Asymptotic and Iterative Methods for Gaussian Quadratures

A. Gil, J. Segura, N. M. Temme

Departamento de Matemáticas, Estadística y Computación
Universidad de Cantabria, Spain

ICIAM 2019

Gauss-quadratures: basic ideas

Given $I(f) = \int_a^b f(x)w(x)dx$, with $w(x)$ a weight function, the n-point quadrature rule

$$Q_n(f) = \sum_{i=1}^{n} w_i f(x_i)$$

is a Gaussian quadrature if $I(f) = Q_n(f)$ for f any polynomial with $\deg(f) \leq 2n-1$.

Gaussian quadrature rules are optimal in a very specific sense and they are one of the more widely used methods of integration.

The difficulty is, of course, computing the nodes x_i and weights w_i.

As it is well known, the nodes x_i, $i = 1, \ldots, n$ of the Gaussian quadrature rule are the roots of the (for instance monic) orthogonal polynomial satisfying

$$\int_a^b x^i p_n(x)w(x)dx = 0, \; i = 0, \ldots, n-1.$$

Computation of Gauss quadratures: overview

Two mains ways

1. Compute the roots of $p_n(x)$, x_i, and with this $w_j = -||p_n||^2/(p_n'(x_i)p_{n+1}(x_i))$. For this we need:
   a) A method to compute the polynomials $p_n(x)$ and the first derivative.
   b) A method to compute the roots of $p_n(x)$ (nodes x_i).

2. A second posibility is provided by the recurrence relation (for monic polynomials):

$$p_{k+1}(x) = (x - B_k)p_k(x) - A_k p_{k-1}(x), \quad k = 1, 2, \ldots,$$

where $A_0 p_{-1} \equiv 0$ and

$$A_k = \frac{||p_k||^2}{||p_{k-1}||^2}, \quad k \geq 1, \quad B_k = \frac{\langle x p_k, p_k \rangle}{||p_k||^2}, \quad k \geq 0.$$

$$< f, g >= \int_a^b f(x)g(x)w(x)dx, \; ||f|| =< f, f >$$

Classical Gaussian quadrature

This is the case for which the OPs are solutions of second order ODEs

$$C(x)y_n''(x) + B(x)y_n'(x) + \lambda_n y(x) = 0$$

with C and B polynomials.
Three cases:

1. Hermite: $w(x) = e^{-x^2}$ in $(-\infty, +\infty) \rightsquigarrow \boldsymbol{H_n(x)}$
2. Laguerre: $w(x) = x^{-\alpha}e^{-x}$, $\alpha > -1$, in $(0, +\infty) \rightsquigarrow \boldsymbol{L_n^{(\alpha)}(x)}$
3. Jacobi: $w(x) = (1-x)^{\alpha}(1+x)^{\beta}$, $\alpha, \beta > -1$, in $(-1, 1) \rightsquigarrow \boldsymbol{P_n^{(\alpha,\beta)}(x)}$

Apart from being solution of a second order ODE, the coefficients of the three-term recurrence relation are simple, as well as the coefficients in

$$y_n'(x) = a_n(x)y_n(x) + b_n y_{n-1}(x)$$

Three main approaches to compute classical Gaussian quadratures:

1. Golub-Welsch: straightforward. We just need to diagonalize a tridiagonal matrix with explicitly know entries. Not so good for large degree n ($\mathcal{O}(n^2)$).
2. Iterative methods: either we use a globally convergent method or we need previous estimations of the nodes (usually from asymptotic methods for large degree). Better for large degree than GW.
3. Asymptotic methods?: can the initial estimations from asymptotics be accurate enough? What about the weights? Best methods for large n

