refer to the transformation given in (4) as Geronimus transformation with parameter \( s \) and shift \( \alpha \).

The expression \( J - \alpha I = U_s L_s \) denotes the UL factorization with parameter \( s \) of \( J - \alpha I \). This UL factorization does not correspond to the standard one since it is not applicable to a semi-infinite matrix. In order to compute \( U_s \) and \( L_s \), let us consider two factors as in (3) and compute its product.

\[
\begin{bmatrix}
  u_1 + l_1 & 1 & 0 & 0 & \ldots \\
  u_2l_1 & u_2 + l_2 & 1 & 0 & \ldots \\
  0 & u_3l_2 & u_3 + l_3 & 1 & \ldots \\
  0 & 0 & u_4l_3 & u_4 + l_4 & \ldots \\
  \vdots & \vdots & \vdots & \vdots & \ddots
\end{bmatrix}
\]

Equating the entries of this matrix and \( J - \alpha I \) we get an algorithm to compute an UL factorization of a shifted monic Jacobi matrix. It is important to point out that this new UL factorization is not unique. The entry \( u_1 \) can be considered a free parameter that we denote by \( s \).
In particular, if \( J(B, G) \) denotes the \( n \)-by-\( n \) leading principal submatrix of \( J \), then
\[
J(B, G) - \alpha I = U_s L_s + l_n e_n e_n^T, \quad J_s(b, g) = L_s U_s + \alpha I,
\]
is the finite version of the Geronimus transformation with parameter \( s \) and shift \( \alpha \), where \( e_n \) denotes the \( n \)-th column of the \( n \)-by-\( n \) identity matrix, and \( l_n \) is the entry of \( L_s \) in the position \((n + 1, n)\). Again, for the sake of simplicity, we write \( U_s \) and \( L_s \) instead of \((U_s)_n\) and \((L_s)_n\) to denote the \( n \)-by-\( n \) leading principal submatrices of \( U_s \) and \( L_s \).

**Remark 1** In practice, in order to compute the Geronimus transform \( J_s(b, g) \) of \( J(B, G) - \alpha I \), we do not need the input \( B_n \) nor the entry \( l_n \). Therefore, we only need to compute entries \( u_1, \ldots, u_n, l_1, \ldots, l_{n-1} \) in the factors \( U_s \) and \( L_s \) from the inputs \( B_1, \ldots, B_{n-1}, G_1, \ldots, G_{n-1} \).

We mentioned in the introduction that Christoffel and Geronimus transformations can be considered one the reciprocal of the other in a certain sense. Let us denote by \( C(\alpha) \) and \( G(\alpha, s) \) a Christoffel transformation with shift \( \alpha \) and a Geronimus transformation with parameter \( s \) and shift \( \alpha \), respectively. Then, the following relations hold for the composition of both transformations:
\[
C(\alpha)G(\alpha, s) = 1, \quad G(\alpha, s)C(\alpha) = U(\alpha, s),
\]
where 1 denotes the identity function, and \( U(\alpha, s) \) is the so-called Uvarov transformation which has the effect of adding a single mass to the measure at the point \( x = \alpha \).

### 2.1 Geronimus transformation as a modified Christoffel transformation

In this section we show that the Geronimus transformation can be expressed as a modified Christoffel transformation applied to a tridiagonal matrix which is not a monic Jacobi matrix. In the following sections we will prove that this connection between both transformations allows us to deduce the numerical properties of the Geronimus transformation from those of the Christoffel transformation. Let \( J(B, G) \) be an \( n \times n \) monic Jacobi matrix. We call the *Christoffel companion of \( J(B, G) \) with parameter \( \alpha \)* the \((n + 1) \times (n + 1)\) tridiagonal matrix \( J_\alpha(B, G) \) given by
\[
J_\alpha(B, G) = \begin{bmatrix}
\alpha & e_1^T \\
0 & J(B, G)
\end{bmatrix}
\]
where \( e_1 \) denotes the first column of the identity matrix of size \( n \).

Next we show that the Geronimus transform of \( J(B, G) \) with parameter \( s \) and shift \( \alpha \) can be computed as a Christoffel transform with shift \( \alpha \) of \( J_\alpha(B, G) \).
Notice that using the notation in (5), we can factorize $J_\alpha(\tilde{B}, \tilde{G}) - \alpha I$ as follows

$$J_\alpha(\tilde{B}, \tilde{G}) - \alpha I = \begin{bmatrix} e_1^t & 0 \\ U_s & e_n^t \end{bmatrix} \begin{bmatrix} 0 & L_s \\ 0 & l_n e_n^t \end{bmatrix},$$

where the first factor is lower triangular and the second one is upper triangular.

