On computing modified moments for half-range Hermite weights

Teresa Laudadio[1*†], Nicola Mastronardi[1†] and Paul Van Dooren[2†]

[1*]Istituto per le Applicazioni del Calcolo "M. Picone", Consiglio Nazionale delle Ricerche, via Amendola 122/D, Bari, Italy.
[2]Department of Mathematical Engineering, Catholic University of Louvain, Avenue Georges Lemaitre 4, Louvain-la-Neuve, Belgium.

*Corresponding author(s). E-mail(s): teresa.laudadio@cnr.it;
Contributing authors: nicola.mastronardi@cnr.it;
paul.vandooren@uclouvain.be;
†These authors contributed equally to this work.

Dedicated to Froilán M. Dopico on the occasion of his 60th birthday.

Abstract

In this paper we consider the computation of the modified moments for the system of Laguerre polynomials on the real semiaxis with the Hermite weight. These moments can be used for the computation of integrals with the Hermite weight on the real semiaxis via product rules. We propose a new computational method based on the construction of the null-space of a rectangular matrix derived from the three–term recurrence relation of the system of orthonormal Laguerre polynomials. It is shown that the proposed algorithm computes the modified moments with high relative accuracy and linear complexity. Numerical examples illustrate the effectiveness of the proposed method.

Keywords: Null–space, Gaussian quadrature rule, modified moments, product rule

1 Introduction

Integrals defined with the Hermite weight on the real semiaxis appear in many applications (see, for instance, [1, 2, 7, 17]). Such integrals can be computed using modified moments via product rules. In [6], Gautschi observed a great loss of accuracy in computing such modified moments, having only three correct decimal digits for the size of the problem equal to 6, and in [7] Gautschi described a way to overcome this problem by computing the modified moments with extended precision arithmetic.

In this paper, we revisit the problem of computing the modified moments for the system of orthonormal Laguerre polynomials on the real semiaxis with the Hermite weight, also called half–range Hermite weight. We propose a new method to compute these modified moments, based on the construction of the null-space of a rectangular matrix derived from the three–term recurrence relation of the system of orthonormal Laguerre polynomials. It is shown that the proposed algorithm computes the modified moments with high relative accuracy in floating point arithmetic, thereby avoiding the use of extended precision arithmetic. Numerical examples show the effectiveness of the proposed approach.

The paper is organized as follows. The basic properties of Laguerre polynomials are introduced in Section 2. The main results of the paper are in Sections 3 and 4. There we show that the modified moments can be computed via the null–space of a particular matrix, derived from the three–term recurrence formulas of the Laguerre polynomials, and we give an algorithm of linear complexity to compute them. In Section 5 we give an application of the use of modified moments based on a product rule, and show the numerical efficiency of this approach in Section 6. We then end with a few concluding remarks in Section 7, and an Appendix including the `Matlab` codes of the algorithms described in the manuscript.

2 Orthonormal Laguerre polynomials

The orthonormal Laguerre polynomials are orthogonal polynomials satisfying the following three–term recurrence relation [16, p. 101]:

$$\begin{cases} \mathcal{L}_0(x) = 1, \\ \beta_1 \mathcal{L}_1(x) = (x - \alpha_1)\mathcal{L}_0(x), \\ \beta_\ell \mathcal{L}_\ell(x) = (x - \alpha_\ell)\mathcal{L}_{\ell-1}(x) - \beta_{\ell-1}\mathcal{L}_{\ell-2}(x), \quad \ell \geq 2, \end{cases} \tag{1}$$

with $\beta_\ell = -\ell$, and $\alpha_\ell = 2\ell - 1$, $\ell = 0, 1, \ldots$.
They are orthonormal in the interval $[0, \infty)$ with respect to the weight function $\omega(x) = e^{-x}$, i.e.,

$$\int_0^\infty e^{-x} \mathcal{L}_i(x)\mathcal{L}_j(x)dx = \delta_{i,j}.$$

Let us denote by J_ℓ the symmetric tridiagonal matrix of order ℓ,

$$J_\ell = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \beta_2 & \ddots & \ddots & \\ & & \ddots & \alpha_{\ell-1} & \beta_{\ell-1} \\ & & & \beta_{\ell-1} & \alpha_\ell \end{bmatrix},$$

with eigenvalue decomposition $J_\ell = Z_\ell \Lambda_\ell Z_\ell^T$, where $\Lambda_\ell = \operatorname{diag}\left(\lambda_1^{(\ell)}, \lambda_2^{(\ell)}, \ldots, \lambda_\ell^{(\ell)}\right)$ and $Z_\ell = [z_{i,j}^{(\ell)}]_{i,j=1}^{\ell}$ orthogonal, i.e., $Z_\ell Z_\ell^T = I_\ell$. It turns out that $\lambda_k^{(\ell)}$ are the zeros of $\mathcal{L}_\ell(x)$, i.e., $\mathcal{L}_\ell(\lambda_k^{(\ell)}) = 0$, and, denoted by $\mathbf{z}_k^{(\ell)}$ the k–th column of Z_ℓ, $k = 1, 2, \ldots, \ell$, i.e.,

$$Z_\ell = \left[\mathbf{z}_1^{(\ell)}, \mathbf{z}_2^{(\ell)}, \cdots, \mathbf{z}_{\ell-1}^{(\ell)}, \mathbf{z}_\ell^{(\ell)} \right],$$

then [10, 11]

$$\mathbf{z}_k^{(\ell)} = \frac{\hat{\mathbf{z}}_k^{(\ell)}}{\|\hat{\mathbf{z}}_k^{(\ell)}\|_2}, \quad \text{with} \quad \hat{\mathbf{z}}_k^{(\ell)} = \pm \begin{bmatrix} \mathcal{L}_0(\lambda_k^{(\ell)}) \\ \mathcal{L}_1(\lambda_k^{(\ell)}) \\ \vdots \\ \mathcal{L}_{\ell-2}(\lambda_k^{(\ell)}) \\ \mathcal{L}_{\ell-1}(\lambda_k^{(\ell)}) \end{bmatrix}. \tag{2}$$

The eigenvector matrix Z_ℓ can be computed with multiple relative robust representation [5, 15].

The nodes and weights of the ℓ–point Gaussian quadrature rule with respect to the Laguerre weight [8, 10] are given by $\lambda_k^{(\ell)}$ for $k = 1, \ldots, \ell$ and since

$$\mu_0 := \int_0^\infty e^{-x} dx = 1,$$

the weights $\omega_k^{(\ell)}$ can be simplified to

$$\omega_k^{(\ell)} = {\mathbf{z}_k^{(\ell)}}^2(1) = \left(\sum_{j=0}^{\ell-1} \mathcal{L}_j^2(\lambda_k^{(\ell)}) \right)^{-1}, \quad k = 1, 2, \ldots, \ell. \tag{3}$$

Furthermore, the following relation holds [16, p. 102]

$$\frac{d}{dx}\mathcal{L}_\ell(x) = \frac{\ell}{x}\left(\mathcal{L}_\ell(x) - \mathcal{L}_{\ell-1}(x)\right). \tag{4}$$

3 Modified moments

Let us first consider the modified moments

$$\hat{\mathcal{M}}_\ell = \int_0^\infty e^{-x^2} \hat{\mathcal{L}}_\ell(x) dx, \qquad \ell = 0, 1, 2, \ldots,$$

where $\hat{\mathcal{L}}_\ell(x)$ is the monic Laguerre polynomial of degree ℓ. Such moments can be expressed as [7]

$$\hat{\mathcal{M}}_\ell = \frac{(-1)^\ell \ell!}{2} \sum_{i=0}^{\ell} \hat{\sigma}_i^{(\ell)}, \quad \text{with} \quad \hat{\sigma}_i^{(\ell)} = (-1)^i \frac{\ell! \Gamma(\frac{i+1}{2})}{(\ell - i)! i!^2}, \qquad \ell = 0, 1, 2, \ldots,$$

where Γ is the gamma function [20], and the terms in the above sum can be generated recursively from the initial value $\hat{\sigma}_0^{(\ell)} = \sqrt{\pi}$ for i even, and from the initial value $\hat{\sigma}_1^{(\ell)} = -\ell$ for i odd. An algorithm for computing $\hat{\mathcal{M}}_\ell$, $\ell = 0, 1, \ldots,$ was described in [6]. Unfortunately, a great loss of accuracy was observed, since only three correct decimal digits were obtained for $\hat{\mathcal{M}}_6$. In order to overcome this severe loss of accuracy, Gautschi proposed in [7] to compute the modified moments with extended precision arithmetic.