Some references on the computation of classical Gauss quadrature

G. H. Golub, J. H. Welsch, Calculation of Gauss quadrature rules.
Math. Comp. (1969) ALL

W. Gautschi, Orthogonal polynomials: computation and approximation.
Oxford U. Press (2004) ALL

E. Yakimiw, Accurate computation of weights in classical Gauss-Christoffel quadrature rules.
J. Comput. Phys. (1996) A+I+RR Legendre (Hermite and Laguerre $\alpha = 0$ to a lesser extent)

K. Petras, On the computation of the Gauss-Legendre quadrature formula with a given precision.
J. Comput. Appl. Math. (1999) A^*+I+? Legendre

P. N. Swarztrauber, On computing the points and weights for Gauss-Legendre quadrature.
SIAM J. Sci. Comput. (2002) A+I+FS Legendre

A. Glaser, X. Liu, V. Rokhlin, A fast algorithm for the calculation of the roots of special functions.
SIAM J. Sci. Comput. (2007) A/O+I+**TS** Hermite, Laguerre ($\alpha = 0$), Legendre

J. Segura, Reliable computation of the zeros of solutions of second order linear ODEs using a fourth order method.
SIAM J. Numer. Anal. (2010) **□+I+?** Hermite, Laguerre, Jacobi (*)

I. Bogaert, B. Michiels, J. Fostier, J., O(1) computation of Legendre polynomials and Gauss-Legendre nodes and weights for parallel computing.
SIAM J. Sci. Comput. (2012) A+I+V Legendre

N. Hale, A. Townsend, Fast and accurate computation of Gauss-Legendre and Gauss-Jacobi quadrature nodes and weights.
SIAM J. Sci. Comput. (2013) A+I+A Jacobi

A. Townsend, The race to compute high-order Gauss-Legendre quadrature.
SIAM News (2015)

I. Bogaert, Iteration-free computation of Gauss-Legendre quadrature nodes and weights.
SIAM J. Sci. Comput. (2014) □+**A**+**A** Legendre

A. Townsend, T. Trogdon, S. Olver, Fast computation of Gauss quadrature nodes and weights on the whole real line.
IMA J. Numer. Anal (2016) A+I+A Hermite*

J. Bremer, On the numerical calculation of the roots of special functions satisfying second order ordinary differential equations.
SIAM J. Sci. Comput (2017) O Laguerre, Jacobi. Very high degree.

A. Gil, J. Segura, N. M. Temme Asymptotic approximations to the nodes and weights of Gauss-Hermite and Gauss-Laguerre quadratures.
Stud. Appl. Math. (2018) □+**A**+**A** Hermite, Laguerre

F. Johansson, M. Mezzarobba Fast and rigorous arbitrary-precision computation of Gauss-Legendre quadrature nodes and weights.
SIAM J. Sci. Comput (2019) A*+I+V Legendre

A. Gil, J. Segura, N. M. Temme Non-iterative computation of Gauss-Jacobi quadrature.
SIAM J. Sci. Comput. (2019) □+**A**+**A** Jacobi

A. Gil, J. Segura, N. M. Temme Fast, reliable and unrestricted iterative computation of Gauss–Hermite and Gauss–Laguerre quadratures.
Numer. Math. (accepted) □+**I**+**TS/CF** Hermite, Laguerre

In the previous list, there are a number of methods which combine asymptotic and convergent iterative methods. We prefer purely asymptotic methods and purely iterative (convergent methods). We argue that these are the best (and complementary) approaches:

1. Iterative-only methods with no asymptotic approximations lead to arbitrary precision algorithms provided that the computations are based on finite or convergent expansions.
2. Asymptotic-only methods can not be used for arbitary precision, but these are the fastest methods, and they can be accurate (15-16 digits) for moderately large degrees (as first proved for Gauss-Legendre in Bogaert's paper).

Methods which combine both iterative and asymptotic methods, may be accurate and efficient, but not so accurate as a purely iterative method and not as fast as a purely asymptotic method.