Note that there is not a unique LU factorization of $J_\alpha(\tilde{B}, \tilde{G}) - \alpha I$ since this matrix is singular. Among all the possible LU factorizations of $J_\alpha(\tilde{B}, \tilde{G}) - \alpha I$, the set of those factorizations in which both factors are bidiagonal has dimension 1 [8]. This means that this set can be parametrized by one single parameter. This parameter is $s$. Recall that $u_1 = s$. Hence, (7) gives the set of all possible LU factorizations of $J_\alpha(\tilde{B}, \tilde{G}) - \alpha I$ with bidiagonal factors parametrized by $s$. If we denote by $\tilde{L}_s$ and $\tilde{U}_s$ the factors obtained in (7), the Geronimus transform $J_s(b, g)$ of $J(B, G)$ with parameter $s$ and shift $\alpha$ can be obtained as

$$J_s(b, g) = (\tilde{U}_s \tilde{L}_s + \alpha I)_n.$$  

Recall from Remark 1 that, in order to compute the Geronimus transform of $J(B, G)$ with parameter $s$ and shift $\alpha$, $B_n$ is not an input of the algorithm and there is no need to compute $l_n$. However, we include this entries in order to get a matrix version of the algorithm.

Notice that (7) and (8) say that $J_s(b, g)$ is a Christoffel-like transform with shift $\alpha$ of the tridiagonal matrix $J_\alpha(\tilde{B}, \tilde{G})$. This is not a standard Christoffel transformation for the following reasons:

1. One of the entries of $J_\alpha(\tilde{B}, \tilde{G})$ depends on the shift $\alpha$ and, therefore, varies with $\alpha$.
2. The tridiagonal matrix $J_\alpha(\tilde{B}, \tilde{G})$ is not a monic Jacobi matrix since the entry in position $(2, 1)$ is zero.
3. This transformation has as inputs not only the entries of $J_\alpha(\tilde{B}, \tilde{G})$, which include the shift $\alpha$, but also the free parameter $s$.
4. The LU factorization of $J_\alpha(\tilde{B}, \tilde{G})$ is not unique.
5. The output of the algorithm is a matrix depending on $s$ too.

All the previous comments and observations lead to the following definition:

**Definition 2** Let $J(B, G)$ be a monic Jacobi matrix and let $J_\alpha(\tilde{B}, \tilde{G})$ be its Christoffel companion with parameter $\alpha$. Let $s$ be a nonzero real number. If $J_\alpha(\tilde{B}, \tilde{G}) - \alpha I = \tilde{L}_s \tilde{U}_s$ denotes the LU factorization of $J_\alpha(\tilde{B}, \tilde{G}) - \alpha I$ with bidiagonal factors and parameter $s$, where $\tilde{L}_s(2, 1) = s$, then the matrix $J_s(b, g)$ obtained as

$$J_\alpha(\tilde{B}, \tilde{G}) - \alpha I = \tilde{L}_s \tilde{U}_s, \quad J_s(b, g) = (\tilde{U}_s \tilde{L}_s + \alpha I)_n,$$
is called the modified Christoffel transform of \( J_\alpha(\tilde{B}, \tilde{G}) \) with parameter \( s \) and shift \( \alpha \). The transformation itself is said to be a modified Christoffel transformation of \( J_\alpha(\tilde{B}, \tilde{G}) \) with parameter \( s \) and shift \( \alpha \).

**Remark 3** Taking into account the previous definition, if \( J(B, G) \) is a monic Jacobi matrix and \( J_\alpha(\tilde{B}, \tilde{G}) \) is its Christoffel companion with parameter \( \alpha \), the Geronimus transform with parameter \( s \) and shift \( \alpha \) of \( J(B, G) \) is just the modified Christoffel transform of \( J_\alpha(\tilde{B}, \tilde{G}) \) with parameter \( s \) and shift \( \alpha \). Note that the algorithms derived from (5) and (9) to compute \( J_s(b, g) \) are exactly the same. In the sequel, we only refer to (9) because of its analogy with the Christoffel transformation. However, it is clear that all the conclusions are also valid for the algorithm derived from (5).

3 A new algorithm for computing the basic Geronimus transformation

A direct application of (2) leads to the standard algorithm to compute the Christoffel transformation. In [3] we presented a new algorithm for computing the Christoffel transformation which is more accurate than the standard one. The new algorithm was obtained by a small modification. Now we show that this modification applied to a modified Christoffel transformation produces an algorithm to compute Geronimus transformation which is also more accurate than the standard one.

The following pseudocode gives the standard algorithm for computing the modified Christoffel transform with parameter \( s \) and shift \( \alpha \) of the Christoffel companion with parameter \( \alpha \) of an \( n \)-by-\( n \) monic Jacobi matrix.

**Algorithm 1** Let \( J(B, G) \) be a monic Jacobi matrix and let \( J_\alpha(\tilde{B}, \tilde{G}) \) be its Christoffel companion with parameter \( \alpha \). This algorithm computes the modified Christoffel transform with parameter \( s \) and shift \( \alpha \) of \( J_\alpha(\tilde{B}, \tilde{G}) \).

\[
\tilde{u}_1 = 0, \tilde{l}_1 = s
\]

for \( i = 1 : n - 1 \)

\[
b_i = \tilde{u}_i + \tilde{l}_i + \alpha
\]

\[
\tilde{u}_{i+1} = \tilde{B}_{i+1} - \alpha - \tilde{l}_i
\]

\[
g_i = \tilde{u}_{i+1} \tilde{l}_i
\]

\[
\tilde{l}_{i+1} = \tilde{G}_{i+1}/\tilde{u}_{i+1}
\]
Let us define the new variables \( \{ \tilde{t}_i \}_{i=1}^n \) as \( \tilde{t}_i := \tilde{u}_i + \alpha \). Then, the following new algorithm to compute the modified Christoffel transformation with parameter \( s \) and shift \( \alpha \) of \( J_\alpha(\tilde{B}, \tilde{G}) \) can be derived. Notice that the variables \( \tilde{u}_1, ..., \tilde{u}_n \) have disappeared since they have been replaced by \( \tilde{t}_1, ..., \tilde{t}_n \).