Here, we focus on the modified moments

$$\mathcal{M}_\ell = \int_0^\infty e^{-x^2} \mathcal{L}_\ell(x) dx, \qquad \ell = 0, 1, 2, \ldots, \tag{5}$$

where $\mathcal{L}_\ell(x)$ is the normalized Laguerre polynomial of degree ℓ defined in (1). Since $\mathcal{L}_\ell(x)$ are orthonormal polynomials, the sequence $\{\mathcal{M}_\ell\}_{\ell=0}^\infty$ goes to zero for $\ell \to \infty$ [6, 7]. Similarly to $\hat{\mathcal{M}}_\ell$, they can be expressed as

$$\mathcal{M}_\ell = \frac{1}{2} \sum_{i=0}^{\ell} \sigma_i^{(\ell)}, \quad \text{with} \quad \sigma_i^{(\ell)} = (-1)^i \frac{\ell! \Gamma(\frac{i+1}{2})}{(\ell - i)! i!^2}, \qquad \ell = 0, 1, 2, \ldots. \tag{6}$$

Since

$$\Gamma(\frac{i+1}{2}) = \begin{cases} \frac{i! \sqrt{\pi}}{2^i \frac{i}{2}!}, & \text{for } i \text{ even}, \\ \\ \frac{i-1}{2}!, & \text{for } i \text{ odd}, \end{cases}$$

it follows that

$$\mathcal{M}_\ell = \frac{1}{2} \sum_{i=0}^{\ell} \sigma_i^{(\ell)} = \frac{1}{2} \left(\Sigma_1^{(\ell)} + \Sigma_2^{(\ell)} \right), \tag{7}$$

where $\Sigma_1^{(\ell)}$ is the sum of the $\sigma_i^{(\ell)}$ for i even, i.e., $\Sigma_1^{(\ell)} = \sum_{i=0}^{\lfloor \frac{\ell}{2} \rfloor} \sigma_{2i}^{(\ell)}$, and $\Sigma_2^{(\ell)}$ is the sum of the $\sigma_i^{(\ell)}$ for i odd, i.e., $\Sigma_2^{(\ell)} = \sum_{i=0}^{\lfloor \frac{\ell-1}{2} \rfloor} \sigma_{2i+1}^{(\ell)}$, with $\lfloor \eta \rfloor$ rounding $\eta \in \mathbb{R}$ to the nearest smaller or equal integer.

The even and the odd $\sigma_i^{(\ell)}$ are recursively and independently computed as follows:

$$\sigma_{i+2}^{(\ell)} = \sigma_i^{(\ell)} \frac{(\ell-i)(\ell-i-1)}{2(i+2)^2(i+1)}, \qquad i = 0, 1 \ldots, \ell-2, \tag{8}$$

with initial values $\sigma_0^{(\ell)} = \sqrt{\pi}$ and $\sigma_1^{(\ell)} = -\ell$.

Observe that all the terms in $\Sigma_1^{(\ell)}$ are positive and all the terms in $\Sigma_2^{(\ell)}$ are negative (with the exception of $\sigma_1^{(\ell)} = 0$ when $\ell = 0$). Therefore, both $\Sigma_1^{(\ell)}$ and $\Sigma_2^{(\ell)}$ can be computed with high relative accuracy.

Since $\lim_{\ell\to\infty} \mathcal{M}_\ell = 0$, then $\lim_{\ell\to\infty} \Sigma_1^{(\ell)} = -\lim_{\ell\to\infty} \Sigma_2^{(\ell)}$. Hence, although $\Sigma_1^{(\ell)}$ and $\Sigma_2^{(\ell)}$ are computed with high relative accuracy, a great loss of accuracy occurs in computing $\mathcal{M}_\ell = \frac{1}{2}\left(\Sigma_1^{(\ell)} + \Sigma_2^{(\ell)}\right)$ as ℓ increases, due to numerical cancellation (see blue plot in Figure 1).

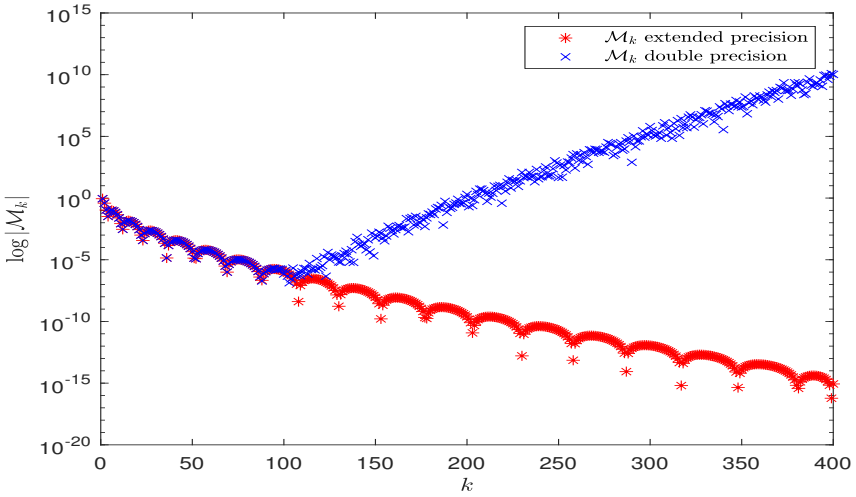

Fig. 1 Plot, on a logarithmic scale, of the modified moments $\mathcal{M}_k$, $k = 0, 1, \ldots, 400$, in absolute value, computed by using (8) (`Matlab` function `MM_1.m` in the Appendix) in double precision (denoted by '$\times$'), and in extended precision with 200 digits (denoted by '$*$').

Below, we propose a different method to compute the modified moments in floating point arithmetic, without requiring any extended precision.

Let us define

$$\mathcal{N}_\ell = \int_0^\infty e^{-x^2} x \mathcal{L}_\ell(x) dx, \qquad \ell = 0, 1, 2, \ldots.$$

Then, multiplying both sides of (1) by e^{-x^2} and considering the integral in the interval $[0, \infty)$, and multiplying both sides of (1) by xe^{-x^2} and considering the

integral in the interval $[0, \infty)$, the following system of recurrence relations for $\mathcal{M}_\ell$ and $\mathcal{N}_\ell$ holds,

$$\mathcal{M}_0 = \frac{\sqrt{\pi}}{2}, \quad \mathcal{N}_0 = \frac{1}{2}, \tag{9}$$

$$\mathcal{M}_1 = -\frac{1}{2} + \frac{\sqrt{\pi}}{2}, \quad \mathcal{N}_1 = \frac{1}{2} - \frac{\sqrt{\pi}}{4}, \tag{10}$$

$$\ell\mathcal{M}_\ell = (2\ell-1)\mathcal{M}_{\ell-1} - \mathcal{N}_{\ell-1} - (\ell-1)\mathcal{M}_{\ell-2}, \quad \ell \geq 2, \tag{11}$$

$$\ell\mathcal{N}_\ell = -\frac{\ell}{2}\mathcal{M}_{\ell-1} + \frac{\ell-1}{2}\mathcal{M}_{\ell-2} + (2\ell-1)\mathcal{N}_{\ell-1} - (\ell-1)\mathcal{N}_{\ell-2}. \tag{12}$$

Unfortunately, the straightforward implementation of (11) and (12) in `Matlab` with double precision (`Matlab` function `MM_2.m` in the Appendix) turns out to be unstable, as illustrated in Figure 2.

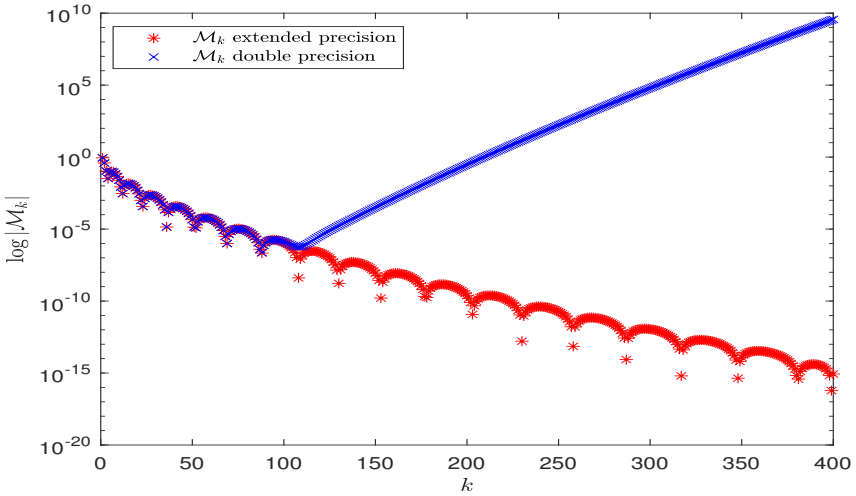

Fig. 2 Plot, on a logarithmic scale, of the entries, in absolute value, of the modified moments $\mathcal{M}_k$, $k = 0, 1, \ldots, 400$, computed by using the implementation of (11) and (12) (`Matlab` function `MM_2.m` in the Appendix) in double precision (denoted by '$\times$'), and in extended precision with 200 digits (denoted by '$*$').