Asymptotic methods

Until recently, only the Gauss-Legendre quadrature was available with $15-16$ accuracy and moderately large n with asymptotic methods (Bogaert's paper). We have used different expansions in terms of other functions and zeros for computing accurately all the zeros and and weight for $n \geq 100$:

1. Hermite, in terms of: elementary functions and Airy functions.
2. Laguerre: two approximations in terms of Bessel functions and one with Airy functions.
3. Jacobi, in terms of: elementary functions and Bessel functions.

We only describe one of the simplest expansions (which is new): the elementary expansion for the Jacobi case, which can be used for computing most of the nodes and weights.

Notice: the Bogaert expansion for Gauss-Legendre is in terms of Bessel functions, and our elementary expansion for the more general Jacobi case is simpler but quite powerful.

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

where $\gamma = \frac{1}{2}(\alpha+\beta+1)$, $\kappa = n+\gamma$, and $\phi(z) = \ln(z-x) - \ln(1-z^2)$.

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

$$\chi = \kappa\theta + \left(\alpha+\frac{1}{2}\right)\frac{\pi}{2},\ x = \cos\theta.$$

with expansions

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

The first coefficients are

$$u_0(x) = 1, \quad v_1(x) = \frac{2\alpha^2 - 2\beta^2 + (2\alpha^2+2\beta^2-1)x}{8\sin\theta},$$

$$u_2(x) = \frac{1}{384\sin^2\theta}(12(5-2\alpha^2-2\beta^2)(\alpha^2-\beta^2)x \ + 4(-3(\alpha^2-\beta^2)^2 + 3(\alpha^2+\beta^2) - 6 + 4\alpha(\alpha^2-1+3\beta^2) +$$
$$(-12(\alpha^2+\beta^2)(\alpha^2+\beta^2-1) - 16\alpha(\alpha^2-1+3\beta^2) - 3)x^2).$$

For computing the weights we also need $\dot{y}(\theta)$.

Let see how to compute the nodes. Let $W(\theta) = \cos\chi\, U(x) - \sin\chi\, V(x)$,

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

A first approximation is $\cos\chi = 0 \to \theta = \theta_0 = \left(n - k + \frac{3}{4} + \frac{\alpha}{2}\right)\frac{\pi}{\kappa}$.

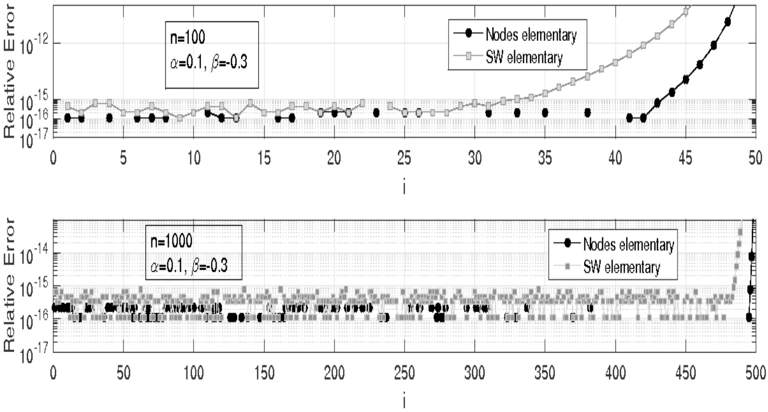
Now we write $W(\theta) = W(\theta_0 + \epsilon) = W(\theta_0) + \epsilon \dot{W}(\theta_0) + \frac{\epsilon^2}{2}\ddot{W}(\theta) + \ldots = 0$, we assume the expansion $\epsilon = \frac{\theta_1}{\kappa^2} + \frac{\theta_2}{\kappa^3} + \frac{\theta_3}{\kappa^4} + \ldots$, and using the expansions of U and V and comparing equal powers of κ we determine the coefficients θ_i, $i \geq 1$, (depending on θ_0). The first two cofficients are:

$$\theta_1 = -\frac{1}{8\sin\theta_0}\left(2\beta^2 x + 2\alpha^2 x - x - 2\beta^2 + 2\alpha^2\right),$$

