Numerical Analysis Project
Manuscript NA-89-09

August 1989

On generating polynomials which are orthogonal over several intervals

by

Gene H. Golub
Bernd Fischer

Numerical Analysis Project
Computer Science Department
Stanford University
Stanford, California 94305

On generating polynomials which are orthogonal over several intervals

Bernd Fischer* and Gene H. Golub

Department of Computer Science
Stanford University
Stanford, CA 94305

Abstract

We consider the problem of generating the recursion coefficients of *orthogonal* polynomials for a *given* weight function. The weight function *is* assumed *to* he the weighted *sum of* weight functions, *each* supported on its own interval. Some of these intervals may *coincide,* overlap or are contiguous. We discuss three *algorithms*. Two of them are based on modified moments, whereas the other is based on an explicit expression for the desired coefficients. Several examples, *illustrating the numerical performance* of the various methods, are presented.

1. Introduction

Let $[l_j, u_j]$, j = $1, 2, \ldots$, N, $l_1 \leq l_2 \ldots \leq l_N$, be N, not necessarily disjoint, real intervals. Furthermore, let ω_j be a nonnegative weight function on $[l_j, u_j]$, j = $1, 2, \ldots$, N. With every ω_j there is associated a system of orthogonal polynomials $\{p_k^{(j)}\}$, in which $p_k^{(j)}$ has exact degree k and

$$\int_{l_j}^{u_j} p_k^{(j)}(x) p_m^{(j)}(x) \omega_j(x) dx \begin{cases} > 0 & \text{if } k = m \\ = 0 & \text{if } k \neq m \end{cases}. \tag{1.1}$$

They satisfy, as is well-known, a three-term relation

$$\begin{aligned} x p_k^{(j)}(x) &= b_k^{(j)} p_{k+1}^{(j)}(x) + a_k^{(j)} p_k^{(j)}(x) + c_k^{(j)} p_{k-1}^{(j)}(x), \quad k = 0, 1, \ldots \\ p_{-1}(x)^{(j)} &\equiv 0, \quad p_0^{(j)}(x) \equiv 1, \end{aligned} \tag{1.2}$$

The work of the first author was supported by the German Research Association (DFG). The second author was in part supported by the US Army under grant DAAl03-87-K-0095.

* On leave from the Institut fir Angewandte Mathematik, Universität Hamburg, D – 2000 Hamburg 13, F.R.G.

where $a_k^{(j)}, c_k^{(j)}$ are real numbers and $b_k^{(j)} > 0$. We set

$$l := l_1 \quad \text{and} \quad u := \max_{1 \le j \le N} u_j$$

and consider the nonnegative weight function $\omega(x)$ defined in the interval $[l, u]$ by

$$w(x) := \sum_{j=1}^{N} \varepsilon_j \, \mathcal{X}_{[l_j, u_j]}(x) \; \omega_j(x) \; (\ge 0) \tag{1.3}$$

where

$$\varepsilon_j \in \{-1, 1\} \quad \text{and} \quad \mathcal{X}_{[l_j,u_j]}(x) := \begin{cases} 1 & \text{if } 2 \in [l_j, Uj] \\ 0 & \text{if } 2 \notin [l_j, Uj] \end{cases} .$$

The inner product associated with $\omega(x)$ will be denoted by $<,>$, i.e.

$$\begin{aligned} < f, g > &:= \int_l^u f(x) g(x) \omega(x) dx \\ &= \sum_{j=1}^{N} \varepsilon_j \int_{l_j}^{u_j} f(x) g(x) \omega_j(x) dx. \end{aligned} \tag{1.4}$$

Clearly, there exist a set of polynomials $\{\psi_k\}$ that is orthogonal with respect to this inner product. In this paper we investigate the problem of numerically generating the recurrence coefficients in the relation

$$\begin{aligned} x\psi_k(x) &= \beta_k \psi_{k+1}(x) + \alpha_k \psi_k(x) + \gamma_k \psi_{k-1}(x), \quad k = 0, 1, \ldots \\ \psi_{-1}(x) &\equiv 0, \quad \psi_0(x) \equiv 1, \end{aligned} \tag{1.5}$$

under the asumption that the coefficients $b_k^{(j)}, a_k^{(j)}, c_k^{(j)}$, $j = 1, 2, \ldots, N$, for whatever value of k is required, and the moments

$$\nu_0^{(j)} := \int_{l_j}^{u_j} \omega_j(x) dx, \quad j = 1, 2, \ldots, N, \tag{1.6}$$

are given.

Problems of this type arise for example in connection with the numerical solution of large systems of linear equations (see e.g., Saad [17]), in theoretical chemistry (see e.g., Wheeler [21]), and of course in the determination of Gaussian quadrature formulae.

We will discuss two classical approaches for generating the recursion coefficients. The first one is based on the fact that the desired coefficients are given by

$$
\begin{aligned}
\alpha_k &= \frac{< x\psi_k, \psi_k >}{< \psi_k, \psi_k >}, \quad k = 0, 1, \ldots \\
\gamma_k &= \beta_{k-1} \frac{< \psi_k, \psi_k >}{< \psi_{k-1}, \psi_{k-1} >}, \quad k = 1, 2, \ldots
\end{aligned} \tag{1.7}
$$

where the β_k (> 0) are arbitrary. The resulting procedure, alternating recursively between (1.7) and (1.5), is called Stieltjes *procedure* [**Stieltjes**] (for historical remarks see Gautschi [3,4]). The Stieltjes procedure **will** be discussed in **Section** 3.1.

Our second approach involves the so - **called** modified *moments*

$$
\nu_k := < q_k, 1 > = \int_l^u q_k(x)\omega(x)dx, \quad k = 0, 1, \ldots \tag{1.8}
$$

where $\{q_k\}$ is a given suitable set of polynomials whith degree $q_k = k$. Two algorithms using the modified moments will be described in Section 3.2. They are generalizations of one derived by Chebyshev in the case of ordinary moments, i.e., $q_k(x) = x^k$, and are therefore called ***modified Chebyshev algorithm*** **[modCheb](for** historical remarks see Gautschi [3,4]). Both algorithms, basically, obtain the desired recursion coefficients in terms of the Cholesky factor R in the Cholesky decomposition (see e.g., Golub, Van Loan [12, Ch. 4.2.3])

$$
M = RR^T \tag{1.9}
$$

of the associated Gram matrix $M = [< q_i, q_j >]$. One method [**modCheb-Cholesky**] computes first the Cholesky decomposition (1.9) and then the coefficients $\beta_k, \alpha_k, \gamma_k$, whereas the other scheme [**modChebUpdate**] alternates recursively between updating R and computing $\beta_k, \alpha_k, \gamma_k$. We conclude Section 3 with a simple proof of a determinantal expression, in terms of the Gram matrix M, for the desired coefficients.

All three algorithms have in common, the need to compute the inner product $<, >$ fast and accurately. We will discuss a method for this purpose in Section 2. In Section 3.2 we will see, that this method leads in particular to an attractive algorithm for computing the modified moments (1.8). **Finally, a number of examples,** illustrating the numerical performance of the various methods, are given in Section 4.

2. Evaluation of the inner product

The success of the Stieltjes procedure, as well as the modified Chebyshev algorithms, depends in part on the ability to compute the inner product $<, >$ fast and accurately. In this section we show how to evaluate $< p, 1 >$, say for a polynomial of degree $\leq 2n$, under the given circumstances.