We now consider a new method to compute the sequence $\{\mathcal{M}_\ell\}_{\ell=0}^{\infty}$. Let

$$\mathbf{m}_\ell = \begin{bmatrix} \mathcal{M}_0 \\ \mathcal{M}_1 \\ \vdots \\ \mathcal{M}_{\ell-1} \\ \mathcal{M}_\ell \end{bmatrix} \in \mathbb{R}^{\ell+1} \quad \text{and} \quad \mathbf{n}_\ell = \begin{bmatrix} \mathcal{N}_0 \\ \mathcal{N}_1 \\ \vdots \\ \mathcal{N}_{\ell-1} \\ \mathcal{N}_\ell \end{bmatrix} \in \mathbb{R}^{\ell+1}, \ \ell = 0, 1, 2, \ldots,$$

and define

$$A_\ell = \begin{bmatrix} 1 & -1 & & & & & \\ -1 & 3 & -2 & & & & \\ & -2 & 5 & -3 & & & \\ & & -3 & \ddots & \ddots & & \\ & & & \ddots & 2\ell-3 & -\ell+1 & \\ & & & & -\ell+1 & 2\ell-1 & -\ell \end{bmatrix} \in \mathbb{R}^{\ell\times(\ell+1)},$$

$$B_\ell = \begin{bmatrix} 1 & & & & \\ -1 & 2 & & & \\ & -2 & 3 & & \\ & & -3 & \ddots & \\ & & & \ddots & \ell-1 \\ & & & & -\ell+1 \ \ \ell \end{bmatrix} \in \mathbb{R}^{\ell\times\ell}.$$

Then, the recurrence relations (11) and (12) can be written in matrix form, respectively, as,

$$A_\ell \mathbf{m}_\ell = \mathbf{n}_{\ell-1}, \tag{13}$$

$$A_{\ell-1}\mathbf{n}_{\ell-1} = \frac{1}{2}B_{\ell-1}\mathbf{m}_{\ell-2}. \tag{14}$$

Multiplying both sides of (13) by $A_{\ell-1}$ yields

$$A_{\ell-1}A_\ell \mathbf{m}_\ell = A_{\ell-1}\mathbf{n}_{\ell-1},$$

and replacing $A_{\ell-1}\mathbf{n}_{\ell-1}$ by $A_{\ell-1}A_\ell\mathbf{m}_\ell$ in (14), we obtain

$$A_{\ell-1}A_\ell \mathbf{m}_\ell = \frac{1}{2}B_{\ell-1}\mathbf{m}_{\ell-2},$$

i.e.,

$$2B_{\ell-1}^{-1}A_{\ell-1}A_\ell\mathbf{m}_\ell = \mathbf{m}_{\ell-2} \Leftrightarrow M_{\ell-1}\mathbf{m}_\ell = \mathbf{0}, \tag{15}$$

where

$$M_{\ell-1} = 2B_{\ell-1}^{-1}A_{\ell-1}A_\ell - [I_{l-1}\ \mathbf{0}\ \mathbf{0}].$$

It is easy to verify that

$$M_\ell = \begin{bmatrix} 3 & -8 & 4 & & & & & \\ -2 & 9 & -14 & 6 & & & & \\ & -4 & 15 & -20 & 8 & & & \\ & & -6 & \ddots & \ddots & \ddots & & \\ & & & \ddots & 6\ell-9 & -6\ell-8 & 2\ell & \\ & & & & -2\ell+2 & 6\ell-3 & -6\ell-2 & 2\ell+2 \end{bmatrix}.$$

Hence, by (15), $\mathbf{m}_{\ell+1}$ belongs to the right null–space of M_ℓ. Since M_ℓ has full row–rank, its right null–space has dimension two. In the next section we describe how to retrieve the vector $\mathbf{m}_{\ell+1}$ from the right null–space of M_ℓ.

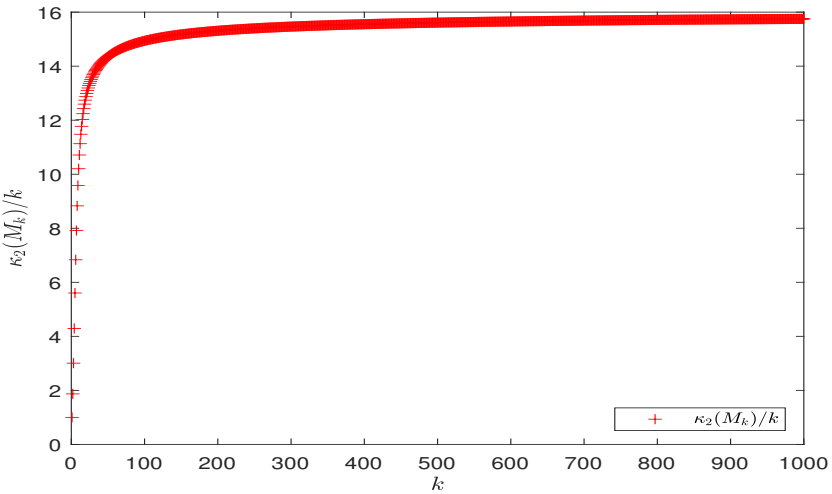

Fig. 3 Plot of the ratio $\kappa_2(M_k)/k$ for $k = 1, \ldots, 1000$, implying that $\kappa_2(M_k)$ grows linearly with k.

Remark 1 In Figure 3, $\kappa_2(M_k)/k$, i.e., the ratio between the condition number of M_k in the 2–norm, and k, is plotted. It can be noticed that this ratio becomes constant, which implies that $\kappa_2(M_k)$ grows linearly with k for large k. It follows from [19] that $\kappa_2(M_k)$ is also the condition number of the right null–space of the matrix M_k, which implies that relative errors in the calculation of this null–space will be bounded by k times the machine precision of the computer used. In fact, since M_k is a structured matrix, we can expect an even smaller sensitivity when the structure is exploited in the algorithm. This will be illustrated in the numerical results shown below.

Remark 2 Defining the diagonal sign matrices $D_k = \mathrm{diag}(d_1, d_2, \ldots, d_k)$ for $k = \ell$ and $k = \ell + 2$, where $d_i = (-1)^{i+1}$, $i \geq 1$, one can show that the matrix $M_\ell^{(D)} = D_\ell M_\ell D_{\ell+2}$ is totally nonnegative. Therefore, its right null–space can be computed with high relative accuracy [14].

4 Computation of the null–space

In this section, we describe an algorithm to compute the sequence $\mathcal{M}_k$, $k = 0, 1, \ldots, \ell$, from the right null-space of M_ℓ. The null-space of M_ℓ can easily be retrieved by reducing the matrix to lower triangular form by applying two sequences of ℓ Givens rotations to the right. For the sake of simplicity let

$M := M_\ell$, $\mathbf{m} := \mathbf{m}_{\ell+1}$, and denote by $\mathbf{e}_i,\ i = 1, \ldots, \ell+2$, the vectors of the canonical basis of $\mathbb{R}^{\ell+2}$.

Let us initialise $\tilde{M}^{(0)} := M_\ell$ and consider the sequence of Givens rotations

$$\tilde{G}_i = \begin{bmatrix} I_i & & & \\ & \tilde{c}_i & \tilde{s}_i & \\ & -\tilde{s}_i & \tilde{c}_i & \\ & & & I_{\ell-i} \end{bmatrix} \in \mathbb{R}^{(\ell+2)\times(\ell+2)},\ i = 1, 2, \ldots, \ell,$$

such that

$$\begin{bmatrix} \tilde{c}_i & \tilde{s}_i \\ -\tilde{s}_i & \tilde{c}_i \end{bmatrix} \begin{bmatrix} \tilde{m}_{i,i+1}^{(i-1)} \\ \tilde{m}_{i,i+2}^{(i-1)} \end{bmatrix} = \begin{bmatrix} \tilde{m}_{i,i+1}^{(i)} \\ 0 \end{bmatrix},$$

i.e.,

$$\tilde{c}_i = \frac{\tilde{m}_{i,i+1}^{(i-1)}}{\sqrt{\tilde{m}_{i,i+1}^{(i-1)^2} + \tilde{m}_{i,i+2}^{(i-1)^2}}}, \quad \tilde{s}_i = \frac{\tilde{m}_{i,i+2}^{(i-1)}}{\sqrt{\tilde{m}_{i,i+1}^{(i-1)^2} + \tilde{m}_{i,i+2}^{(i-1)^2}}}.$$

Then,

$$\tilde{M}^{(i)} := \tilde{M}^{(i-1)} \tilde{G}_i^T$$

has its entry $(i, i+2)$ annihilated.

Since $\tilde{G}_i \mathbf{e}_1 = \mathbf{e}_1,\ i = 1, \ldots, \ell$, then the accumulated product

$$\tilde{Q} = \tilde{G}_1 \tilde{G}_2 \cdots \tilde{G}_{\ell-1} \tilde{G}_\ell,$$

satisfies also

$$\tilde{Q}\mathbf{e}_1 = \mathbf{e}_1. \tag{16}$$

Moreover, the last column of $\tilde{M}^{(\ell)} = \tilde{M}^{(0)} \tilde{Q}^T$ is zero. Therefore, the last column of $\tilde{Q}$, given by

$$\tilde{Q}\mathbf{e}_{\ell+2} = \begin{bmatrix} 0 \\ (-1)^\ell \prod_{i=1}^{\ell} \tilde{s}_i \\ (-1)^{\ell-1} \tilde{c}_\ell \prod_{i=2}^{\ell} \tilde{s}_i \\ \vdots \\ -\tilde{c}_{n-3} \tilde{s}_n \tilde{s}_{n-1} \tilde{s}_{n-2} \\ \tilde{c}_{n-2} \tilde{s}_n \tilde{s}_{n-1} \\ -\tilde{c}_{n-1} \tilde{s}_n \\ \tilde{c}_n \end{bmatrix},$$

belongs to the right null–space of $M^{(0)}$.

To compute a second vector of an orthogonal basis of the null–space of M, let $\hat{M}^{(0)} := \tilde{M}^{(\ell)}$. Hence, a second sequence of Givens rotations is chosen

$$\hat{G}_i = \begin{bmatrix} I_{i-1} & & & \\ & \hat{c}_i & \hat{s}_i & \\ & -\hat{s}_i & \hat{c}_i & \\ & & & I_{\ell-i+1} \end{bmatrix} \in \mathbb{R}^{(\ell+2)\times(\ell+2)},\ i = 1, 2, \ldots, \ell,$$

with

$$\begin{bmatrix} \hat{c}_i & \hat{s}_i \\ -\hat{s}_i & \hat{c}_i \end{bmatrix} \begin{bmatrix} \hat{m}_{i,i}^{(i-1)} \\ \hat{m}_{i,i+1}^{(i-1)} \end{bmatrix} = \begin{bmatrix} \hat{m}_{i,i}^{(i)} \\ 0 \end{bmatrix},$$

and

$$\hat{c}_i = \frac{\hat{m}_{i,i}^{(i-1)}}{\sqrt{\hat{m}_{i,i}^{(i-1)^2} + \hat{m}_{i,i+1}^{(i-1)^2}}}, \quad \hat{s}_i = \frac{\hat{m}_{i,i+1}^{(i-1)}}{\sqrt{\hat{m}_{i,i}^{(i-1)^2} + \hat{m}_{i,i+1}^{(i-1)^2}}}.$$

The above sequence is applied to the right of the matrices $\hat{M}^{(i)}$, such that,

$$\hat{M}^{(i)} = \hat{M}^{(i-1)} \hat{G}_i^T$$

has its entry $(i, i+1)$ annihilated.