**Algorithm 2** Let \( J(B, G) \) be an \( n \times n \) monic Jacobi matrix and let \( J_\alpha(\tilde{B}, \tilde{G}) \) be its Christoffel companion matrix with parameter \( \alpha \). This algorithm computes the modified Christoffel transformation \( J_s(b, g) \) of order \( n \) with parameter \( s \) and shift \( \alpha \) of \( J_\alpha(\tilde{B}, \tilde{G}) \).

\[
\begin{align*}
\tilde{t}_1 &= \alpha, \quad \tilde{l}_1 = s, \\
& \text{for } i = 1 : n - 1 \\
& \quad b_i = \tilde{t}_i + \tilde{l}_i \\
& \quad \tilde{t}_{i+1} = \tilde{B}_{i+1} - \tilde{l}_i \\
& \quad g_i = (\tilde{t}_{i+1} - \alpha) \tilde{l}_i \\
& \quad \tilde{l}_{i+1} = \tilde{G}_{i+1} / (\tilde{t}_{i+1} - \alpha) \\
& \text{end} \\
& b_n = \tilde{t}_n + \tilde{l}_n
\end{align*}
\]

We have obtained a new algorithm to compute the Geronimus transformation. The computational cost of Algorithm 2 is \( 5n - 4 \) flops, while the cost of Algorithm 1 is \( 6n - 5 \). Moreover, the same number of divisions are performed in both algorithms.

The modification applied to Algorithm 1 to get Algorithm 2 seems to be slight but it has an essential influence on the stability and accuracy of the new algorithm. One of the main reasons why this algorithm has a much better numerical behavior than Algorithm 1 is that in the computation of the outputs \( b(i) \) by Algorithm 1 some cancellations may arise. A significative situation where this problem can be clearly understood is the case of large shifts \( \alpha \): it can easily be shown that \( \lim_{|\alpha| \to \infty} \tilde{l}_k = 0 \) –see Lemma 7 in Section 4–, therefore \( \tilde{u}_i = \tilde{B}_i - \alpha - \tilde{l}_{i-1} \sim -\alpha \) when \( |\alpha| \to \infty \), and then \( b_i = \tilde{l}_i + \tilde{u}_i + \alpha \sim (-\alpha) + \alpha \) when \( |\alpha| \to \infty \). The reader should notice that this cancellation is avoided in Algorithm 2.
Taking into account Remark 3, we can deduce that the modification applied to Algorithm 1 produces the following changes in the matrix version (5) of the Geronimus transformation:

\[ J(B, G) - \alpha I = U_s(T_s - \alpha B) + l_n e_n e_n^t, \quad J_s(b, g) = (T_s - \alpha B) U_s + \alpha I, \]

where \( U_s \) is an \( n \times n \) upper bidiagonal matrix with \( u_1, ..., u_n \) in the positions \((1, 1), (2, 2), ..., (n, n)\), \( T_s \) is an \( n \times n \) lower bidiagonal matrix with \( t_1, ..., t_{n-1} \) in the positions \((2, 1), (3, 2), ..., (n, n-1)\) and 1's on the main diagonal. Finally,

\[ B = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{pmatrix}. \]

Below we present some numerical experiments to show that Algorithm 2 is more accurate than Algorithm 1. We compare the forward errors produced by both algorithms for different Jacobi matrices and for different values of \( \alpha \). We keep the value of the parameter \( s \) fixed in each experiment since no significant variation in the results can be observed when the parameter \( s \) varies. (See Tables 2 and 3) We have tested the following types of monic Jacobi matrices:

1. A 3 \( \times \) 3 monic Jacobi matrix with \( B = [10^{-6}, -3 \cdot 10^{-6}, -1] \) and \( G = [2 \cdot 10^{-6}, 10^{-6}] \).
2. Monic Jacobi matrices of dimension 30 \( \times \) 30 associated with Laguerre polynomials with parameter \( a = -19/10 + k \), where \( k = 1 : 20 \).
3. Monic Jacobi matrices of dimension 30 \( \times \) 30 associated with Jacobi polynomials with parameters \( a = -19/10 + k, \ b = (-9 + k)/10 \), where \( k = 1 : 20 \).
4. Monic Jacobi matrices of dimension 30 \( \times \) 30 associated with Bessel polynomials with parameter \( a = -101/7 + k^2 \), where \( k = 1 : 20 \).
5. Monic Jacobi matrix of dimension 30 \( \times \) 30 associated with Hermite polynomials.