The computation of $< p, 1 >$ can be performed effectively using the Gauss quadrature rule corresponding to the weight function ω_j. In view of (1.4), we have to generate the rules

$$\int_{l_j}^{u_j} p(x)\omega_j(x)dx = \nu_0^{(j)} \sum_{i=0}^{n} (v_{i1}^{(j)})^2 p(\lambda_i^{(j)}),\ j = 1, 2, \ldots, N. \tag{2.1}$$

We first recall some basic facts on Gauss quadrature. We associate with the weight-function ω_j the tridiagonal matrix (compare 1.2)

$$T_n^{(j)} := \begin{pmatrix} a_0^{(j)} & b_0^{(j)} & & & \\ c_1^{(j)} & a_1^{(j)} & b_1^{(j)} & & \\ & \ddots & \ddots & \ddots & \\ & & c_{n-1}^{(j)} & a_{n-1}^{(j)} & b_{n-1}^{(j)} \\ & & & c_n^{(j)} & a_n^{(j)} \end{pmatrix}. \tag{2.2}$$

Note, that $p_{n+1}^{(j)}$ is, up to the factor $\prod_{i=0}^{n}(-b_i)$, the characteristic polynomial of $T_n^{(j)}$. Hence, as is well known, the nodes $\lambda_i^{(j)}$ of (2.1) are the eigenvalues of $T_n^{(j)}$. If $T_n^{(j)}$ is not symmetric, it can be symmetrized by a diagonal similarity transformation $D_n^{(j)} := \operatorname{diag}(d_0^{(j)}, d_1^{(j)}, \ldots, d_n^{(j)})$, where the diagonal elements $d_k^{(j)}$ are given by

$$d_{k+1}^{(j)} = d_k^{(j)} \sqrt{c_{k+1}^{(j)}/b_k^{(j)}}, \quad k = 0, 1, \ldots, n-1.$$

Here, $d_0^{(j)} (\neq 0)$ is arbitrary. Thus

$$J_n^{(j)} := (D_n^{(j)})^{-1} T_n^{(j)} D_n^{(j)} = \begin{pmatrix} a_0^{(j)} & \hat{b}_0^{(j)} & & & \\ \hat{b}_0^{(j)} & a_1^{(j)} & \hat{b}_1^{(j)} & & \\ & \ddots & \ddots & \ddots & \\ & & \hat{b}_{n-2}^{(j)} & a_{n-1}^{(j)} & \hat{b}_{n-1}^{(j)} \\ & & & \hat{b}_{n-1}^{(j)} & a_n^{(j)} \end{pmatrix}, \tag{2.3}$$

where $\hat{b}_k^{(j)} = \sqrt{b_k^{(j)} c_{k+1}^{(j)}}$, $k = 0, 1, \ldots, n-1$. We refer to $J_n^{(j)}$ as the (nth) *Jacobi matrix* of ω_j. The polynomials $\hat{p}_k^{(j)}$ corresponding to $J_n^{(j)}$ are related to $p_k^{(j)}$ by $\hat{p}_k^{(j)} = p_k^{(j)}/d_k^{(j)}$. Hence, if we choose the free parameter $d_0^{(j)} = \sqrt{\nu_0^{(j)}}$ the resulting polynomials $\hat{p}_k^j$ are orthonormal with respect to ω_j. However, it is well known (see e.g., Wilf[22, Ch. 2]), that the weights $v_{i1}^{(j)}$ in (2.1) are the first component of the normalized eigenvector $v_i^{(j)}$ of $J_n^{(j)}$ corresponding to $\lambda_i^{(j)}$

$$J_n^{(j)} v_i^{(j)} = \lambda_i^{(j)} v_i^{(j)},\ (v_i^{(j)})^T v_i^{(j)} = 1,\ i = 0, 1, \ldots, n. \tag{2.4}$$

In principle, we could compute $\lambda_i^{(j)}$, $v_i^{(j)}$ using one of the standard methods for calculating eigenvalues **and** eigenvectors (see e.g., Golub, **Welsch [11]**). Fortunately, we do not need to know them explicitly. Since $J_n^{(j)}$ is hermitian, there exist a unitary matrix $U_n^{(j)}$ with

$$(U_n^{(j)})^{-1} J_n^{(j)} U_n^{(j)} = (U_n^{(j)})^T J_n^{(j)} U_n^{(j)} = \mathrm{diag}\left(\lambda_0^{(j)}, \lambda_1^{(j)}, \ldots, \lambda_n^{j}\right) (=: \Sigma_n^{(j)}), \qquad (2.5)$$

where each column $v_i^{(j)}$ of $U_n^{(j)}$ is a normalized eigenvector of $J_n^{(j)}$. Therefore, we have by (2.5), (2.1) and (1.4)

$$\boxed{\begin{aligned} \sum_{j=1}^{N} \varepsilon_j \nu_0^{(j)} e_1^T p(J_n^{(j)}) e_1 &= \sum_{j=1}^{N} \varepsilon_j \nu_0^{(j)} e_1^T (U_n^{(j)})^T p(\Sigma_n^{(j)}) U_n^{(j)} e_1 \\ &= \sum_{j=1}^{N} \varepsilon_j \nu_0^{(j)} \sum_{i=0}^{n} (v_{i1}^{(j)})^2 p(\lambda_i^{(j)}) \\ &= \sum_{j=1}^{N} \varepsilon_j \int_{l_j}^{u_j} p(x) \omega_j(x) dx \\ &= < p, 1 >, \end{aligned} \qquad (2.6)}$$

where $e_1^T = (1, 0, \ldots, 0)$ denotes the first unit vector. The "method" (2.6) will be frequently used in the following algorithms. It is not surprising, as we will see in the next sections, that the calculation of $< p, 1 >$ is even more effective, if p itself **fulfills** a certain recurrence relation.

3. Algorithms

In the following section we present a detailed description of the Stieltjes procedure and the modified Chebyshev algorithms.

The procedures compute a system of orthogonal polynomials $\{\psi_k\}_{k=0}^n$ for the given nonnegative weight function (compare 1.3)

$$w(x) := \sum_{j=1}^{N} \varepsilon_j \mathcal{X}_{[l_j, u_j]}(x)\, \omega_j(x). \qquad (3.1)$$

More precisely, the algorithms determine the coefficients in the three-term recurrence relation

$$\begin{aligned} x\psi_k(x) &= \beta_k \psi_{k+1}(x) + \alpha_k \psi_k(x) + \gamma_k \psi_{k-1}(x), \quad k = 0, 1, \ldots, n-1, \\ &\psi_{-1}(x) \equiv 0, \quad \psi_0(x) \equiv 1. \end{aligned} \qquad (3.2)$$

We remark that the system $\{\psi_k\}_{k=0}^n$ has all of the properties of polynomials orthogonal on one interval, provided we consider ψ_k orthogonal on $[l,u]$ rather than on $\bigcup_{j=1}^N [l_j, u_j]$. For example, the polynomials ψ_k have all roots in $[l,u]$, but not necessarily in $\bigcup_{j=1}^N [l_j, u_j]$ (see Example 4.5).