Since $\hat{G}_i \mathbf{e}_{\ell+2} = \mathbf{e}_{\ell+2}$, $i = 1, \ldots, \ell$, then the accumulated product

$$\hat{Q} = \hat{G}_1 \hat{G}_2 \cdots \hat{G}_{\ell-1} \hat{G}_\ell$$

satisfies also

$$\hat{Q} \mathbf{e}_{\ell+2} = \mathbf{e}_{\ell+2}. \tag{17}$$

Hence, the last two columns of $\hat{M}^{(\ell)} = \hat{M}^{(0)} \hat{Q}^T$ are zero and thus, the last two columns of $Q := \tilde{Q}\hat{Q}$, are an orthogonal basis for the right null–space of M. By (16) and (17),

$$\left[\mathbf{v}_1 \middle| \mathbf{v}_2 \right] := \tilde{Q}\hat{Q} \left[\mathbf{e}_{\ell+1} \middle| \mathbf{e}_{\ell+2} \right] = \left[\tilde{Q} \begin{bmatrix} (-1)^\ell \prod_{i=1}^{\ell} \hat{s}_i \\ (-1)^{\ell-1} \hat{c}_\ell \prod_{i=2}^{\ell} \hat{s}_i \\ \vdots \\ -\hat{c}_{n-3} \hat{s}_n \hat{s}_{n-1} \hat{s}_{n-2} \\ \hat{c}_{n-2} \hat{s}_n \hat{s}_{n-1} \\ -\hat{c}_{n-1} \hat{s}_n \\ \hat{c}_n \\ 0 \end{bmatrix} \middle| \begin{bmatrix} 0 \\ (-1)^\ell \prod_{i=1}^{\ell} \tilde{s}_i \\ (-1)^{\ell-1} \tilde{c}_\ell \prod_{i=2}^{\ell} \tilde{s}_i \\ \vdots \\ -\tilde{c}_{n-3} \tilde{s}_n \tilde{s}_{n-1} \tilde{s}_{n-2} \\ \tilde{c}_{n-2} \tilde{s}_n \tilde{s}_{n-1} \\ -\tilde{c}_{n-1} \tilde{s}_n \\ \tilde{c}_n \end{bmatrix} \right]$$

is an orthogonal basis of the right null-space of M_ℓ.

In Figure 4, the absolute values of the entries of $\mathbf{v}_1$ and $\mathbf{v}_2$, for $\ell = 400$, are plotted on a logarithmic scale.

Since $\lim_{\ell \to \infty} \mathbf{v}_1(\ell) = 0$ and $\lim_{\ell \to \infty} \mathbf{v}_2(\ell) = \infty$, then $\mathbf{v}_1$ and $\mathbf{v}_2$ are the minimal and the dominant solutions of (15), respectively [9]. Moreover, $\mathbf{v}_1$ is unique up to a constant multiplicative factor [9].

On the other hand, $\mathcal{M}_\ell$ goes to 0 as ℓ goes to ∞. Hence,

$$\{\mathcal{M}_\ell\}_{\ell=0}^{\infty} = \frac{\mathcal{M}_0}{\mathbf{v}_1(0)} \{\mathbf{v}_1\}_{\ell=0}^{\infty}.$$

The vector $\mathbf{v}_1$ is computed with $O(\ell)$ floating point operations by the `Matlab` function `MM_3.m` given in the Appendix.

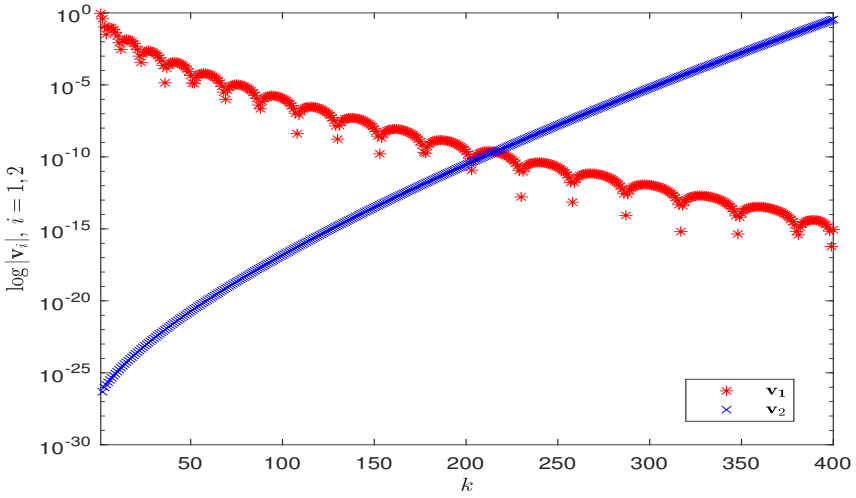

Fig. 4 Plot, on a logarithmic scale, of the entries, in absolute value, of the basis vectors $\mathbf{v}_1$ and $\mathbf{v}_2$, denoted respectively by '$*$' and '$\times$', of the null–space of M_{400}.

Furthermore, if we denote by $\mathbf{v}_3$ the solution computed by implementing straightforwardly the recurrence relations (11) and (12) (function `MM_2.m` in the Appendix), then it belongs to the right null–space of M_ℓ (see Figure 5), as

$$\texttt{svd}([\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3]) = \begin{bmatrix} 1.046203719403972 \times 10^{10} \\ 9.909635450807446 \times 10^{-1} \\ 1.379989783205324 \times 10^{-14} \end{bmatrix}.$$

Moreover,

$$\mathbf{v}_3 \approx \alpha_2 \mathbf{v}_1 + \beta_2 \mathbf{v}_2,$$

where $\alpha_2 = 9.999997323206090 \times 10^{-1}$ and $\beta_2 = -1.046203719403972 \times 10^{10}$ are computed by solving the following least squares problem

$$[\mathbf{v}_1 \mathbf{v}_2][\alpha_2, \beta_2]^T \approx \mathbf{v}_3,$$

with a relative error of the order of $\mathcal{O}(10^{-12})$.

Remark 3 A different approach to compute the null–space of M_ℓ is to compute the RQ factorization of M_ℓ :

$$M_\ell = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \check{R} \end{bmatrix} \check{Q},$$

with $\check{R} \in \mathbb{R}^{\ell \times \ell}$ nonsingular upper triangular and $\check{Q} \in \mathbb{R}^{(\ell+2)\times(\ell+2)}$ orthogonal. Since the features of this algorithm are similar to the one described earlier, we omit the details.

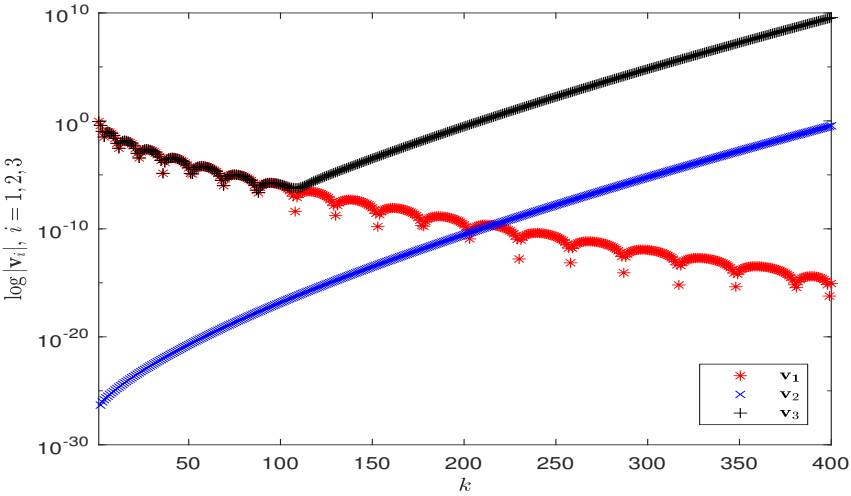

Fig. 5 Plot, on a logarithmic scale, of the entries, in absolute value, of $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$, denoted, respectively, by '$*$', '$\times$' and '$+$'.