For each of these matrices we have computed the following componentwise forward error:

\[ \max \left\{ \max_{k=1 \ldots n-1} \left\{ \frac{|b_k - \hat{b}_k|}{b_k} \right\}, \quad \max_{k=1 \ldots n-2} \left\{ \frac{|g_k - \hat{g}_k|}{g_k} \right\} \right\}, \tag{10} \]

where \( \hat{b}_k \) and \( \hat{g}_k \) denote the outputs computed by Algorithm 1 or 2 in standard double precision, i.e., \( \epsilon \approx 1.11 \times 10^{-16} \) is the unit roundoff of the finite arithmetic, while \( b_k \) and \( g_k \) denote the outputs obtained by running the algorithms with 64 decimal digits of precision. The experiments have been done
using MATLAB 5.3, and we have used the variable precision arithmetic of the Symbolic Math Toolbox of MATLAB. In all our tests, theoretical error bounds guarantee that the outputs obtained by running Algorithm 1 and 2 with 64 decimal digits of precision have more than 50 significant decimal digits.

For the different types of Jacobi matrices considered, \( v_{m1} \) and \( v_{m2} \) are vectors whose components are the componentwise forward errors obtained for each matrix by applying Algorithm 1 and 2, respectively. The results we have got are presented in Table 1, where for the sake of brevity only \( \max(v_{m1}) \) and \( \max(v_{m2}) \) are shown. Notice that the examples relative to the \( 3 \times 3 \) matrix and the Hermite polynomials only consider one matrix for each value of \( \alpha \) and, therefore \( v_{m1} \) and \( v_{m2} \) are just numbers. However, in Table 1 we keep the notation \( \max(v_{m1}) \) and \( \max(v_{m2}) \) for simplicity.

<table>
<thead>
<tr>
<th>3x3 Matrix, s=1.3</th>
<th>( \alpha = 1 )</th>
<th>( \alpha = 0.3 )</th>
<th>( \alpha = 0 )</th>
<th>( \alpha = -1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \max(v_{m1}) )</td>
<td>( 1.3 \cdot 10^{-11} )</td>
<td>( 7.3 \cdot 10^{-12} )</td>
<td>( 1.7 \cdot 10^{-16} )</td>
<td>( 7.1 \cdot 10^{-12} )</td>
</tr>
<tr>
<td>( \max(v_{m2}) )</td>
<td>( 1.7 \cdot 10^{-16} )</td>
<td>( 2.2 \cdot 10^{-16} )</td>
<td>( 1.7 \cdot 10^{-16} )</td>
<td>( 5.7 \cdot 10^{-16} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Laguerre, s=17</th>
<th>( \alpha = 0 )</th>
<th>( \alpha = -100 )</th>
<th>( \alpha = -10^4 )</th>
<th>( \alpha = -10^6 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \max(v_{m1}) )</td>
<td>( 3 \cdot 10^{-14} )</td>
<td>( 4.12 \cdot 10^{-14} )</td>
<td>( 3.3 \cdot 10^{-12} )</td>
<td>( 7.8 \cdot 10^{-11} )</td>
</tr>
<tr>
<td>( \max(v_{m2}) )</td>
<td>( 3 \cdot 10^{-14} )</td>
<td>( 5.1 \cdot 10^{-15} )</td>
<td>( 1.4 \cdot 10^{-14} )</td>
<td>( 1.4 \cdot 10^{-14} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Jacobi, s=7</th>
<th>( \alpha = 0 )</th>
<th>( \alpha = 10 )</th>
<th>( \alpha = 100 )</th>
<th>( \alpha = 10^4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \max(v_{m1}) )</td>
<td>( 3.6 \cdot 10^{-16} )</td>
<td>( 7.7 \cdot 10^{-15} )</td>
<td>( 6.7 \cdot 10^{-13} )</td>
<td>( 8.3 \cdot 10^{-10} )</td>
</tr>
<tr>
<td>( \max(v_{m2}) )</td>
<td>( 3.6 \cdot 10^{-16} )</td>
<td>( 2 \cdot 10^{-16} )</td>
<td>( 1.5 \cdot 10^{-16} )</td>
<td>( 2 \cdot 10^{-16} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bessel, s=38</th>
<th>( \alpha = 0 )</th>
<th>( \alpha = -10 )</th>
<th>( \alpha = -100 )</th>
<th>( \alpha = -10^3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \max(v_{m1}) )</td>
<td>( 2 \cdot 10^{-12} )</td>
<td>( 1.2 \cdot 10^{-12} )</td>
<td>( 1.2 \cdot 10^{-11} )</td>
<td>( 9.7 \cdot 10^{-10} )</td>
</tr>
<tr>
<td>( \max(v_{m2}) )</td>
<td>( 2 \cdot 10^{-12} )</td>
<td>( 1.4 \cdot 10^{-15} )</td>
<td>( 4.3 \cdot 10^{-16} )</td>
<td>( 3.3 \cdot 10^{-16} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hermite, s=29</th>
<th>( \alpha = -100 )</th>
<th>( \alpha = -10^{-4} )</th>
<th>( \alpha = 10 )</th>
<th>( \alpha = 10^6 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \max(v_{m1}) )</td>
<td>( 2.8 \cdot 10^{-12} )</td>
<td>( 1.9 \cdot 10^{-15} )</td>
<td>( 2.6 \cdot 10^{-14} )</td>
<td>( 2.2 \cdot 10^{-4} )</td>
</tr>
<tr>
<td>( \max(v_{m2}) )</td>
<td>( 6.3 \cdot 10^{-15} )</td>
<td>( 1.9 \cdot 10^{-15} )</td>
<td>( 3.9 \cdot 10^{-15} )</td>
<td>( 2.2 \cdot 10^{-15} )</td>
</tr>
</tbody>
</table>

Table 1 Errors of Algorithms 1 and 2.