However, we have not yet specified a condition that will uniquely determine the orthogonal polynomials $\{\psi_k\}_{k=0}^n$. In order to make the computational effort of the various methods comparable, we will devise algorithms that generate the system of ***orthonormal*** polynomials $\{\hat{\psi}_k\}_{k=0}^n$ w.r.t. w. Here we have by **(1.4)** and (1.6)

$$\begin{aligned} x\hat{\psi}_k(x) &= \hat{\gamma}_{k+1}\hat{\psi}_{k+1}(x) + \hat{\alpha}_k\hat{\psi}_k(x) + \hat{\gamma}_k\hat{\psi}_{k-1}(x), \quad k = 0,1,\ldots,n-1, \\ \hat{\psi}_{-1}(x) &\equiv 0, \quad \hat{\psi}_0(x) \equiv \hat{\psi}_0 = \left(\sum_{j=1}^N \varepsilon_j \nu_0^{(j)}\right)^{-1/2}. \end{aligned} \tag{3.3}$$

Observe that the symmetric relation in $\hat{\gamma}_k$ forces

$$<\hat{\psi}_n, \hat{\psi}_n> = <\hat{\psi}_{n-1}, \hat{\psi}_{n-1}> = \cdots = <\hat{\psi}_0, \hat{\psi}_0> = 1, \tag{3.4}$$

and that $\hat{\psi}_k$ is related to ψ_k by

$$\hat{\psi}_k(x) = <\psi_k, \psi_k>^{-1/2} \psi_k(x). \tag{3.5}$$

3.1 Stieltjes procedure

An explicit expression for the coefficients of $\{\hat{\psi}_k\}_{k=0}^n$ is easily deduced from **(1.7)** and (3.5). For convenience we set $\beta_k = 1$, i.e., ψ_k is a monic polynomial, and obtain

$$\begin{aligned} \hat{\alpha}_k &= \alpha_k = \frac{<x\psi_k, \psi_k>}{<\psi_k, \psi_k>}, \quad k = 0,1,\ldots, n-1, \\ \hat{\gamma}_k &= \sqrt{\gamma_k} = \left(\frac{<\psi_k, \psi_k>}{<\psi_{k-1}, \psi_{k-1}>}\right)^{1/2} \quad k = 1,2,\ldots, n. \end{aligned} \tag{3.6}$$

In order to evaluate the inner products in **(3.6)** we recursively combine **(3.2)** and **(2.6)**. Therefore, let

$$\begin{aligned} z_{k+1}^{(j)} &:= \psi_{k+1}(J_n^{(j)})e_1 \\ &= (J_n^{(j)} - \alpha_k I)\psi_k(J_n^{(j)})e_1 - \gamma_k\psi_{k-1}(J_n^{(j)})e_1 \\ &= (J_n^{(j)} - \alpha_k I)z_k^{(j)} - \gamma_k z_{k-1}^{(j)}. \end{aligned}$$

Then

$$\begin{aligned} <\psi_{k+1}, \psi_{k+1}> &= \sum_{j=1}^N \varepsilon_j \nu_0^{(j)} e_1^T \psi_{k+1}(J_n^{(j)})^T \psi_{k+1}(J_n^{(j)}) e_1 \\ &= \sum_{j=1}^N \varepsilon_j \nu_0^{(j)} (z_{k+1}^{(j)})^T z_{k+1}^{(j)}. \end{aligned}$$

Altogether, we arrive at:

Stieltjes. Given a set of weight **functions** ω_j and the associated Jacobi matrices $J_n^{(j)}$ by *(2.3)* and the moments $\nu_0^{(j)}$ by (1.6) , j = **1,2. . .** , N, this algorithm computes the recurrence coefficients of the **polynomials** $\hat{\psi}_k$, $k = 1, 2, \ldots$, n, ***orthonormal*** with respect to ω.

Initialize. Set $z_0^{(j)} := e_1$, j = **1,2,. . .** , N.

- compute $\hat{\alpha}_0 (\rightarrow \alpha_0)$ by (3.6) and *(2.6)*

$$\alpha_0 = \frac{\sum_{j=1}^N \varepsilon_j \nu_0^{(j)} (z_0^{(j)})^T J_n^{(j)} z_0^{(j)}}{\sum_{j=1}^N \varepsilon_j \nu_0^{(j)} (z_0^{(j)})^T z_0^{(j)}} = \frac{\sum_{j=1}^N \varepsilon_j \nu_0^{(j)} a_0^{(j)}}{\sum_{j=1}^N \varepsilon_j \nu_0^{(j)}}.$$

- compute $z_1^{(j)} := \psi_1(J_n^{(j)}) e_1$ by *(3.2)* with $\beta_0 = 1$

$$z_1^{(j)} = \left(J_n^{(j)} - \alpha_0 I\right) z_0^{(j)}, \; j = 1, 2, \ldots, \text{N}.$$

Iterate. ***For*** $k = 1, 2, \ldots,$ ***n-l do***

- ♍□○□◆◆♏ $\hat{\alpha}_k (\rightarrow \alpha_k)$ and γ_k by *(1. 7)*, (3.6) and (2.6)

$$\alpha_k = \frac{\sum_{j=1}^N \varepsilon_j \nu_0^{(j)} (z_k^{(j)})^T J_n^{(j)} z_k^{(j)}}{\sum_{j=1}^N \varepsilon_j \nu_0^{(j)} (z_k^{(j)})^T z_k^{(j)}}, \quad \gamma_k = \frac{\sum_{j=1}^N \varepsilon_j \nu_0^{(j)} (z_k^{(j)})^T z_k^{(j)}}{\sum_{j=1}^N \varepsilon_j \nu_0^{(j)} (z_{k-1}^{(j)})^T z_{k-1}^{(j)}}.$$

- ♍□○□◆◆♏ $z_{k+1}^{(j)} := \psi_{k+1}(J_n^{(j)}) e_1$ by (3 *2)* with $\beta_k = 1$

$$z_{k+1}^{(j)} = (J_n^{(j)} - \alpha_k I) z_k^{(j)} - \gamma_k z_{k-1}^{(j)}, \quad j = 1, 2, \ldots, \text{N}.$$

- compute $\hat{\gamma}_k (\rightarrow \gamma_k)$ by (3.6)

$$\gamma_k = \sqrt{\gamma_k}.$$

if $k = n - 1$ ***then***

$$\gamma_n = \left(\frac{\sum_{j=1}^N \varepsilon_j \nu_0^{(j)} (z_n^{(j)})^T z_n^{(j)}}{\sum_{j=1}^N \varepsilon_j \nu_0^{(j)} (z_{n-1}^{(j)})^T z_{n-1}^{(j)}} \right)^{1/2}.$$

End.

Remarks. 1. The algorithm requires $NO((n + 1)^2)$ flops plus n square root computations. 2. The number of recursion coefficients that can be calculated is bounded by the dimension of the Jacobi-matrices. In order to compute more coefficients, one has to restart the computation of $z_k^{(j)}$ with appropiate **Jacobi**-matrices. 3. The last $n - k$ elements of $z_k^{(j)}$ are zero. This can be used for designing a more efficient algorithm.

3.2 Modified Chebyshev algorithm

In this section we present two algorithms involving the modified moments

$$\nu_k := \langle q_k, 1 \rangle = \int_l^u q_k(x)\omega(x)dx, \quad k = 0, 1, \dots, 2n. \tag{3.7}$$

Both algorithms differ **from** the corresponding algorithm for a single **interval,i.e.,** N = 1, **only** in the computation of ν_k. Therefore, if the ν_k are known **analytically,** the algorithms for a single and several intervals coincide, i.e., have the same complexity.

However, in general we have to compute the modified moments. Here, we arrive at an efficient algorithm, if we assume that the system of **polynomials** $\{q_k\}_{k=0}^{2n}$ as **well** satisfies a three-term recurrence relation

$$\begin{aligned} xq_k(x) &= b_k q_{k+1}(x) + a_k q_k(x) + c_k q_{k-1}(x), \quad k = 0, 1, \dots 2n-1, \\ q_{-1}(x) &\equiv 0, \quad q_0(x) \equiv 1. \end{aligned} \tag{3.8}$$

Once more using the method (2.6) we obtain:

Modmoment(q_n). Given a set of weight *functions* ω_j and the associated Jacobi matrices $J_n^{(j)}$ by (2.3) and *the* moments $\nu_0^{(j)}$ by *(1.6)*, $j = 1, 2, \dots, N$, and *the system of* polynomials $\{q_l\}_{l=0}^{2n}$ *by (3.8),* this *algorithm computes* the *modified* moments ν_k, $k = 0, 1, \dots, 2n$, of w relative to $\{q_l\}_{l=0}^{2n}$.

Initialize. *Set* $z_0^{(j)} := e_1$, $z_{-1}^{(j)} \equiv 0$, $j = 1, 2, \dots, N$, $c_0 := 0$.