5 Product rule

In this section we briefly describe how the integral

$$\mathcal{I}(f) = \int_0^\infty e^{-x^2} f(x)dx \tag{18}$$

with f a continuous function in $[0,\infty)$, can be computed by a product rule of interpolatory type [4]. Let $\mathcal{P}_{\ell-1}(f,x)$ be the Lagrange polynomial of degree $\ell-1$ interpolating the function f on the zeros of $\mathcal{L}_\ell(x)$, the Laguerre polynomial of degree ℓ. Let $\lambda_k^{(\ell)}$ and $\omega_k^{(\ell)}$, $k=1,\ldots,\ell$, be the nodes and the weights of the ℓ–point Gauss–Laguerre quadrature rule. They can be computed by solving a symmetric tridiagonal eigenvalue problem [10, 11]. Then

$$\begin{aligned}
\mathcal{P}_{\ell-1}(f,x) &= \sum_{j=0}^{\ell-1} \mathcal{L}_j(x) \int_0^\infty e^{-x} \mathcal{P}_{\ell-1}(f,x)\mathcal{L}_j(x)dx \\
&= \sum_{j=0}^{\ell-1} \mathcal{L}_j(x) \sum_{k=1}^{\ell} \omega_k^{(\ell)} \mathcal{P}_{\ell-1}(f,\lambda_k^{(\ell)})\mathcal{L}_j(\lambda_k^{(\ell)}) \\
&= \sum_{j=0}^{\ell-1} \mathcal{L}_j(x) \sum_{k=1}^{\ell} \omega_k^{(\ell)} f(\lambda_k^{(\ell)})\mathcal{L}_j(\lambda_k^{(\ell)}) \\
&= \sum_{k=1}^{\ell} \omega_k^{(\ell)} f(\lambda_k^{(\ell)}) \sum_{j=0}^{\ell-1} \mathcal{L}_j(\lambda_k^{(\ell)})\mathcal{L}_j(x).
\end{aligned}$$

Hence, to compute (18), the function f is replaced by the Lagrange polynomial $\mathcal{P}_{\ell-1}(f,x)$, obtaining the product rule,

$$\begin{aligned} P_\ell(f) = \int_0^\infty e^{-x^2}\mathcal{P}_{\ell-1}(f,x)dx &= \int_0^\infty e^{-x^2}\sum_{k=1}^{\ell}\omega_k^{(\ell)}f(\lambda_k^{(\ell)})\sum_{j=0}^{\ell-1}\mathcal{L}_j(\lambda_k^{(\ell)})\mathcal{L}_j(x)dx \\ &= \sum_{k=1}^{\ell}\omega_k^{(\ell)}f(\lambda_k^{(\ell)})\sum_{j=0}^{\ell-1}\mathcal{L}_j(\lambda_k^{(\ell)})\int_0^\infty e^{-x^2}\mathcal{L}_j(x)dx \\ &= \sum_{k=1}^{\ell}\omega_k^{(\ell)}f(\lambda_k^{(\ell)})\sum_{j=0}^{\ell-1}\mathcal{L}_j(\lambda_k^{(\ell)})\mathcal{M}_j \\ &= \sum_{k=1}^{\ell}\bar{\omega}_k^{(\ell)}f(\lambda_k^{(\ell)}), \end{aligned}$$

where

$$\mathcal{M}_k = \int_0^\infty e^{-x^2}\mathcal{L}_k(x)dx, \qquad k = 0,1,\ldots \tag{19}$$

are the modified moments, and

$$\bar{\omega}_k^{(\ell)} := \omega_k^{(\ell)}\sum_{j=0}^{\ell-1}\mathcal{L}_j(\lambda_k^{(\ell)})\mathcal{M}_j, \qquad k = 1,\ldots,\ell. \tag{20}$$

Let $E_\ell(f,x) = \mid \mathcal{I}(f) - P_\ell(f) \mid$ be the error approximating $\mathcal{I}(f)$ by the product rule $P_\ell(f)$. Then $E_\ell(f,x) = 0$, if f is a polynomial of degree up to $\ell - 1$ [4].

The sums $\sum_{j=0}^{\ell-1}\mathcal{L}_j(\lambda_k^{(\ell)})\mathcal{M}_j$, $k = 1,\ldots,\ell$, in (20) can be computed by means of Clenshaw's algorithm [3], that has been shown to be backward stable in [18]. To improve the results obtained by Clenshaw's algorithm, one step of iterative refinement can be applied [12]. The `Matlab` function implementing Clenshaw's algorithm with one step of iterative refinement for computing the nodes and the weights of the product rule, called `Clenshaw_PR.m`, is reported in the Appendix.

Let us denote by $\tilde{P}_\ell$ the ℓ–point product rule implemented by using Clenshaw's algorithm, and by $\hat{P}_\ell$ the ℓ–point product rule implemented by using Clenshaw's algorithm with one step of iterative refinement.

In Example 1, we show that the latter algorithm performs better than the former one. Therefore, we will use $\hat{P}_\ell$, the ℓ–point product rule implemented by using Clenshaw's algorithm with one step of iterative refinement, for the numerical experiments in Section 6.

Example 1 In this example we compute approximations of the integral

$$\mathcal{I}(x^5) = \int_0^\infty e^{-x^2}x^5dx = 1, \tag{21}$$

by the ℓ-point product rules $\tilde{P}_\ell$ and $\hat{P}_\ell$, for different values of ℓ.

Since $f(x) = x^5$ is a polynomial of degree 5, the product rules $\tilde{P}_\ell$ and $\hat{P}_\ell$ are exact for $\ell \geq 6$. The results are reported in Table 1. We can see that one step of iterative refinement improves the accuracy of the computed integral by about one digit.

$\mathcal{I}(f) = 1$				
ℓ	$\tilde{P}_\ell(f)$	$\frac{\lvert\mathcal{I}(f)-\tilde{P}_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$	$\hat{P}_\ell(f)$	$\frac{\lvert\mathcal{I}(f)-\hat{P}_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$
6	1.000000000000210	2.09×10^{-13}	1.000000000000238	2.38×10^{-13}
8	$9.999999999999909 \times 10^{-1}$	9.10×10^{-15}	$9.999999999999873 \times 10^{-1}$	1.26×10^{-14}
16	1.000000000000044	4.44×10^{-14}	1.000000000000018	1.84×10^{-14}
32	1.000000000000032	3.24×10^{-14}	1.000000000000019	1.93×10^{-14}
62	1.000000000000014	1.39×10^{-14}	1.000000000000008	7.99×10^{-15}
128	1.000000000000012	1.15×10^{-14}	1.000000000000008	8.21×10^{-15}
256	1.000000000000016	1.62×10^{-14}	1.000000000000008	8.21×10^{-15}

Table 1 Results obtained by computing the integral (21) by $\tilde{P}_\ell$ (column 2) and $\hat{P}_\ell$ (column 4) for different values of ℓ (column 1). The values of the corresponding relative errors are reported in columns 3 and 5, respectively.

6 Numerical Examples

In this section, the ℓ–point product rule $\hat{P}_\ell(f)$, described in Section 5, is used to compute (18) for different functions $f(x)$, and compared to the ℓ–point Gauss–Laguerre quadrature rule, denoted by $G_\ell(f)$, applied to

$$\mathcal{I}(f) = \int_0^\infty e^{-x^2} f(x)dx = \int_0^\infty e^{-x} \hat{f}(x)dx,$$

with $\hat{f}(x) = e^{-x^2+x} f(x)$, and to the ℓ–point Gaussian quadrature rule associated tho the half–range Hermite weight, denoted by GH_ℓ [7].

In order to compute GH_ℓ, the eigenvalue decomposition of the associated Jacobi matrix is computed [10]. Observe that the entries of the latter matrix, i.e., the coefficients of the three–term recurrence relation of the half–range Hermite polynomials, are computed by the classical Chebyshev algorithm in high-precision arithmetic, as described in [7]. These coefficients are listed in the file "`ab_hrhermite`" to 32–digit accuracy and available as supplementary material of [7].

For each numerical example, we report the results obtained by $\hat{P}_\ell(f)$, $G_\ell(f)$ and $GH_\ell(f)$, for $\ell = 10, 20, 30, \ldots, 100$, in two tables. Specifically, in the first table, the value of ℓ is displayed in the first column; the computed integral by $\hat{P}_\ell(f)$, $G_\ell(f)$ and $GH_\ell(f)$, are reported in columns 2, 3, and 4, respectively. In the second table, the value of ℓ is displayed in the first column; the relative error of the integral computed by $\hat{P}_\ell(f)$, $G_\ell(f)$ and $GH_\ell(f)$, are reported in columns 2, 3, and 4, respectively.

For all the considered examples, $GH_\ell(f)$ converges faster than $\hat{P}_\ell(f)$ to the considered integral, since $GH_\ell(f)$ and $\hat{P}_\ell(f)$ have degree of exactness $2\ell-1$ and $\ell-1$, respectively, and both converge faster than $G_\ell(f)$.

However, as already mentioned, GH_ℓ relies on the computation of the coefficients of the associated Jacobi matrix in high-precision arithmetic, whereas $\hat{P}_\ell(f)$ relies only on floating point arithmetic.

All the computations are performed in `Matlab R2022a` with machine precision $\epsilon \approx 2.22 \times 10^{-16}$.

Example 2 In this example the integral [7]

$$\mathcal{I}(f(t,n)) = \int_0^\infty e^{-x^2} t^n e^{-2tx} dx, \tag{22}$$

is computed for some values of n and t. The value of (22) is known [20] and given by $t^n(\sqrt{\pi}/2)e^{t^2}\,\texttt{erfc}(t)$, with $\texttt{erfc}(t)$ the complementary error function [20] :

$$\texttt{erfc}(t) = \frac{2}{\sqrt{\pi}} \int_t^\infty e^{-x^2} dx.$$

The results, for $t = 0.1$ and $n = 10$, and for $t = 0.5$ and $n = 20$, are reported in Tables 2–3 and 4–5, respectively.