Notice that in most of the examples presented in Table 1, while the forward errors from Algorithm 1 increase as the absolute value of \( \alpha \) increases, the forward errors from Algorithm 2 decrease as \( |\alpha| \) grows.

In the next section we present a tight first order bound for the forward errors produced by Algorithm 2, and we show that this bound is always smaller than a corresponding bound for Algorithm 1. The bound for Algorithm 2 is
given in terms of the condition number of the problem taking into account the proper backward errors. We also show that when $|\alpha|$ is large enough Algorithm 2 becomes stable and accurate. Finally, we say that the errors produced by Algorithm 2 are the best one can expect, because they reflect the sensitivity of Christoffel transformation to componentwise relative perturbations of $O(\epsilon)$ in the data.

4 Main results

The results in this section can be obtained using the same techniques used in [3]. For this reason, we do not include any proofs. It can be easily proven that the new algorithm to compute the modified Christoffel transformation, i.e., Algorithm 2, has the same numerical behavior as the new algorithm introduced in [3] to compute the standard Christoffel transformation.

The following theorem presents a backward error analysis of Algorithm 2. We use the standard model of floating point arithmetic.

**Theorem 4** Let $J(B, G)$ be an $n \times n$ monic Jacobi matrix and let $J_\alpha(\hat{B}, \hat{G})$ be its Christoffel companion with parameter $\alpha$. Let $\alpha$ and $s$ be real numbers such that $s \neq 0$. Consider the nearest floating point number $\hat{\alpha}$ and $\hat{s}$ to $\alpha$ and $s$, respectively. Let $J_\alpha(\hat{B}, \hat{G})$ be the modified Christoffel transform of $J_\alpha(\hat{B}, \hat{G})$ with parameter $s$ and shift $\alpha$. Let us apply Algorithm 2 to the matrix with floating entries $J_\alpha(\hat{B}, \hat{G})$ where

$$\hat{B}_i = \hat{B}_i(1 + \epsilon_{\hat{B}_i}), \quad \hat{G}_i = \hat{G}_i(1 + \epsilon_{\hat{G}_i}), \quad 1 \leq i \leq n - 1,$$

and

$$\max_{1 \leq i \leq n-1} \{|\epsilon_{\hat{B}_i}|, |\epsilon_{\hat{G}_i}|\} \leq \frac{Cu}{1 - Cu},$$

for a positive integer number $C$ such that $Cu \ll 1$. If $J_\hat{s}(\hat{b}, \hat{g})$ is the matrix computed by Algorithm 2, and $\hat{L}_\hat{s}, \hat{T}_\hat{s}$ are the computed intermediate matrices appearing in the same algorithm, then

$$J_\alpha(\hat{B} + \Delta \hat{B}, \hat{G} + \Delta \hat{G}) - \hat{\alpha}I = \hat{L}_\hat{s}(\hat{T}_\hat{s} - \alpha I)_s,$$
$$J_\hat{s}(\hat{b} + \Delta \hat{b}, \hat{g} + \Delta \hat{g}) = \left((\hat{T}_\hat{s} - \alpha I)\hat{L}_\hat{s} + \hat{\alpha}I\right)_n,$$

where

$$|\hat{\alpha} - \alpha| \leq u|\alpha|, \quad |\hat{s} - s| \leq u|s|,$$
$$|\Delta \hat{B}_1| \leq \frac{Cu}{1 - Cu}|\hat{B}_1|,$$
\[
|\Delta \tilde{B}_i| \leq \frac{(C + 1)u}{1 - Cu}(|\tilde{B}_i| + |\hat{l}_{i-1}|), \quad 2 \leq i \leq n,
\]

\[
|\Delta \tilde{G}_i| \leq \frac{(C + 2)u + u^2}{1 - Cu}|\tilde{G}_i|, \quad 1 \leq i \leq n,
\]

\[
|\Delta \hat{b}| \leq u|\hat{b}|, \quad |\Delta \hat{g}| \leq (2u + u^2)|\hat{g}|.
\]

Notice that Algorithm 2 is componentwise stable in the mixed forward-backward sense [20] if \(|\hat{l}_{i-1}| = O(|\tilde{B}_i|)|, \quad 2 \leq i \leq n.\) Unfortunately, this is not always the case. Therefore, we cannot assure mixed forward-backward stability.

In order to bound the forward errors produced by Algorithm 2 we study the sensitivity of the modified Christoffel transformation with respect to perturbations of the initial data, i.e., the parameters of the tridiagonal matrix \(J_\alpha(\tilde{B}, \tilde{G})\), and the free parameter \(s\). Notice that \(\alpha\) is being considered among the entries of \(J_\alpha(\tilde{B}, \tilde{G})\) since \(\tilde{B}_1 = \alpha\). We consider perturbations associated with the backward error found in Theorem 4. According to Theorem 4, we only consider small relative perturbations for the parameter \(s\), i.e., \(|\Delta s| \leq \epsilon|s|\). We measure the sensitivity of the problem by using the notion of componentwise relative condition number. In the following definition the variables \(\hat{l}_1, \hat{l}_2, \ldots, \hat{l}_{n-1}\) correspond to the subdiagonal entries of the \(\hat{L}_s\) factor in the LU factorization with bidiagonal factors and parameter \(s\) of \(J_\alpha(\tilde{B}, \tilde{G}) - \alpha I\).