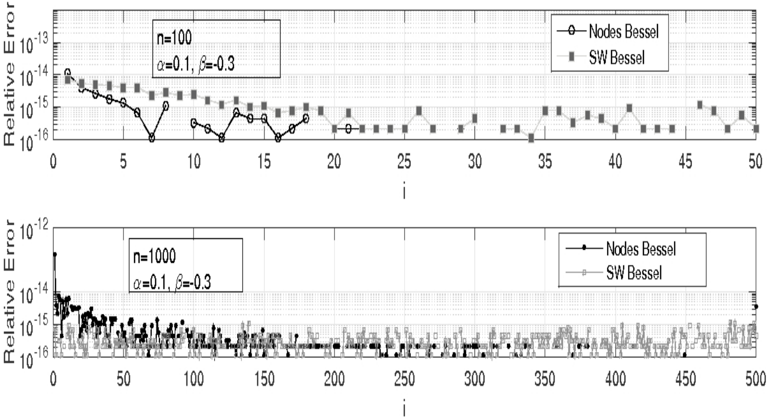
$$\begin{aligned}\theta_2 = \frac{1}{384\sin^3\theta_0}(&-33x - 36\beta^2x^2 + 36\alpha^2x^2 + 24\beta^4x^2 - 24\alpha^4x^2 + 2x^3 + \\ &84\beta^2x - 60\alpha^4x - 60\beta^4x + 84\alpha^2x + 4\beta^4x^3 + 4\alpha^4x^3 - 8\beta^2x^3 + \\ &40\alpha^2 - 8\alpha^2x^3 - 40\beta^2 + 32\beta^4 - 32\alpha^4 + 24\alpha^2\beta^2x^3 - 24\alpha^2\beta^2x),\end{aligned}$$

where $x = \cos\theta_0$.

With this we compute $\theta = \theta_0 + \epsilon$ and then $x_k \sim \cos\theta$



Relative Error
n=100
α=0.1, β=-0.3
Nodes elementary
SW elementary
i
n=1000
α=0.1, β=-0.3
Nodes elementary
SW elementary
i


n=100
α=0.1, β=-0.3
Nodes Bessel
SW Bessel
Relative Error
i
n=1000
α=0.1, β=-0.3
Nodes Bessel
SW Bessel
Relative Error
i

For more on classical Gaussian quadratures from an asymptotic point of view:

A. Gil, JS, N. M. Temme Asymptotic approximations to the nodes and weights of Gauss-Hermite and Gauss-Laguerre quadratures. **Stud. Appl. Math. (2018)** **□+A+A**

A. Gil, JS, N. M. Temme Non-iterative computation of Gauss-Jacobi quadrature. **SIAM J. Sci. Comput. (2019)** **□+A+A**

A. Gil, JS, N. M. Temme Asymptotic expansions of Jacobi polynomials and of the nodes and weights of Gauss-Jacobi quadrature for large degree and parameters in terms of elementary functions. **Submitted** **□+A+A**

Iterative computation of classical Gauss quadratures

Two remarks regarding iterative methods in the literature:

○ Most papers use Newton's method for computing the roots (order 2):

Newton's method:

The NM, $x_{n+1} = x_n - \dfrac{f(x_n)}{f'(x_n)}$, has order of convergence 2 because

$$\frac{\epsilon_{n+1}}{\epsilon_n^2} = \frac{f''(\alpha)}{2f'(\alpha)} + \mathcal{O}(\epsilon_n) \text{ as } n \to \infty,\ \epsilon_n = x_n - \alpha.$$

○ The only proof of convergence for Newton method is for the Legendre case (Petras, 1999).