$n=10,\ t=0.1,\ \mathcal{I}(f(0.1,10)) = 7.944643131587042 \times 10^{-11}$			
ℓ	$\hat{P}_\ell(f)$	$G_\ell(f)$	$GH_\ell(f)$
10	$7.944643133334907 \times 10^{-11}$	$7.954320866373177 \times 10^{-11}$	$7.944643131587038 \times 10^{-11}$
20	$7.944643131587042 \times 10^{-11}$	$7.944598086162397 \times 10^{-11}$	$7.944643131587047 \times 10^{-11}$
30	$7.944643131587051 \times 10^{-11}$	$7.944646489519083 \times 10^{-11}$	$7.944643131587044 \times 10^{-11}$
40	$7.944643131587045 \times 10^{-11}$	$7.944643623217692 \times 10^{-11}$	$7.944643131587049 \times 10^{-11}$
50	$7.944643131587045 \times 10^{-11}$	$7.944643140283216 \times 10^{-11}$	$7.944643131587044 \times 10^{-11}$
60	$7.944643131587052 \times 10^{-11}$	$7.944643129491703 \times 10^{-11}$	$7.944643131587053 \times 10^{-11}$
70	$7.944643131587045 \times 10^{-11}$	$7.944643131647662 \times 10^{-11}$	$7.944643131587044 \times 10^{-11}$
80	$7.944643131587040 \times 10^{-11}$	$7.944643131595562 \times 10^{-11}$	$7.944643131587040 \times 10^{-11}$
90	$7.944643131587038 \times 10^{-11}$	$7.944643131585473 \times 10^{-11}$	$7.944643131587048 \times 10^{-11}$
100	$7.944643131587036 \times 10^{-11}$	$7.944643131587206 \times 10^{-11}$	$7.944643131587040 \times 10^{-11}$

Table 2 Results obtained by computing the integral (22), for $t = 0.1$ and $n = 10$, by $\hat{P}_\ell(f)$, $G_\ell(f)$ and $GH_\ell(f)$, for different values of ℓ. The exact value of $\mathcal{I}(f(0.1,10))$, is computed in `Matlab` as $t^n(\sqrt{\pi}/2)e^{t^2}\,\texttt{erfc}(t)$.

Example 3 In this example, the integrand function f in (18) is $f(x) = \log(x+10)$. The exact value $\mathcal{I}(f) = 2.088549149913451$ is computed by `Mathematica ver. 13`. The results are reported in Table 6–7.

Example 4 We consider here the integrand function $f(x) = \sin(x)$ in (18). Also in this case, the exact value $\mathcal{I}(f) = 4.244363835020223 \times 10^{-1}$, is computed by `Mathematica ver. 13`. The results are displayed in Table 8–9.

$n = 10,\ t = 0.1,$ $\mathcal{I}(f(0.1, 10)) = 7.944643131587042 \times 10^{-11}$			
ℓ	$\frac{\lvert\mathcal{I}(f)-\hat{P}_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$	$\frac{\lvert\mathcal{I}(f)-G_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$	$\frac{\lvert\mathcal{I}(f)-GH_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$
10	2.20×10^{-10}	1.21×10^{-03}	1.30×10^{-15}
20	0	5.66×10^{-06}	1.62×10^{-16}
30	1.13×10^{-15}	4.22×10^{-07}	4.88×10^{-16}
40	4.88×10^{-16}	6.18×10^{-08}	1.62×10^{-16}
50	4.88×10^{-16}	1.09×10^{-09}	4.88×10^{-16}
60	1.30×10^{-15}	2.63×10^{-10}	6.50×10^{-16}
70	4.88×10^{-16}	7.63×10^{-12}	4.88×10^{-16}
80	1.62×10^{-16}	1.07×10^{-12}	9.76×10^{-16}
90	4.88×10^{-16}	1.97×10^{-13}	0
100	6.50×10^{-16}	2.06×10^{-14}	9.76×10^{-16}

Table 3 Relative errors obtained by computing the integral (22), for $t = 0.1$ and $n = 10$, by $\hat{P}_\ell(f)$, $G_\ell(f)$ and $GH_\ell(f)$, for different values of ℓ. The exact value of $\mathcal{I}(f(0.1, 10))$, is computed in `Matlab` as $t^n(\sqrt{\pi}/2)e^{t^2}$`erfc`(t).

$n = 20,\ t = 0.5,\ \mathcal{I}(f(0.5, 20)) = 5.203641517305822 \times 10^{-7}$			
ℓ	$\hat{P}_\ell(f)$	$G_\ell(f)$	$GH_\ell(f)$
10	$5.203811344522166 \times 10^{-7}$	$5.212141640324599 \times 10^{-7}$	$5.203641517305820 \times 10^{-07}$
20	$5.203641475039164 \times 10^{-7}$	$5.203764322313433 \times 10^{-7}$	$5.203641517305825 \times 10^{-07}$
30	$5.203641517323896 \times 10^{-7}$	$5.203643721827632 \times 10^{-7}$	$5.203641517305823 \times 10^{-07}$
40	$5.203641517305818 \times 10^{-7}$	$5.203641441812595 \times 10^{-7}$	$5.203641517305825 \times 10^{-07}$
50	$5.203641517305824 \times 10^{-7}$	$5.203641511437634 \times 10^{-7}$	$5.203641517305825 \times 10^{-07}$
60	$5.203641517305830 \times 10^{-7}$	$5.203641517717577 \times 10^{-7}$	$5.203641517305828 \times 10^{-07}$
70	$5.203641517305824 \times 10^{-7}$	$5.203641517304023 \times 10^{-7}$	$5.203641517305821 \times 10^{-07}$
80	$5.203641517305821 \times 10^{-7}$	$5.203641517303713 \times 10^{-7}$	$5.203641517305823 \times 10^{-07}$
90	$5.203641517305820 \times 10^{-7}$	$5.203641517306110 \times 10^{-7}$	$5.203641517305823 \times 10^{-07}$
100	$5.203641517305823 \times 10^{-7}$	$5.203641517305795 \times 10^{-7}$	$5.203641517305822 \times 10^{-07}$

Table 4 Results obtained by computing the integral (22), for $t = 0.5$ and $n = 20$, by $\hat{P}_\ell(f)$, $G_\ell(f)$ and $GH_\ell(f)$, for different values of ℓ. The exact value $\mathcal{I}(f(0.5, 10))$, is computed in `Matlab` as $t^n(\sqrt{\pi}/2)e^{t^2}$`erfc`(t).

Example 5 In this example $f(x) = \cos(x)$. The exact value $\mathcal{I}(f) = 6.901942235215714 \times 10^{-1}$, is computed by `Mathematica ver. 13`. The results are displayed in Table 10–11.

Since $\cos(x)$ is an even function, then

$$\int_0^\infty e^{-x^2}\cos(x)dx = \frac{1}{2}\int_{-\infty}^\infty e^{-x^2}\cos(x)dx. \tag{23}$$

Therefore, an approximation of the integral (23) can be obtained dividing the result of the ℓ–point Gauss–Hermite quadrature rule $H_\ell(f)$, by 2.

We can observe that the speed of convergence of $H_\ell(f)$ is similar to the one of $GH_\ell(f)$.

7 Conclusions and future work

In this work, an algorithm to compute the modified moments for the system of orthonormal Laguerre polynomials on the real semiaxis with the Hermite weight is described.

$n = 20,\ t = 0.5,$ $\mathcal{I}(f(0.5, 20)) = 5.203641517305822 \times 10^{-7}$			
ℓ	$\frac{\lvert\mathcal{I}(f)-\hat{P}_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$	$\frac{\lvert\mathcal{I}(f)-G_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$	$\frac{\lvert\mathcal{I}(f)-GH_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$
10	3.26×10^{-5}	1.63×10^{-3}	4.06×10^{-16}
20	8.12×10^{-9}	2.35×10^{-5}	6.10×10^{-16}
30	3.47×10^{-12}	4.23×10^{-7}	2.03×10^{-16}
40	8.13×10^{-16}	1.45×10^{-8}	6.10×10^{-16}
50	4.06×10^{-16}	1.12×10^{-9}	6.10×10^{-16}
60	1.42×10^{-15}	7.91×10^{-11}	1.01×10^{-15}
70	4.06×10^{-16}	3.45×10^{-13}	2.03×10^{-16}
80	2.03×10^{-16}	4.05×10^{-13}	2.03×10^{-16}
90	4.06×10^{-16}	5.53×10^{-14}	2.03×10^{-16}
100	2.03×10^{-16}	5.29×10^{-15}	0

Table 5 Relative errors obtained by computing the integral (22), for $t = 0.5$ and $n = 20$, by $\hat{P}_\ell(f)$, $G_\ell(f)$ and $GH_\ell(f)$, for different values of ℓ. The exact value $\mathcal{I}(f(0.5, 10))$, is computed in `Matlab` as $t^n(\sqrt{\pi}/2)e^{t^2}$`erfc`(t).

$\mathcal{I}(f) = 2.088549149913451$			
ℓ	$\hat{P}_\ell(f)$	$G_\ell(f)$	$GH_\ell(f)$
10	2.088549148171673	2.088936273112783	2.088549149913450
20	2.088549149913582	2.088480163499814	2.088549149913452
30	2.088549149913452	2.088548028299200	2.088549149913452
40	2.088549149913452	2.088549262068462	2.088549149913453
50	2.088549149913451	2.088549159184301	2.088549149913451
60	2.088549149913453	2.088549149405690	2.088549149913454
70	2.088549149913452	2.088549149883956	2.088549149913451
80	2.088549149913451	2.088549149919626	2.088549149913450
90	2.088549149913451	2.088549149912933	2.088549149913452
100	2.088549149913450	2.088549149913470	2.088549149913451

Table 6 Results obtained by computing the integral (22), for $f(x) = \log(x + 10)$, by $\hat{P}_\ell(f)$, $G_\ell(f)$ and $GH_\ell(f)$, for different values of ℓ.