**Definition 5** Let \(J_\alpha(b, g)\) be the modified Christoffel transform of order \(n\) with parameter \(s\) and shift \(\alpha\) of the \((n + 1) \times (n + 1)\) tridiagonal matrix \(J_\alpha(\tilde{B}, \tilde{G})\). Let \(J_{\alpha + \Delta \alpha}(b + \Delta b, g + \Delta g)\) be the modified Christoffel transform of order \(n\) with parameter \(s + \Delta s\) and shift \(\alpha + \Delta \alpha\) of the \((n + 1) \times (n + 1)\) tridiagonal matrix \(J_{\alpha + \Delta \alpha}(\tilde{B} + \Delta \tilde{B}, \tilde{G} + \Delta \tilde{G})\). Let us define

\[
DB = \max \left\{ \max_{2 \leq i \leq n} \left\{ \frac{|\Delta \tilde{B}_i|}{|\tilde{B}_i| + |\hat{l}_{i-1}|} \right\}, \max_{2 \leq i \leq n} \left\{ \frac{|\Delta \tilde{G}_i|}{|\tilde{G}_i|} \right\}, \frac{|\Delta s|}{|s|}, \frac{|\Delta \alpha|}{|\alpha|} \right\},
\]

where the quotients \(\frac{|\Delta \tilde{B}_i|}{|\tilde{B}_i| + |\hat{l}_{i-1}|}\), \(\frac{|\Delta \tilde{B}_i|}{|\tilde{B}_i|}\), \(\frac{|\Delta \tilde{G}_i|}{|\tilde{G}_i|}\), or \(\frac{|\Delta \alpha|}{|\alpha|}\) have to be understood as zero if the denominators are equal to zero. Then the relative componentwise condition number of the modified Christoffel transformation with parameter \(s\) and shift \(\alpha\) with respect to perturbations associated to the backward errors in Theorem 4 is defined as

\[
\text{cond}_B(J_\alpha(\tilde{B}, \tilde{G}), s) = \lim_{\epsilon \to 0} \sup_{0 \leq DB \leq \epsilon} \frac{\max \left\{ \max_{1 \leq i \leq n} \left\{ \frac{|\Delta b_i|}{|b_i|} \right\}, \max_{1 \leq i \leq (n-1)} \left\{ \frac{|\Delta g_i|}{|g_i|} \right\} \right\}}{DB}.
\]

13
Next we give a sharp upper bound on the forward errors produced by Algorithm 2 in terms of the condition number \( \text{cond}(J_a(\tilde{B}, \tilde{G}), s) \). Moreover, notice that this bound can be computed in \( O(n) \) operations.

**Theorem 6** Let \( J_s(b, g) \) and \( J_s(\tilde{b}, \tilde{g}) \) be, respectively, the exact and the computed modified Christoffel transform with parameter \( s \) and shift \( \alpha \) of \( J_a(\tilde{B}, \tilde{G}) \) obtained from Algorithm 2, then

\[
\max_k \left\{ \frac{b_k - \tilde{b}_k}{b_k}, \left| \frac{g_k - \tilde{g}_k}{g_k} \right| \right\} \leq (C + 2)u \left( 1 + \text{cond}(J_a(\tilde{B}, \tilde{G}), s) \right) + O(u^2),
\]

where the left hand side of the previous inequality is a shorthand expression for (10) and

\[
\text{cond}(J_a(\tilde{B}, \tilde{G}), s) = \max\{ \max_{1 \leq k \leq n} \{ \text{cond}_s(b_k) \}, \max_{1 \leq k \leq n-1} \{ \text{cond}_s(g_k) \} \},
\]

where

\[
\text{cond}_s(b_1) = \left| \frac{s}{b_1} \right| + \left| \frac{\alpha}{b_1} \right|,
\]

\[
\text{cond}_s(b_k) = \frac{\tilde{l}_k}{b_k} + \frac{\tilde{l}_k - \tilde{t}_k + \alpha}{b_k} \left( \left| \frac{\tilde{B}_k}{\tilde{t}_k - \alpha} \right| + \left| \frac{\tilde{t}_k - 1}{\tilde{t}_k - \alpha} \right| (1 + \text{cond}_{BGs}(\tilde{l}_{k-1})) \right)
+ \left| \frac{\alpha}{\tilde{t}_k - \alpha} \right| \left( \frac{\tilde{l}_k}{b_k} + \left( \frac{\tilde{l}_k - \tilde{t}_k + \alpha}{b_k} \right) \frac{\partial \tilde{l}_{k-1}}{\partial \alpha} \right), \quad \text{for } 2 \leq k \leq n,
\]