For computing zeros of solutions of

$$w''(x) + B(x)w'(x) + C(x)w(x) = 0$$

Newton method gives **order** 2 generally. **But the ODE can be used to speed-up the method.**
Assuming that $B(x)$ is differentiable we can transform (16) by setting

$$y(x) = \exp\left(\int \frac{1}{2}B(x)dx\right) w(x)$$

Then, $y''(x) + A(x)y(x) = 0$, with $A(x) = C(x) - \frac{1}{2}B'(x) - \frac{1}{4}B(x)^2$ and

$$\frac{y(x)}{y'(x)} = \frac{w(x)}{\frac{1}{2}B'(x)w(x) + w'(x)}$$

The Newton method $x_{n+1} = x_n - \dfrac{y(x_n)}{y'(x_n)}$ is now of third order.

The reason: if α is such that $y(\alpha) = 0$, then $y''(\alpha) = 0$.
And we haven't used $A(x)$ so far...

We define the following fixed point iteration (JS, SIAM J. Numer. Anal, 2010):

Let $j = \text{sign}(A'(x))$, we define

$$T(x) = x - \frac{1}{\sqrt{A(x)}} \text{arctan}_j(\sqrt{A(x)}h(x))$$

with

$$\text{arctan}_j(\zeta) = \begin{cases} \arctan(\zeta) \text{ if } jz > 0, \\ \arctan(\zeta) + j\pi \text{ if } jz \leq 0, \\ j\pi/2 \text{ if } z = \pm\infty \end{cases}$$

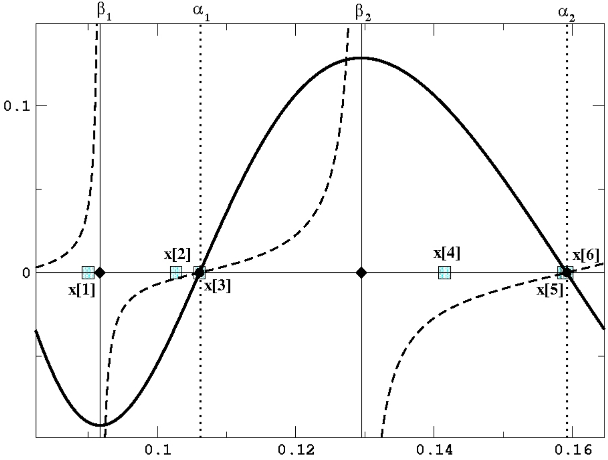
This method converges to α for any x_0 in $[\alpha', \alpha)$ if $A'(x) < 0$, with α' the largest zero smaller than α (analogously for $A'(x) > 0$).

The method has fourth order convergence:

$$\epsilon_{n+1} = \frac{A'(\alpha)}{12}\epsilon_n^4 + \mathcal{O}(\epsilon_n^5),\ \epsilon_k = x_k - \alpha$$

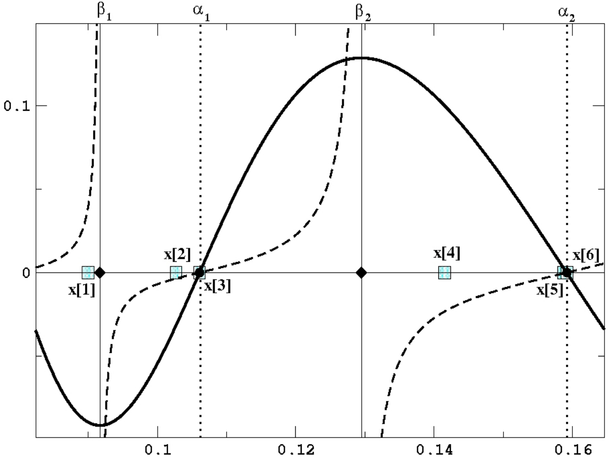
Computing the zeros in an interval where $A(x)$ is monotonic.

Example: zeros of $y(x) = x\sin(1/x)$, satisfying $y''(x) + x^{-4}y(x) = 0$ (4 digits of acc.).



Computing the zeros in an interval where $A(x)$ is monotonic.