It is shown that these moments can be efficiently retrieved from the null–space of a particular totally nonnegative matrix. Therefore, their computation can be carried out in floating point arithmetic with high relative accuracy. The modified moments are then used to compute integrals on the real semiaxis with the Hermite weight by a product rule.

The numerical experiments confirm the effectiveness of the proposed approach.

Declarations

- **Funding** This work was partly supported by Gruppo Nazionale Calcolo Scientifico (GNCS) of Istituto Nazionale di Alta Matematica (INdDAM).
- **Conflict of interest/Competing interests** The authors declare no competing interests.
- **Ethics approval** Not applicable.
- **Availability of data and materials** Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

$\mathcal{I}(f) = 2.088549149913451$			
ℓ	$\frac{\lvert\mathcal{I}(f)-\hat{P}_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$	$\frac{\lvert\mathcal{I}(f)-G_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$	$\frac{\lvert\mathcal{I}(f)-GH_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$
10	8.33×10^{-10}	1.85×10^{-4}	4.25×10^{-16}
20	6.25×10^{-14}	3.30×10^{-5}	4.25×10^{-16}
30	6.37×10^{-16}	5.37×10^{-7}	4.25×10^{-16}
40	6.37×10^{-16}	5.36×10^{-8}	8.50×10^{-16}
50	2.12×10^{-16}	4.43×10^{-9}	2.12×10^{-16}
60	8.50×10^{-16}	2.43×10^{-10}	1.27×10^{-15}
70	4.25×10^{-16}	1.41×10^{-11}	2.12×10^{-16}
80	0	2.95×10^{-12}	4.25×10^{-16}
90	2.12×10^{-16}	2.47×10^{-13}	4.25×10^{-16}
100	6.37×10^{-16}	8.93×10^{-15}	2.12×10^{-16}

Table 7 Relative errors obtained by computing the integral (22), for $f(x) = \log(x + 10)$, by $\hat{P}_\ell(f)$, $G_\ell(f)$ and $GH_\ell(f)$, for different values of ℓ.

$\mathcal{I}(f) = 4.244363835020223 \times 10^{-1}$			
ℓ	$\hat{P}_\ell(f)$	$G_\ell(f)$	$GH_\ell(f)$
10	$4.241937287214931 \times 10^{-1}$	$4.224971865832609 \times 10^{-1}$	$4.244363835020221 \times 10^{-01}$
20	$4.244346394183932 \times 10^{-1}$	$4.243017987824790 \times 10^{-1}$	$4.244363835020226 \times 10^{-01}$
30	$4.244364491252017 \times 10^{-1}$	$4.244345811937503 \times 10^{-1}$	$4.244363835020223 \times 10^{-01}$
40	$4.244363836583950 \times 10^{-1}$	$4.244368430550727 \times 10^{-1}$	$4.244363835020228 \times 10^{-01}$
50	$4.244363835023258 \times 10^{-1}$	$4.244364088632842 \times 10^{-1}$	$4.244363835020224 \times 10^{-01}$
60	$4.244363835020442 \times 10^{-1}$	$4.244363799253345 \times 10^{-1}$	$4.244363835020229 \times 10^{-01}$
70	$4.244363835020219 \times 10^{-1}$	$4.244363835643365 \times 10^{-1}$	$4.244363835020222 \times 10^{-01}$
80	$4.244363835020224 \times 10^{-1}$	$4.244363835291677 \times 10^{-1}$	$4.244363835020222 \times 10^{-01}$
90	$4.244363835020216 \times 10^{-1}$	$4.244363834970408 \times 10^{-1}$	$4.244363835020225 \times 10^{-01}$
100	$4.244363835020218 \times 10^{-1}$	$4.244363835026196 \times 10^{-1}$	$4.244363835020223 \times 10^{-01}$

Table 8 Results obtained by computing the integral (22), with $f(x) = \sin(x)$, $\hat{P}_\ell(f)$, $G_\ell(f)$ and $GH_\ell(f)$, for different values of ℓ.

- **Code availability** The codes described in the manuscript are reported in the Appendix.
- **Authors' contributions** The authors contributed equally to this work.
- **Acknowledgements** This work was supported partly by Gruppo Nazionale Calcolo Scientifico (GNCS) of Istituto Nazionale di Alta Matematica (INdDAM) and partly by MIUR, Project n. 20227PCCKZ.
Dedicated to Froilán M. Dopico on the occasion of his 60th birthday.
The authors would like to thank the anonymous reviewers, whose valuable comments helped to improve the manuscript.

References

[1] V.E. Ambruş, V. Sofonea, Lattice Boltzmann models based on half-range Gauss–Hermite quadratures, Journal of Computational Physics 316(1) (2016) 760–788.

[2] V.E. Ambruş, V. Sofonea, Half-range lattice Boltzmann models for the simulation of Couette flow using the Shakhov collision term, Phys. Rev. E 98 (2018) 063311.

$\mathcal{I}(f) = 4.244363835020223 \times 10^{-1}$			
ℓ	$\frac{\lvert\mathcal{I}(f)-\hat{P}_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$	$\frac{\lvert\mathcal{I}(f)-G_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$	$\frac{\lvert\mathcal{I}(f)-GH_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$
10	5.71×10^{-4}	4.56×10^{-3}	3.92×10^{-16}
20	4.10×10^{-6}	3.17×10^{-4}	6.53×10^{-16}
30	1.54×10^{-7}	4.24×10^{-6}	1.30×10^{-16}
40	3.68×10^{-10}	1.08×10^{-6}	1.17×10^{-15}
50	7.15×10^{-13}	5.97×10^{-8}	2.61×10^{-16}
60	5.16×10^{-14}	8.42×10^{-9}	1.43×10^{-15}
70	9.15×10^{-16}	1.46×10^{-10}	1.30×10^{-16}
80	2.61×10^{-16}	6.39×10^{-11}	1.30×10^{-16}
90	1.70×10^{-15}	1.17×10^{-11}	5.23×10^{-16}
100	1.04×10^{-15}	1.40×10^{-12}	0

Table 9 Relative errors obtained by computing the integral (22), with $f(x) = \sin(x)$, by $\hat{P}_\ell(f)$, $G_\ell(f)$ and $GH_\ell(f)$, for different values of ℓ.

$\mathcal{I}(f) = 0.6901942235215714$				
ℓ	$\hat{P}_\ell(f)$	$G_\ell(f)$	$H_\ell(f)$	$GH_\ell(f)$
10	0.6898966659637024	0.6964400567164973	0.6901942235215709	0.6901942235215706
20	0.6901977811358698	0.6903308552047150	0.6901942235215718	0.6901942235215710
30	0.6901942443660714	0.6902029974830444	0.6901942235215716	0.6901942235215712
40	0.6901942229305114	0.6901945465252393	0.6901942235215719	0.6901942235215717
50	0.6901942235219658	0.6901941889363955	0.6901942235215716	0.6901942235215719
60	0.6901942235215042	0.6901942222222768	0.6901942235215722	0.6901942235215716
70	0.6901942235215713	0.6901942238970094	0.6901942235215711	0.6901942235215711
80	0.6901942235215711	0.6901942234894167	0.6901942235215712	0.6901942235215716
90	0.6901942235215713	0.6901942235223959	0.6901942235215716	0.6901942235215716
100	0.6901942235215713	0.6901942235218280	0.6901942235215712	0.6901942235215719

Table 10 Results obtained by computing the integral (22), with $f(x) = \cos(x)$, by $\hat{P}_\ell(f)$, $G_\ell(f)$ and $GH_\ell(f)$, for different values of ℓ.

[3] C.W. Clenshaw, A note on the summation of Chebyshev series, Math. Tables Aids Comput., 9 (1955) 118–120.

[4] P.J. Davis, P. Rabinowitz, Methods of Numerical Integration, 2nd Edition, Academic Press, New York, 1984.

[5] I.S. Dhillon, B.N. Parlett, Multiple representations to compute orthogonal eigenvectors of symmetric tridiagonal matrices, Linear Algebra Appl., 387 (2004) pp. 1–28.

[6] W. Gautschi, On the Construction of Gaussian Quadrature Rules from Modified Moments, Mathematics of Computation, 24(110) (1970), 245–260+s1–s52.

[7] W. Gautschi, Algorithm 957: Evaluation of the Repeated Integral of the Coerror Function by Half-Range Gauss-Hermite Quadrature, ACM Transactions on Mathematical Software, 42(1) (2016) 1–10.