\[
\text{cond}_s(g_k) = \frac{\tilde{l}_k}{\tilde{t}_{k+1} - \alpha} + \frac{\tilde{B}_{k+1}}{\tilde{t}_{k+1} - \alpha} + \frac{\tilde{l}_k - 1}{\tilde{t}_{k+1} - \alpha} \text{cond}_{BGs}(\tilde{l}_k)
+ \left| \frac{\alpha}{g_k} \right| \left( -\tilde{l}_k + (\tilde{t}_{k+1} - \alpha - \tilde{l}_k) \frac{\partial \tilde{l}_k}{\partial \alpha} \right), \quad \text{for } 1 \leq k \leq n - 1,
\]

\[
\text{cond}_{BGs}(\tilde{l}_k) := 1 + \frac{\tilde{B}_k}{\tilde{t}_k - \alpha} + \frac{\tilde{l}_k - 1}{\tilde{t}_k - \alpha} (1 + \text{cond}_{BGs}(\tilde{l}_{k-1})),
\]

\[
\text{cond}_{BGs}(\tilde{l}_1) = 1.
\]

If the analysis done for Algorithm 2 is repeated for Algorithm 1, a counterpart version of Theorem 4 can be obtained. Similarly, the corresponding condition number \( \text{cond}(J_a(\tilde{B}, \tilde{G}), s) \) can be found. It is easy to prove that

\[
\text{cond}(J_a(\tilde{B}, \tilde{G}), s) \leq \text{cond}(J_a(\tilde{B}, \tilde{G}), s).
\]

This result together with the numerical experiments in Table 1 show that Algorithm 2 is more accurate than Algorithm 1.
4.1 Stability and accuracy for large shifts

In the next lemma we show that Algorithm 2 becomes stable when $\alpha$ is large enough and $|s| = O(|\tilde{B}_2|)$. As we mentioned in [3], this is proven for the exact values of $|\tilde{l}_k|$ and not for the computed values $|\hat{\tilde{l}}_k|$, which means that we are only proving stability up to $O(u^2)$ terms.

**Lemma 7** Let $\tilde{l}_k$, $1 \leq k \leq n$, be the variables appearing in Algorithm 2, i.e., the subdiagonal elements in the $\tilde{L}_s$ factor of the LU factorization with bidiagonal factors and parameter $s$ of $J_\alpha(\tilde{B}, \tilde{G}) - \alpha I$. Then

$$\lim_{|\alpha| \to \infty} |\tilde{l}_k| = 0, \quad k \geq 2.$$ 

Since $\tilde{l}_1 = s$, Algorithm 2 is stable for $|\alpha|$ large enough if $|s| = O(|\tilde{B}_2|)$.

The algorithm also becomes accurate for large enough shifts, i.e., it produces outputs with small componentwise forward errors. In the next theorem we show that the condition number for the modified Christoffel transformation with parameter $s$ and shift $\alpha$ is bounded by 3 for large enough values of the shift $\alpha$, and therefore, we obtain accuracy.

**Theorem 8** Let $\text{cond}_B(J_\alpha(\tilde{B}, \tilde{G}), s)$ be the condition number for the modified Christoffel transformation with parameter $s$ and shift $\alpha$ introduced in Definition 5. Then

$$\lim_{|\alpha| \to \infty} \text{cond}_B(J_\alpha(\tilde{B}, \tilde{G}), s) = 3.$$ 

This implies that Algorithm 2 is accurate for $|\alpha|$ large enough.

It is worth to mention that a variation in the value of the parameter $s$ does not affect significantly the accuracy of the Algorithm 2 as the results in Table 2 and Table 3 show. We repeat some of the experiments done in Section 3. This time we keep fixed the shift $\alpha$ and let the parameter $s$ vary. We repeat the experiment for different values of $\alpha$ in each case. We are keeping the notation introduced in Section 3. Table 1 shows the results for the $3 \times 3$ monic Jacobi matrix with $B = [10^{-6}, -3 \cdot 10^{-6}, -1]$ and $G = [2 \cdot 10^{-6}, 10^{-6}]$. Table 3 shows the results for Jacobi polynomials.