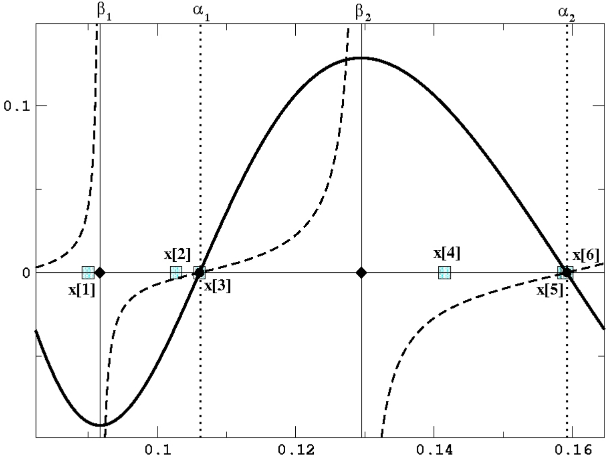
Example: zeros of $y(x) = x\sin(1/x)$, satisfying $y''(x) + x^{-4}y(x) = 0$ (4 digits of acc.).



1. $T(x[1]) = x[2]$, $T(x[2]) = x[3]$ (with four digits acc.)

Computing the zeros in an interval where $A(x)$ is monotonic.

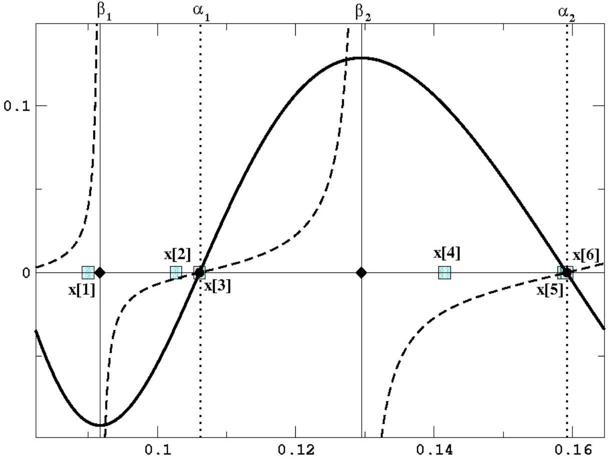
Example: zeros of $y(x) = x\sin(1/x)$, satisfying $y''(x) + x^{-4}y(x) = 0$ (4 digits of acc.).



1. $T(x[1]) = x[2]$, $T(x[2]) = x[3]$ (with four digits acc.)
2. $x[4] = x[3] + \pi/\sqrt{A(x[3])}$ (smaller than the next zero by Sturm comparison)

Computing the zeros in an interval where $A(x)$ is monotonic.

Example: zeros of $y(x) = x\sin(1/x)$, satisfying $y''(x) + x^{-4}y(x) = 0$ (4 digits of acc.).



1. $T(x[1]) = x[2]$, $T(x[2]) = x[3]$ (with four digits acc.)
2. $x[4] = x[3] + \pi/\sqrt{A(x[3])}$ (smaller than the next zero by Sturm comparison)
3. $T(x[4]) = x[5]$, $T(x[5]) = x[6]$ (with four digits acc.)

1. Guaranteed convergence. Does not require initial estimates.
2. Fourth order convergence (Newton has order 2)
3. Same computational cost per iteration as Newton!
4. And more: it will be asymptotically exact if a convenient variable is chosen.

Gauss-Hermite quadrature

We have that $y_n(x) = C_n e^{-x^2/2} H_n(x)$ satisfies

$$y_n''(x) + A(x) y_n(x) = 0,\ A(x) = 2n + 1 - x^2.$$

$A(x)$ has its maximum at $x = 0$. The nodes are symmetric around the origin.