- compute ν_0 by (2.6)

$$\nu_0 = \sum_{j=1}^{N} \varepsilon_j \nu_0^{(j)} e_1^T z_0^{(j)} = \sum_{j=1}^{N} \varepsilon_j \nu_0^{(j)}.$$

Iterate. *For* $k = 1, 2, \dots, 2n$ *do*

- compute $z_k^{(j)} := q_k(J_n^{(j)})e_1$ *by (3.8)*

$$z_k^{(j)} = \frac{1}{b_{k-1}} \left((J_n^{(j)} - a_{k-1}I) z_{k-1}^{(j)} - c_{k-1} z_{k-2}^{(j)} \right), \quad j = 1, 2, \dots, N.$$

- *compute* ν_k *by (2.6)*

$$\nu_k = \sum_{j=1}^{N} \varepsilon_j \nu_0^{(j)} e_1^T z_k^{(j)}.$$

End.

Remarks. **1.** The algorithm requires $N\mathcal{O}((n+1)^2)$ flops. **2.** The number of modified moments that can be calculated is bounded by the dimension of the Jacobi-matrices. In order to compute more modified moments, one has to restart the computation of $z_k^{(j)}$ with appropiate Jacobi matrices. **3.** It is easy to see, that the algorithm does not require symmetric Jacobi matrices $J_n^{(j)}$. Instead, one can also use $T_n^{(j)}$, given by **(2.2).** 4. The last $n-k$ elements of $z_k^{(j)}$ are zero. This can be used for designing a more efficient algorithm. **5.** In order to start the algorithm, one has to choose a set of polynomials $\{q_l\}_{l=0}^{2n}$. An obvious choice is $q_k \equiv p_k^{(i)}, i \in \{1,2,\dots,N\}$. Here **(3.7)** reduces to

$$\nu_k = \sum_{\substack{j=1\\ j\neq i}}^{N} \varepsilon_j \int_{l_j}^{u_j} p_k^{(i)}(x)\omega_j(x)dx.$$

However, we **only** recommend this choice for $[l_i, u_i] \approx [l, u]$. Otherwise $p_k^{(i)}$ would in general produce **extremly** large $|\nu_k|$, due to the fact that $p_k^{(i)}$ has all zeros in $[l_i, u_i]$.

We **now** give a short derivation of the three term relationship of $\hat{\psi}_k$ in terms of the Gram matrix associated with q_l and the inner product $<,>$ (compare Kent [14, Ch. 2]). The Fourier-expansion of q_l in terms of $\hat{\psi}_k$ reads (recall $<\hat{\psi}_k, \hat{\psi}_k> = 1$)

$$\begin{aligned} q_l(x) &= \sum_{k=0}^{l} r_{lk}\hat{\psi}_k(x) \\ &= \sum_{k=0}^{l} <q_l, \hat{\psi}_k> \hat{\psi}_k(x) \end{aligned} \quad , \ l = 0,1,\dots, n \tag{3.9}$$

or vice versa

$$\hat{\psi}_k(x) = \sum_{m=0}^{k} s_{km} q_m(x), \quad k = 0,1,\dots,n. \tag{3.10}$$

The above equations define the nonsingular and lower triangular matrices $R := [r_{lk}]_{l,k=0}^{n} = [<q_l, \hat{\psi}_k>]_{l,k=0}^{n}$ and $S := [s_{km}]_{k,m=0}^{n}$, with $R = S^{-1}$. Moreover, we deduce from *(3.9)* and **(3.10)**

$$\begin{aligned} r_{lk} &= <q_l, \hat{\psi}_k> = <q_l, \sum_{m=0}^{k} s_{km} q_m(x)> \\ &= \sum_{m=0}^{k} s_{km} <q_l, q_m>. \end{aligned} \tag{3.11}$$

Now, consider the associated Gram matrix $M = [<q_l, q_m>]_{l,m=0}^{n}$. The system of equations **(3.11)** is equivalent to

$$R^T = SM, \text{ or } M = RR^T, \text{ or } M^{-1} = S^T S. \tag{3.12}$$

Therefore, R is the Cholesky factor of M and S is the inverse Cholesky factor of M.

Substituting **(3.10)** into (3.3) (resp. (3.9) in (3.8)) and comparing the coefficients of q_{k+1} and q_k (resp. $\hat{\psi}_{k+1}$ and $\hat{\psi}_k$) we obtain

$$\begin{aligned} \hat{\gamma}_{k+1} &= b_k \frac{s_{kk}}{s_{k+1,k+1}} = b_k \frac{r_{k+1,k+1}}{r_{kk}} \\ \hat{\alpha}_k &= a_k + b_{k-1} \frac{s_{k,k-1}}{s_{kk}} - b_k \frac{s_{k+1,k}}{s_{k+1,k+1}}, \quad k = 0,1,\ldots, n-1. \\ &= a_k - b_{k-1} \frac{r_{k,k-1}}{r_{k-1,k-1}} + b_k \frac{r_{k+1,k}}{r_{kk}} \end{aligned} \tag{3.13}$$

Thus the desired coefficients $\hat{\gamma}_k, \hat{\alpha}_k$ can be obtained from **(3.13)** in view of **(3.12)** by an inverse Cholesky decomposition of M^{-1} (resp. Cholesky decomposition of M), where only the diagonal and subdiagonal elements of S (resp. R) are involved.

3.2.1 Fast Cholesky decomposition

The derivation in the last paragraph leads directly to the following basic algorithm (compare Gautschi [2, Ch. 4]):

- build up the Gram matrix M by applying the recursion (3.8)
- compute the Cholesky decomposition $M = RR^T$ (resp. $M^{-1} = S^T S$)
- compute $\hat{\gamma}_k, \hat{\alpha}_k$ by (3.13).

Since the Cholesky decomposition of a $(n + 1) \times (n + 1)$ matrix takes in general $\mathcal{O}((n + 1)^3)$ arithmetic operations, this algorithmn does not compare favorably with the Stieltjes procedure in terms of speed.

One way to overcome this bottleneck is a clever choice of the system of polynomials $\{q_k\}$ which defines the modified moments ν_k. Let

$$T_k(x) := \cos(k \arccos(x)) \tag{3.14}$$

denote the ***k-th Chebyshev polynomial of the first kind.*** It is well known, that

$$T_l(x)T_m(x) = \frac{1}{2}\left(T_{|l-m|} + T_{l+m}\right). \tag{3.15}$$

Hence, for the setting $q_k \equiv T_k$, the associated Gram matrix reduces to (compare Branders [1, Ch. 6.4])

$$\begin{aligned} M = [< T_l, T_m >] &= \frac{1}{2}[< T_{|l-m|} + T_{l+m}, 1 >] \\ &= \frac{1}{2}[\nu_{|l-m|} + \nu_{l+m}] \\ &= \frac{1}{2}(\mathcal{T} + \mathcal{H}), \end{aligned} \tag{3.16}$$

where $\mathcal{T}$ is Toeplitz and $\mathcal{H}$ is Hankel. This special structure of M **allows** the construction of a fast, i.e. $\mathcal{O}((n+1)^2)$ or less, algorithm for the Cholesky decomposition (see e.g., Gohberg, Kailath, Koltracht [10], Heinig, Jankowski, Rost [13], Lev-Ari, Kailath [15]).