$\mathcal{I}(f) = 6.901942235215714 \times 10^{-1}$				
ℓ	$\frac{\lvert\mathcal{I}(f)-\hat{P}_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$	$\frac{\lvert\mathcal{I}(f)-G_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$	$\frac{\lvert\mathcal{I}(f)-H_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$	$\frac{\lvert\mathcal{I}(f)-GH_\ell(f)\rvert}{\lvert\mathcal{I}(f)\rvert}$
10	4.31×10^{-4}	9.04×10^{-3}	8.04×10^{-16}	1.28×10^{-15}
20	5.15×10^{-6}	1.97×10^{-4}	4.82×10^{-16}	6.43×10^{-16}
30	3.02×10^{-8}	1.27×10^{-5}	1.60×10^{-16}	3.21×10^{-16}
40	8.56×10^{-10}	4.67×10^{-7}	6.43×10^{-16}	3.21×10^{-16}
50	5.71×10^{-13}	5.01×10^{-8}	1.60×10^{-16}	6.43×10^{-16}
60	9.74×10^{-14}	1.88×10^{-9}	1.12×10^{-15}	1.60×10^{-16}
70	1.60×10^{-16}	5.43×10^{-10}	4.82×10^{-16}	4.82×10^{-16}
80	4.82×10^{-16}	4.65×10^{-11}	3.21×10^{-16}	1.60×10^{-16}
90	1.60×10^{-16}	1.19×10^{-12}	1.60×10^{-16}	1.60×10^{-16}
100	1.60×10^{-16}	3.71×10^{-13}	3.21×10^{-16}	6.43×10^{-16}

Table 11 Relative errors obtained by computing the integral (22), with $f(x) = \cos(x)$, by $\hat{P}_\ell(f)$, $G_\ell(f)$ and $GH_\ell(f)$, for different values of ℓ.

[8] W. Gautschi, Orthogonal Polynomials: Computation and Approximation, Oxford University Press, Oxford, UK, 2004.

[9] Gil, A., Segura, J.,Temme, N.M., Numerical Methods for Special functions. SIAM, Philadelphia, 2007.

[10] G.H. Golub, J.H. Welsch, Calculation of Gauss quadrature rules, Mathematics of Computation 23(106) (1969) 221–230.

[11] T. Laudadio, N. Mastronardi, P. Van Dooren, Computing Gaussian quadrature rules with high relative accuracy, Numerical Algorithms, 9, (2023) 767–793.

[12] N.J. Higham, Fast Solution of Vandermonde–Like Systems Involving Orthogonal Polynomials, IMA Journal of Numerical Analysis, 8(4) (1988) 473–486.

[13] N.J. Higham, Accuracy and Stability of Numerical Algorithms, 2nd Ed., Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2002.

[14] P. Koev, Accurate computations with totally nonnegative matrices, SIAM J. Matrix Anal. Appl. 29 (2007) 731–751.

[15] M. Petschow, P. Bientinesi, MR^3-SMP: A Symmetric Tridiagonal Eigensolver for Multi-Core Architectures, Parallel Computing, 37(12), (2011) 795–805.

[16] G. Szegö, Orthogonal Polynomials, 4th Ed., American Mathematical Society, Providence, RI, 1975.

[17] G. Tatsios, A. Tsimpoukis, D. Valougeorgis, The Half-Range Moment Method in Harmonically Oscillating Rarefied Gas Flows, Fluids, 2021, 6,

17. https://doi.org/10.3390/fluids6010017

[18] A. Smoktunowicz, Backward stability of Clenshaw's algorithm. BIT 42(3) (2002) 600–610.

[19] G.W. Stewart, On the perturbation of pseudo-inverses, projections and linear least squares problems, SIAM Review, 19(4), (1977) 633-662.

[20] N.M. Temme, Special Functions. An introduction to the classical functions of mathematical physics, John Wiley & Sons Inc., New York, 1996.

Appendix

The Matlab codes of the algorithms described in the manuscript are displayed in the sequel.

Function computing the modified moments by formulae (7) and (8).

```
function [mom] = MM_1(n)
% Computation of the first $n+1$ modified moments
% $M_k=\int_{0}^{\infty}e^{-x^2} L_{k}(x)dx, k=0,1,2,\ldots,n,$
% $L_k$ the orthonormal Laguerre polynomial of degree $k$
% input: n, the number of moments to be computed
% output: mom, the modified moments $M_k,k=0,1,2,\ldots,n,$
%         stored in mom(1:n+1)
 mom=zeros(n+1,1);
for k=0:n,
    mt=zeros(k+1,1);
    mt(1)=sqrt(pi);
    mt(2)=-k;
    sum=mt(1)+mt(2);
    for i=0:k-2,
        mt(i+3)=mt(i+1)*((k-i)*(k-i-1))/(2*(i+2)^2*(i+1));
        sum=sum+mt(i+3);
    end
    mom(k+1)=(sum)/2;
end
```

Function computing the modified moments by formulae (9)–(12).

```
function[mM]=MM_2(n)
% Computation of the first $n+1 $ modified moments
% $M_k=\int_{0}^{\infty}e^{-x^2} L_{k}(x)dx, k=0,1,2,\ldots,n,$
% $L_k$ the orthonormal Laguerre polynomial of degree $k$
% input: n, the number of moments to be computed
% output: mM, the modified moments $M_k, k=0,1,2,\ldots,n,$
%         stored  in mM(1:n+1)
mM=(zeros(n,1));
mN=(zeros(n,1));
mM(1)=(sqrt(pi)/2);
mM(2)=(-1/2+sqrt(pi)/2);
mN(1)=1/2;
mN(2)=(1/2-sqrt(pi)/4);
for k=2:n,
    mM(k+1)=((2*k-1)*mM(k)-(k-1)*mM(k-1)-mN(k))/k;
    mN(k+1)=(-k/2*mM(k)+(k-1)/2*mM(k-1)+(2*k-1)*mN(k)-(k-1)*mN(k-1))/k;
end
```

Function computing the modified moments by the algorithm described in Section 4.

```
function [mom] = MM_3(n)
% Computation of the first $n+1 $ modified moments
% $M_k=\int_{0}^{\infty}e^{-x^2} L_{k}(x)dx, k=0,1,2,\ldots,n,$
% $L_k$ the orthonormal Laguerre polynomial of degree $k$
% input: n, the number of moments to be computed
% output: mom, the modified moments $M_k, k=0,1,2,\ldots,n,$
%         stored in mom(1:n+1)

% construction of the matrix $M$
M=zeros(n,n+2);
d0=[3:6:6*n-3]; dm1=-[2:2:2*n-2]; dp1=-[8:6:6*n+2]; dp2=[4:2:2*n+2];
for i=1:n,
    M(i,i)=d0(i); M(i,i+1)=dp1(i); M(i,i+2)=dp2(i);
end
for i=1:n-1,
    M(i+1,i)=dm1(i);
end
% end construction matrix
% computation of the Givens rotations needed
% to compute the vector of the modified moments
j1=0;
for j=1:-1:0,
    j1=j1+1;
 for i=1:n,
        G=givens(M(i,i+j),M(i,i+j+1));
        c(i,{\color{blue}j1})=G(1,1);s(i,{\color{blue}j1})=G(1,2);
        M(:,i+j:i+j+1)=M(:,i+j:i+j+1)*G';
 end
end
 % computation of the vector of modified moments
mom=zeros(n+2,1);
mom(n+1)=c(n,2);
pr=-s(n,2);
for i=n-1:-1:1,
    mom(i+1)=c(i,2)*pr;
    pr=-pr*s(i,2);
end
mom(1)=pr;
for i=n:-1:1,
    mom(i+1:i+2)=[c(i,1) -s(i,1); s(i,1) c(i,1)]*mom(i+1:i+2);
end
% normalization of the vector
mom=mom/mom(1)*sqrt(pi)/2;
end
```

Function computing the nodes and weights of the product rule by the Clenshaw's algorithm described in Section 4.

```
function [x,wpr,wpr1] = Clenshaw_PR(n)
% computation of the nodes and weights of the product rule
% by means of the Clenshaw's algorithm (PRCA)
% with one step of iterative refinement (PRCAI)
% input: n, number of nodes and weights of the product rule
% output: x, nodes of the product rule
%         wpr, weights  (PRCA)
%         wpr1, weights  (PRCAI)
% external functions: Mod_Mom_3.m. It computes the modofied moments.
%                     gaussq,m. It computes the nodes and the weigths
% of the n-point Gauss-Laguerre quadrature rule
[x,w,a1,b1,mu1]=gaussq(6,n,0,0,0,0);
[mm] = Mod_mm_3(1200);
y=zeros(n,1);
 wpr=zeros(n,1); wpr1=zeros(n,1);
 for i=1:n-1,
     al(i)=2*i-1;
     be(i)=i;
 end
for k=1:n,
    % Clenshaw's algorithm
    y(n)=mm(n)/be(n-1);
    y(n-1)=(mm(n-1)+(al(n-1)-x(k))*y(n))/be(n-2);
    for i=n-2:-1:2,
        y(i)=(mm(i)+(al(i)-x(k))*y(i+1)-be(i)*y(i+2))/be(i-1);
    end
    y(1)=(mm(1)+(al(1)-x(k))*y(2)-be(1)*y(3));
    wpr(k)=w(k)*y(1);
    % one step of iterative refinement
    r=zeros(n,1);
    r(1)=mm(1)-(y(1)-(al(1)-x(k))*y(2)+be(1)*y(3));
    for i=2:n-2,
        r(i)=mm(i)-(be(i-1)*y(i)-(al(i)-x(k))*y(i+1)+be(i)*y(i+2));
    end
    r(n-1)=mm(n-1)-(be(n-2)*y(n-1)-(al(n-1)-x(k))*y(n));
    r(n)=mm(n)-(be(n-1)*y(n));
    y1=zeros(n,1);
    y1(n)=r(n)/be(n-1);
    y1(n-1)=(r(n-1)+(al(n-1)-x(k))*y1(n))/be(n-2);
    for i=n-2:-1:2,
        y1(i)=(r(i)+(al(i)-x(k))*y1(i+1)-be(i)*y1(i+2))/be(i-1);
    end
    y1(1)=(r(1)+(al(1)-x(k))*y1(2)-be(1)*y1(3));
```

```
    y=y+y1;
    wpr1(k)=w(k)*y(1);
end
```