Finally, similar arguments to those used in [3] show that Algorithm 2 is forward stable, that is, it produces answers with forward errors of similar magnitude to those produced by a backward stable algorithm. This implies that the forward error bound we have obtained for Algorithm 2 is the best one can expect.
<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>$s$ = 0.5</th>
<th>$s$ = 50</th>
<th>$s$ = 100</th>
<th>$s$ = 1000</th>
<th>$s$ = 10$^4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>max($vm_1$)</td>
<td>$2 \cdot 10^{-16}$</td>
<td>$1.8 \cdot 10^{-16}$</td>
<td>0</td>
<td>$1.7 \cdot 10^{-16}$</td>
<td>0</td>
</tr>
<tr>
<td>max($vm_2$)</td>
<td>$2 \cdot 10^{-16}$</td>
<td>$1.8 \cdot 10^{-16}$</td>
<td>0</td>
<td>$1.7 \cdot 10^{-16}$</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\alpha$ = -1</th>
<th>$s$ = 0.5</th>
<th>$s$ = 50</th>
<th>$s$ = 100</th>
<th>$s$ = 1000</th>
<th>$s$ = 10$^4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>max($vm_1$)</td>
<td>$4.3 \cdot 10^{-12}$</td>
<td>$2 \cdot 10^{-12}$</td>
<td>$2.1 \cdot 10^{-11}$</td>
<td>$6.8 \cdot 10^{-12}$</td>
<td>$3.3 \cdot 10^{-11}$</td>
</tr>
<tr>
<td>max($vm_2$)</td>
<td>$2.2 \cdot 10^{-16}$</td>
<td>$2 \cdot 10^{-16}$</td>
<td>0</td>
<td>$2 \cdot 10^{-16}$</td>
<td>$1.3 \cdot 10^{-16}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\alpha$ = -2</th>
<th>$s$ = 0.5</th>
<th>$s$ = 50</th>
<th>$s$ = 100</th>
<th>$s$ = 1000</th>
<th>$s$ = 10$^4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>max($vm_1$)</td>
<td>$2.8 \cdot 10^{-11}$</td>
<td>$6.8 \cdot 10^{-11}$</td>
<td>$6.4 \cdot 10^{-11}$</td>
<td>$1.9 \cdot 10^{-11}$</td>
<td>$1.4 \cdot 10^{-11}$</td>
</tr>
<tr>
<td>max($vm_2$)</td>
<td>0</td>
<td>$1.9 \cdot 10^{-16}$</td>
<td>$1.6 \cdot 10^{-16}$</td>
<td>$1.7 \cdot 10^{-16}$</td>
<td>$1.7 \cdot 10^{-16}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\alpha$ = 1</th>
<th>$s$ = 0.5</th>
<th>$s$ = 50</th>
<th>$s$ = 100</th>
<th>$s$ = 1000</th>
<th>$s$ = 10$^4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>max($vm_1$)</td>
<td>$1.1 \cdot 10^{-11}$</td>
<td>$4.4 \cdot 10^{-11}$</td>
<td>$2.1 \cdot 10^{-11}$</td>
<td>$1.9 \cdot 10^{-11}$</td>
<td>$1.3 \cdot 10^{-11}$</td>
</tr>
<tr>
<td>max($vm_2$)</td>
<td>$1.6 \cdot 10^{-16}$</td>
<td>$2 \cdot 10^{-16}$</td>
<td>$2.1 \cdot 10^{-16}$</td>
<td>0</td>
<td>$1.3 \cdot 10^{-16}$</td>
</tr>
</tbody>
</table>

Table 2  Results for the 3-by-3 matrix

<table>
<thead>
<tr>
<th>$\alpha$ = 0</th>
<th>$s$ = 0.5</th>
<th>$s$ = 50</th>
<th>$s$ = 100</th>
<th>$s$ = 1000</th>
<th>$s$ = 10$^4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>max($vm_1$)</td>
<td>$6.8 \cdot 10^{-14}$</td>
<td>$3.8 \cdot 10^{-13}$</td>
<td>$2 \cdot 10^{-13}$</td>
<td>$4.4 \cdot 10^{-14}$</td>
<td>$5.9 \cdot 10^{-13}$</td>
</tr>
<tr>
<td>max($vm_2$)</td>
<td>$6.8 \cdot 10^{-14}$</td>
<td>$3.8 \cdot 10^{-13}$</td>
<td>$2 \cdot 10^{-13}$</td>
<td>$4.4 \cdot 10^{-14}$</td>
<td>$5.9 \cdot 10^{-13}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\alpha$ = 10</th>
<th>$s$ = 0.5</th>
<th>$s$ = 50</th>
<th>$s$ = 100</th>
<th>$s$ = 1000</th>
<th>$s$ = 10$^4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>max($vm_1$)</td>
<td>$3.3 \cdot 10^{-11}$</td>
<td>$3.3 \cdot 10^{-11}$</td>
<td>$3.3 \cdot 10^{-11}$</td>
<td>$3.3 \cdot 10^{-11}$</td>
<td>$3.3 \cdot 10^{-11}$</td>
</tr>
<tr>
<td>max($vm_2$)</td>
<td>$7.8 \cdot 10^{-14}$</td>
<td>$7.8 \cdot 10^{-14}$</td>
<td>$7.8 \cdot 10^{-14}$</td>
<td>$7.8 \cdot 10^{-14}$</td>
<td>$7.8 \cdot 10^{-14}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\alpha$ = 100</th>
<th>$s$ = 0.5</th>
<th>$s$ = 50</th>
<th>$s$ = 100</th>
<th>$s$ = 1000</th>
<th>$s$ = 10$^4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>max($vm_1$)</td>
<td>$2 \cdot 10^{-10}$</td>
<td>$2.1 \cdot 10^{-10}$</td>
<td>$2.1 \cdot 10^{-10}$</td>
<td>$2 \cdot 10^{-10}$</td>
<td>$2 \cdot 10^{-10}$</td>
</tr>
<tr>
<td>max($vm_2$)</td>
<td>$1.1 \cdot 10^{-14}$</td>
<td>$1.1 \cdot 10^{-14}$</td>
<td>$1.1 \cdot 10^{-14}$</td>
<td>$1.1 \cdot 10^{-14}$</td>
<td>$1.1 \cdot 10^{-14}$</td>
</tr>
</tbody>
</table>

Table 3  Results for Jacobi polynomials

References