The fast algorithm we used for our computations, is based on the following (general) observation. Let M be a symmetric and positive definite matrix , e.g., M is a Gram matrix. Notice that the Cholesky decomposition $M = RR^T$ is *nested*, i.e.,

$$M_k = R_k R_k^T, \quad k = 0, 1, \ldots, n, \tag{3.17}$$

where $M_k = [m_{ij}]_{i,j=0}^k$ (resp. $R_k = [r_{ij}]_{i,j=0}^k$) denotes the k-th leading principal submatrix of M (resp. R). Since the inversion of a lower triangular matrix is also nested, we have from (3.17), that the inverse Cholesky decomposition $M^{-1} = (R^{-1})^T R^{-1} = S^T S$ is "semi-nested", i.e.,

$$M_k^{-1} = S_k^T S_k. \tag{3.18}$$

Here $M_k^{-1} = [u_{i,j}^{(k)}]_{i,j=0}^k$ is the inverse of M_k and $S_k = [s_{i,j}]_{i,j=0}^k$ denotes the k-th leading principal submatrix of S. Assume we have already computed S_{k-1}, then we obtain S_k by appending one row $s_k := (s_{k0}, s_{k1}, \ldots, s_{kk})$ and one column $(0, \ldots, 0, s_{kk})^T$ to S_{k-1}. In view of (3.18), the new elements s_{kj} are uniquely determined by $s_{kj} s_{kk} = u_{jk}^{(k)}$, that is

$$s_{kj} = \frac{u_{jk}^{(k)}}{\sqrt{u_{kk}^{(k)}}}, \quad j = 0, 1, \ldots, k. \tag{3.19}$$

Hence, s_k^T is up to a factor the last column of M_k^{-1}. Therefore, we obtain the inverse Cholesky factor

$$S = \begin{pmatrix} \frac{u_{00}^{(0)}}{\sqrt{u_{00}^{(0)}}} & & & \\ \frac{u_{01}^{(1)}}{\sqrt{u_{11}^{(1)}}} & \frac{u_{11}^{(1)}}{\sqrt{u_{11}^{(1)}}} & & \\ \vdots & \vdots & \ddots & \\ \frac{u_{0n}^{(n)}}{\sqrt{u_{nn}^{(n)}}} & \frac{u_{1n}^{(n)}}{\sqrt{u_{nn}^{(n)}}} & \cdots & \frac{u_{nn}^{(n)}}{\sqrt{u_{nn}^{(n)}}} \end{pmatrix} \tag{3.20}$$

by solving the linear systems

$$M_k u_k^{(k)} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}, \quad k = 0, 1, \ldots, n, \tag{3.21}$$

where $u_k^{(k)} = (u_{0,k}^{(k)}, u_{1,k}^{(k)}, \ldots, u_{k,k}^{(k)})^T$.

Again, the solution of this $n+1$ linear systems requires in general $\mathcal{O}((n+1)^3)$ arithmetic operations. However, if M is Toeplitz + Hankel, there exists $\mathcal{O}((n+1)^2)$ algorithms for the solution of **(3.21).** They are based on the fact, that the solution of two adjoining sections M_{k-1} and M_k are recursively connected. For details we refer to Heinig, Jankowsky and Rost [†] [13, pp. 671-674].

Observe, that we only need to compute the first $2n+1$ modified moments, in order to built up the Gram matrix $M = \frac{1}{2}[\nu_{|l-m|} + \nu_{l+m}]$. Once we have computed the modified moments and the inverse Cholesky decomposition the desired coefficients are given by *(3.13):*

ModChebCholesky. Given a *set of* weight *functions* ω_j *and the* associated Jacobi mat rices $J_n^{(j)}$ *by* (2.3) and *the* moments $\nu_0^{(j)}$ *by (1.6)* , $j = 1, 2 \ldots, N$, *this algorithm computes the recurrence coefficients of the polynomials* $\hat{\psi}_k$, $k = 1, 2, \ldots, n$, *orthonormal* with *respect to w.*

Initialize/Set $b_{-1} = s_{0,-1} = 0$.

- *compute the modified* moments ν_l relative *to* T_l, $l = 0, 1 \ldots, 2n$, *by* **Modmoment**(T_l)
- *compute the inverse Cholesky* factor $S = [s_{ij}]_{i,j=0}^n$ *by (3.20)* and (3.21)

Iterate. *For* $k = 0, 1, \ldots, n-1$ *do*

- *compute* $\hat{\alpha}_k$, $\hat{\gamma}_{k+1}$ *by (3.13)*

$$\hat{\alpha}_k = a_k + b_{k-1}\frac{s_{k,k-1}}{s_{k,k}} - b_k\frac{s_{k+1,k}}{s_{k+1,k+1}}$$

$$\hat{\gamma}_{k+1} = b_k\frac{s_{kk}}{s_{k+1,k+1}}$$

End.

Remarks. **1.** The algorithm requires $N\mathcal{O}((n+1)^2)$ flops plus n square root computations. *2.* We only need the diagonal and subdiagonal elements of S for the computation of the recursion coefficients. However, the recursion formulae for the solutions of (3.21) involve (unfortunately) the whole vector $u^{(k)}$.

We conclude this section with a more theoretical result. Let $M = [m_{ij}]_{i,j=0}^n = [< q_i, q_j >]_{i,j=0}^n$ denote once more the Gram matrix associated with $\{q_l\}$ and let M_k be the k-th leading principal submatrix of M. Furthermore let $D_k := \det(M_k)$

---

† Equations (5.6) and (5.8) in [13] are misprinted: β_{2m} should read $\beta_{2m} = (f_2^{m+1})^T x^{m+1} - \cos\frac{m+1}{2}\pi$, $\beta_{2m} = \lambda_{m+1} / \alpha_m - \lambda_m/\alpha_{m-1}$, respectively.

designate the k-th principal minor of M, while

$$\hat{D}_k := \det \begin{pmatrix} m_{00} & m_{01} & \cdots & m_{0,k-2} & m_{0k} \\ m_{10} & m_{11} & \cdots & m_{1,k-2} & m_{1k} \\ \vdots & \vdots & & \vdots & \vdots \\ m_{k-1,0} & m_{k-1,1} & \cdots & m_{k-1,k-2} & m_{k-1,k} \end{pmatrix} \tag{3.22}$$

is obtained by deleting the last row and $(k-1)$-th column "of D_k". By applying Cramers rule to (3.21) we easily deduce from (3.19) and (3.13), with the setting $D_{-1} = 1$, $\hat{D}_0 = 0$,

$$\begin{aligned} \hat{\alpha}_k &= a_k - b_{k-1}\frac{\hat{D}_k}{D_{k-1}} + b_k\frac{\hat{D}_{k+1}}{D_k} \\ \hat{\gamma}_{k+1} &= b_k\frac{\sqrt{D_{k-1}D_{k+1}}}{D_k} \end{aligned} \qquad k = 0,1,\ldots, n-1. \tag{3.23}$$

For the special case of ordinary moments $q_k(x) \equiv x^k$, i.e., $b_k = 1$, $a_k = c_k = 0$, we recover the well known relationship

$$\begin{aligned} \hat{\alpha}_k &= \frac{\hat{D}_k}{D_{k-1}} + \frac{\hat{D}_{k+1}}{D_k} \\ \hat{\gamma}_{k+1} &= \frac{\sqrt{D_{k-1}D_{k+1}}}{D_k} \end{aligned} \qquad k = 0, 1, \ldots, n-1. \tag{3.24}$$

In other words, the computation of $\hat{\alpha}_k$, γ_{k+1}, using the equation (3.23), is nothing else than an (expensive) implementation of a modified Chebyshev algorithm.

However, since the condition number of M depends in part on the polynomial system $\{q_l\}$, a clever choice of this system will improve a test, based on (3.24), for the validation of Gaussian quadrature formulae, proposed by Gautschi [5,pp. 214-215].

3.2.2 Updating the mixed moment matrix

The next (fast) algorithm computes the desired coefficients in terms of the Cholesky factor R, which is essential a "mixed moment" matrix (compare 3.9)

$$R = [r_{lk}] = [< q_l, \hat{\psi}_k >]. \tag{3.25}$$

Instead of explicitly computing the Cholesky decomposition, we update R continually as the process unfolds (compare Gautschi [8, Ch. 5.4], Sack, Donovan [18], Wheeler [20]). The key equation is easily obtained from the two recurrence relations (3.3) and (3.8)

$$\begin{aligned} r_{lk} &= < q_l, \hat{\psi}_k > \\ &= \frac{1}{\hat{\gamma}_k}\left(< xq_l, \hat{\psi}_{k-1} > - \hat{\alpha}_{k-1}\, r_{l,k-1} - \hat{\gamma}_{k-1}\, r_{l,k-2}\right) \\ &= \frac{1}{\hat{\gamma}_k}\left(b_l r_{l+1,k-1} + (a_l - \hat{\alpha}_{k-1}) r_{l,k-1} + c_l r_{l-1,k-1} - \hat{\gamma}_{k-1} r_{l,k-2}\right). \end{aligned} \tag{3.26}$$

This equation combined with **(3.13)** almost furnish the algorithm. Since $\hat{\gamma}_k$ is defined in terms of r_{kk} we slightly have to change (3.26) for $l = k$ and finally obtain:

ModChebUpdate(q_n) *Given* a *set of* weight functions ω_j and *the* associated *Jacobi* matrices $J_n^{(j)}$ *by (2.3), the* moments $\nu_0^{(j)}$ *by (1.6)* , $j = 1, 2, \ldots, N$, *and the system of polynomials* $\{ q_l \}_{l=0}^{2n}$ *by (3.8), this algorithm computes the recurrence coefficients of the polynomials* $\hat{\psi}_k$, $k = 1, 2, \ldots, n$, *orthonormal with respect to w.*

Initialize. *Set* $\hat{\gamma}_0 = 0$ and $r_{l,-1} = 0$, $l = 1, \ldots, 2n - 1$.

- *compute the modified moments* ν_l *relative to* q_l, $l = 0, 1, \ldots, 2n$, *by* **Modmoment(q_l)**
- *compute* $\hat{\psi}_0$ *by (3.3)*
$$\hat{\psi}_0 = \left(\sum_{j=1}^{N} \varepsilon_j \nu_0^{(j)} \right)^{-1/2}.$$
- *compute* r_{l0} *by (3.25)*
$$r_{l0} = < q_l, \hat{\psi}_0 > = \hat{\psi}_0 \nu_l, \quad l = 0, \ldots, 2n - 1.$$
- *compute* $\hat{\alpha}_0$ *by (3.13)*
$$\hat{\alpha}_0 = a_0 + b_0 \frac{r_{10}}{r_{00}}.$$

Iterate. *For* $k = 1, 2, \ldots, n$ *do*

- *compute* r_{kk} by (3.26) and (3.13)
$$r_{kk} = \Big(\frac{r_{k-1,k-1}}{b_{k-1}} [b_k r_{k+1,k-1} + (a_k - \hat{\alpha}_{k-1}) r_{k,k-1} + c_k r_{k-1,k-1} - \hat{\gamma}_{k-1} r_{k,k-2}] \Big)^{1/2}.$$
- compute $\hat{\gamma}_k$ by (3.13)
$$\hat{\gamma}_k = b_{k-1} \frac{r_{kk}}{r_{k-1,k-1}}.$$

if $k < n$ *then for* $l = k+1, k+2, \ldots, 2n-k$ *do*

- *compute* r_{lk} *by (3.26)*
$$r_{lk} = \frac{1}{\hat{\gamma}_k} \left(b_l r_{l+1,k-1} + (a_l - \hat{\alpha}_{k-1}) r_{l,k-1} + c_l r_{l-1,k-1} - \hat{\gamma}_{k-1} r_{l,k-2} \right).$$
- *compute* $\hat{\alpha}_k$ *by (3.13)*
$$\hat{\alpha}_k = a_k - b_{k-1} \frac{r_{k,k-1}}{r_{k-1,k-1}} + b_k \frac{r_{k+1,k}}{r_{kk}}.$$

End.

Remarks. **1.** The algorithm requires $N\mathcal{O}((n+1)^2)$ flops plus n square root computations. 2. It is well known (see e.g. Gautschi [9]), that the choice of the system $\{q_l\}$ affects the condition of the nonlinear map from the modified moments to the recursion coefficients.

4. Examples

The purpose of this section is to illustrate the numerical performance of the three algorithms. All computations were carried out on a SUN 3/50 in double precision.

As we will see, due to roundoff errors, the algorithms dont produce in any case the same numbers. How do we decide which numbers are the right ones? The most obvious test - using the associate Gauss quadrature rule for checking the orthonormality of the computed polynomials - is not without difficulties (compare Gautschi [5]). Therefore we transcribed one algorithm also into MATHEMATICA and used high precision arithmetic.

In all examples we have computed the orthonormal polynomials, more precisely the three term recurrence coefficients, up to degree 50. For every algorithm we have compared the FORTRAN double precision results with MATHEMATICA high precision results. In the corresponding tables we have listed the maximal polynomial degree for which the relative deviation of this two results is less than 10^{-14}. We only consider the case of two intervals, since the extension to more intervals does not produce any additional difficulties.

Example 4.1. Let $\omega_1(x) := \mathcal{X}_{[l_1,u_1]}(x)$, $\omega_2(x) := \mathcal{X}_{[l_2,u_2]}(x)$ and

$$\omega(x) := \omega_1(x) + \omega_2(x). \tag{4.1}$$

The orthogonal polynomials $p_n^{(1)}, p_n^{(2)}$ w.r.t. ω_1, ω_2 are the suitable translated *Legendre polynomials.* The modified moments are based on the Legendre polynomial L_n and on the Chebyshev polynomial of the first kind T_n w.r.t. the whole interval [I, u] = $[-1, 1]$.

					modCheb		
l_1	u_1	l_2	u_2	St.	Ch.	U.(L_n)	U.(T_n)
-1.0	-0.1	0.2	1.0	>50	27	27	31
-1.0	-0.4	0.6	1.0	>50	8	7	7
-1.0	-0.8	0.9	1.0	>50	4	4	4
-1.0	0.8	0.9	1.0	>50	46	38	44
-1.0	-0.3	-0.3	1.0	>50	>50	>50	>50
-1.0	-0.3	-0.5	1.0	>50	>50	>50	>50
-1.0	0.5	-0.7	1.0	>50	>50	>50	>50
-1.0	1.0	0.8	0.9	>50	>50	>50	>50
-1.0	1.0	-0.7	0.5	>50	>50	>50	>50
-1.0	1.0	-1.0	0.8	>50	>50	>50	>50
-1.0	1.0	-1.0	1.0	>50	>50	>50	>50

Table 4.1. ***Performance of the Stieltjes algorithm and the modified Chebyshev algorithms for w given by (4.1)***

The Stieltjes algorithm works extremely well in all cases. So do the modified **Chebyshev** algorithms as long as the two intervals have at least one point in common. If there is a gap between the intervals the latter algorithms become severely unstable, compare also Gautschi [4, Example 4.7]. As suggested by Gautschi [8, Example 5.5] one might use in this cases modified moments defined by orthogonal polynomials w.r.t. a weight function which has the same support as w. Therefore we introduce the following weight functions, which may be viewed as a generalisation of the ordinary Chebyshev weight function onto two intervals,

$$\omega^{u_1}(x) = \begin{cases} \dfrac{|u_1 - x|}{\sqrt{(l_1 - x)(u_1 - x)(l_2 - x)(u_2 - x)}} & \text{for } x \in [l_1, u_1] \cup [l_2, u_2] \\ 0 & \text{otherwise,} \end{cases}$$

$$\omega^{l_2}(x) = \begin{cases} \dfrac{|l_2 - x|}{\sqrt{(l_1 - x)(u_1 - x)(l_2 - x)(u_2 - x)}} & \text{for } x \in [l_1, u_1] \cup [l_2, u_2] \\ 0 & \text{otherwise.} \end{cases} \tag{4.2}$$

The associated orthogonal polynomials $P_n^{u_1}$, $P_n^{l_2}$ were studied by Peherstorfer [16]. In particular he derived a recurrence relation for the three term recurrence coefficients. Using this polynomials we obtain:

					modCheb		
l_1	u_1	l_2	u_2	St.	Ch.	U.($P_n^{u_1}$)	U.($P_n^{l_2}$)
-1.0	-0.1	0.2	1.0	>50	27	>50	>50
-1.0	-0.4	0.6	1.0	>50	8	>50	42
-1.0	-0.8	0.9	1.0	>50	4	11	37
-1.0	0.8	0.9	1.0	>50	46	>50	>50

Table 4.2. *Performance of the Stieltjes algorithm and the modified Chebyshev algorithms for w given by (4.1)*

Now the performance of **modChebUpdate** is indead better, but in general not as good as **Stieltjes**.

The next computations are based on a different representation of w. For $l_1 < u_1 < l_2 < u_2$ we have

$$\begin{aligned} \omega(x) &= \mathcal{X}_{[l_1,u_1]}(x) + \mathcal{X}_{[l_2,u_2]}(x) \\ &= \mathcal{X}_{[l_1,u_2]}(x) - \mathcal{X}_{[u_1,l_2]}(x). \end{aligned} \tag{4.3}$$

Using the second representation (4.3) of w we obtain:

					modCheb			
l_1	u_1	l_2	u_2	St.	Ch.	U.(L_n)	U.($P_n^{u_1}$)	U.($P_n^{l_2}$)
-1.0	-0.1	0.2	1.0	24	27	25	27	27
-1.0	-0.4	0.6	1.0	6	8	7	9	9
-1.0	-0.8	0.9	1.0	2	3	4	3	3
-1.0	0.8	0.9	1.0	34	35	36	38	38

Table 4.3. *Performance of the Stieltjes algorithm and the modified Chebyshev algorithms for w given by (4.3)*

As one might expect, here are all algorithms tend to be unstable. It seems that this approach is only of academic interest.

Example 4.2. Let $c_1 := (l_1 + u_1)/2$ and $d_1 := (u_1 - l_1)/2$. Define the weight functions $\omega_1(x) := [d_1^2 - (x - c_1)^2]^{-1/2}\mathcal{X}_{[l_1,u_1]}(x)$, $\omega_2(x) := \mathcal{X}_{[l_2,u_2]}(x)$, and

$$\omega(x) = \omega_1(x) + \omega_2(x). \tag{4.4}$$

$p_n^{(1)}$ is now a suitable scaled ***Chebyshev polynomial of the first kind.*** Although ω_1 and ω_2 have a "different nature", the algorithms have the same qualitative behavior as in example 4.1, compare also Gautschi [4, Example 4.9].

					modCheb			
l_1	u_1	l_2	u_2	St.	Ch.	U.(L_n)	U.($P_n^{u_1}$)	U.($P_n^{l_2}$)
-1.0	-0.1	0.2	1.0	>50	34	25	>50	>50
-1.0	-0.4	0.6	1.0	>50	7	9	>50	31
-1.0	-0.8	0.9	1.0	>50	4	3	27	5
-1.0	0.8	0.9	1.0	>50	>50	>50	>50	>50
-1.0	-0.3	-0.3	1.0	>50	>50	>50	–	–
-1.0	-0.3	-0.5	1.0	>50	>50	>50	–	–
-1.0	0.5	-0.7	1.0	>50	>50	>50		
-1.0	1.0	0.8	0.9	>50	>50	>50	–	–
-1.0	1.0	-0.7	0.5	>50	>50	>50	–	–
-1.0	1.0	-1.0	0.8	>50	>50	>50	–	–
-1.0	1.0	-1.0	1.0	>50	>50	>50	–	

Table 4.4. *Performance of the Stieltjes algorithm and the modified Chebyshev algorithms for w given by (4.4)*

Example 4.3. Let $c_i := (l_i + u_i)/2$, $d_i := (u_i - l_i)/2$, and $\omega_i(x) := [d_i^2 - (x - c_i)^2]^{-1/2}\mathcal{X}_{[l_i,u_i]}(x)$, $i = 0, 1$. The polynomials ψ_n that are orthogonal in $[l_1, u_1] \cup [l_2, u_2]$ with respect to the weight function

$$\omega(x) = \omega_1(x) + \omega_2(x). \tag{4.5}$$

were studied by Saad [17], for $l_1 < u_1 < l_2 < u_2$, in connection with the solution of indefinite linear systems. He derived a method for computing this polynomials by exploiting the properties of Chebyshev polynomials.

Note, that the orthogonal polynomials ψ_n are also of interest in Gaussian quadrature. Here, one has now the possibility to deal in a closed form with functions having a singularity in the interior of a given interval $[l, u]$, e.g., $l = l_1 < u_1 = l_2 < u_2 = u$.

Again, the Stieltjes algorithm as well as the modified Chebyshev algorithms behave like in the previous examples.

					modCheb			
l_1	u_1	l_2	u_2	St.	Ch.	U.(L_n)	U.($P_n^{u_1}$)	U.($P_n^{l_2}$)
-1.0	-0.1	0.2	1.0	>50	35	29	>50	>50
-1.0	-0.4	0.6	1.0	>50	8	9	>50	>50
-1.0	-0.8	0.9	1.0	>50	4	4	15	7
-1.0	0.8	0.9	1.0	>50	>50	>50	>50	>50
-1.0	-0.3	-0.3	1.0	>50	>50	>50	–	–
-1.0	-0.3	-0.5	1.0	>50	>50	>50	–	–
-1.0	0.5	-0.7	1.0	>50	>50	>50	–	–
-1.0	1.0	0.8	0.9	>50	>50	>50	–	–
-1.0	1.0	-0.7	0.5	>50	>50	>50	–	–
-1.0	1.0	-1.0	0.8	>50	>50	>50	–	–
-1.0	1.0	-1.0	1.0	>50	>50	>50	–	–

Table 4.5. *Performance of the Stieltjes algorithm and the modified Chebyshev algorithms for w given by (4.5)*

Example 4.4. In this example we consider the weight function $w(x) := \omega^{u_1}(x) + \omega^{l_2}(x)$, where ω^{u_1} and ω^{l_2} are defined by (4.2). We have

$$\omega(x) = \begin{cases} \dfrac{|x - (u_1 + l_2)/2|}{2\sqrt{(l_1 - x)(u_1 - x)(l_2 - x)(u_2 - x)}} & \text{for } x \in [l_1, u_1] \cup [l_2, u_2] \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

The symmetric case $l_1 = -u_2$ and $u_1 = -l_2$ is of interest in the diatomic linear chain (Wheeler [21]). This special case has been studied also by Gautschi [6]. He computed the three term recurrence coefficients in a closed form.

However, for the general case we obtain:

					modCheb			
l_1	u_1	l_2	u_2	St.	Ch.	U.(L_n)	U.($P_n^{u_1}$)	U.($P_n^{l_2}$)
-1.0	-0.1	0.2	1.0	>50	35	33	>50	>50
-1.0	-0.4	0.6	1.0	>50	10	9	>50	>50
-1.0	-0.8	0.9	1.0	11	4	4	11	11
-1.0	0.8	0.9	1.0	>50	>50	>50	>50	>50

Table *4.6. Performance of the Stieltjes algorithm and the modified Chebyshev algorithms for w given by (4.6)*

As long as the gap between the two intervals is not too big the Stieltjes algorithm and the modified Chebyshev algorithm based on the orthogonal polynomials w.r.t. ω^{u_1} and ω^{l_2} perform very well.

Example 4.5. Let

$$\omega(x) := \omega_1(x)+\omega_2(x)= \begin{cases} 1 \text{ for } x \in [-1.0, -0.4] \cup [0.6, 1.0] \\ 0 \text{ otherwise} \end{cases} . \qquad (4.7)$$

The following figure shows the corresponding orthonormal polynomials of degree 2 (dotted curve), 3 (continous curve), 4 (dashed curve), and 5 (dash-dotted curve).

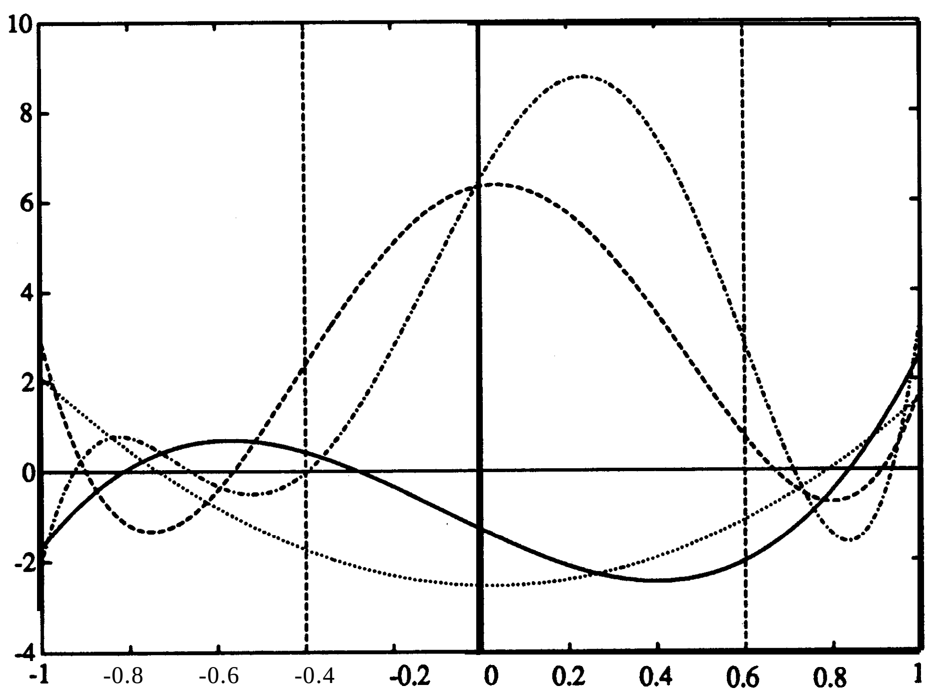

Figure 4.7. Orthonormal polynomials of degree 2,3,4,5 with respect to w given by (4.7)

Here, the orthonormal polynomial of degree 3 has a zero in the gap $[-0.4, 0.6]$. However, it is easy to show that orthogonal polynomials on two disjoint intervals have at most one zero in the gap (see e.g. Szegö [19, p.50]).

Conclusions

The Stieltjes algorithm seems to be the method of choice for generating orthogonal polynomials over several intervals under the given circumstances. It is stable in almost every case and, unlike in the usual situation, the computation of the inner products is relatively simple. But, if the map from the modified moments to the recurrence coefficients is well conditioned one may also choose one of the modified moment based algorithms. They are in particular attractive, when the required modified moments are known analytically. In this case the complexity of these algorithms does not depend on the number of underlying intervals.

The Stieltjes algorithm as well as the algorithm for computing the modified moments is-straight forward to parallelize.

Acknowledgement

We would like to thank George Cybenko for providing us his MATLAB program for the Cholesky decomposition of a Toeplitz plus Hankel Matrix. The first author would like to thank Mark Kent for several helpful discussions.

References

1. M. Branders, Application of Chebyshev polynomials in numerical integration (Flemish), Dissertation, Catholic University of Leuven, Leuven, 1976.
2. W. Gautschi, On the construction of Gaussian quadrature rules from modified moments, Math. Comp. , 24 (1970), pp. 245-260.
3. W. Gautschi, A survey of Gauss-Christoffel quadrature formulae, in E. B. Christoffel: The influence of his work in mathematics and the physical sciences; International Christoffel symposium; A collection of articles in honour of Christoffel on the 150th anniversary of his birth, P. L. Butzer and F. Fehkr, eds., Birkhäuser, Base1 (1981), pp. 72-147.
4. W. Gautschi, On generating orthogonal polynomials, SIAM J. Sci. Stat. Comput., 3 (1982), pp. 289-317.
5. W. Gautschi, How and how not to check Gaussian quadrature formulae, BIT 23 (1983), pp. 209-216.
6. W. Gautschi, On some orthogonal polynomials of interest in theoretical chemistry, BIT 24 (1984), pp. 473-483.

7. W. Gautschi, Orthogonal polynomials - Constructive theory and applications, J. Comput. Appl. Math. , **12 & 13 (1985),** pp. 61-76.
8. W. Gautschi, Questions of numerical condition related to polynomials, in Studies in Numerical Analysis, G.H. Golub, ed., Math. Assoc. America, Washington, DC **(1985),** pp. 140-177.
9. W. Gautschi, On the sensitivity of orthogonal polynomials to perturbations in the moments, Numer. Math. , 48 **(1986),** pp. 369-382.
10. I. Gohberg, T. Kailath, and I. Koltracht, Efficient solution of linear systems of equations with recursive structure, Linear Alg. Appl. **, 80 (1986),** pp. **81-113.**
11. G.H. Golub and J.H. **Welsch,** Calculation of Gauss quadrature rules, Math. Comp. , 23 **(1969),** pp. 221-230.
12. G.H. Golub and C.F. Van Loan, Matrix computations, 2nd ed., Johns Hopkins, Baltimore, 1989.
13. G. Heinig, P. Jankowski, and K. Rost, Fast inversion of Toeplitz-plus-Hankel matrices, Numer. Math. , 52 **(1988),** pp. 665-682.
14. M.D. Kent, Chebyshev, Krylov, Lanczos: Matrix relationships and computations, **PhD** thesis, Stanford, 1989.
15. H. Lev-Ari and T. Kailath, Triangular factorization of structured hermitian matrices, in I. **Schur** methods in operator theory and signal processing, Operator Theory: Advances and Application, vol. 18, I. Gohberg, ed., **Birkhäuser,** Base1 **(1986),** pp. 301-323.
16. F. Peherstorfer, Extremalpolynome in der L^1- und L^2- *Norm* auf zwei **disjunkten** Intervallen, in Approximation Theory and Functional Analysis, P.L. Butzer, R.L. Stens, and B. Sz.-Nagy, eds., **Procedings,** Conf. Math. Res. Inst. Oberwolfach, 1983, ISNM. 65, **Birkhäueser,** Basel, 1984, pp. 269-280.
17. Y. Saad, Iterative solution of **indefinite** symmetric linear systems by methods using orthogonal polynomials over two disjoint **intervals,SIAM** J. Numer. Anal., **20 (1983),** pp. **784-811.**
18. R.A. Sack and A.F. Donovan, An algorithm for Gaussian quadrature given modified moments, Numer. Math. , **18 (1971/72),** pp. **465-478.**
19. G. **Szegö,** Orthogonal polynomials, AMS Colloquium publications, Vol. **23, 3rd** ed., Providence, RI, 1967.
20. J.C. Wheeler, Modified moments and Gaussian quadrature, Rocky Mt. J. Math. , **4 (1974),** pp. 287-296.
21. J.C. Wheeler, Modified moments and continued fraction coefficients for the **diatomic** linear chain, J. Chem. Phys. **80 (1984),** pp. 472-476.
22. H. Wilf, Mathematics for the physical sciences, Wiley, New York, **1962.**