We compute the positive roots in the direction of decreasing $A(x)$, starting at $x = 0$ until we have computed $\lfloor n/2 \rfloor$ zeros.
The first step is

$$x = T_{-1}(0) = \begin{cases} \dfrac{\pi}{\sqrt{2n+1}}, & n \text{ odd} \\ \dfrac{\pi}{2\sqrt{2n+1}}, & n \text{ even} \end{cases}$$

As $n \to +\infty$ the coefficient $A(x)$ is essentially constant for not too large x. In this sense, the method will be asymptotically exact.

In the next steps we need to **compute the polynomials**. How to do that?

We prefer to avoid asymptotics so that arbitary accuracy for any degree is available.
The three-term recurrence relation IS NOT a good idea, particularly for large degree.
A good alternative: use **local Taylor series** (as done in Glaser, Liu, Rokhlin (2007)):

Given $y(x) = C_n e^{-x^2/2} H_n(x)$, and assuming that the derivatives at x_0 are known:

$$\boldsymbol{y(x) = \sum_{k=0}^{\infty} \frac{y^{(k)}(x_0)}{k!}(x - x_0)^k.}$$

and similarly for $y'(x)$, truncating the series for a given precision.

From $y(x_0)$ and $y'(x_0)$, we compute the sucessive derivatives by differentiating $y''(x) + (2n + 1 - x^2)y(x) = 0$:

$$\boldsymbol{y^{(k+2)} + (2n + 1 - x^2)y^{(k)} - 2kxy^{(k-1)} - k(k-1)y^{(k-2)} = 0.}$$

Perron-Kreuser theorem: all the solutions of the difference equation are such that $\boldsymbol{\limsup_{k\to+\infty} \left(|y^{(k)}|/(k!)^{2/3}\right)^{1/k} = 1}$ (the series converges everywhere, as expected).

$h = x - x_0$ will be always less than the maximal distance between zeros of $H_n(x)$.

Algorithm for Gauss-Hermite based on local Taylor series.
Take $y(x) = (2^n n!)^{-1/2} e^{-x^2/2} H_n(x)$.

1. With $x = \pi/\sqrt{2n+1}$ for n odd and $x = \pi/(2\sqrt{2n+1})$ for n even, compute $y(x)$ and $y'(x)$ by Taylor series starting from the known values $y(0)$ and $y'(0)$ **(alternatively, we can take $y(0) = 1$ and $y'(0) = 0$ for n even and $y(0) = 0$ and $y'(0) = 1$ for n odd, and rescale later)**. Let $i = 1$.
2. Iterate the fixed point method

$$T(x) = x - \frac{1}{\sqrt{A(x)}} \mathrm{arctan}_{-1}(\sqrt{A(x)}h(x))$$

until convergence is reached within a given accuracy. The values of $y(x)$ and $y'(x)$ are computed from Taylor series, starting with the values at the previous point. Let x_i be the resulting zero (node)

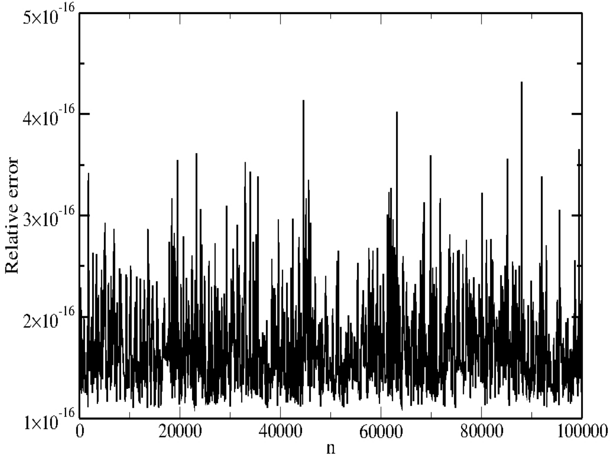
3. The corresponding weight is given by $w_i = 2e^{-x_i^2}/(y_n'(x_i))^2$.
4. Set $x = x_i + \pi/\sqrt{A(x_i)}$, $i = i + 1$, and if additional nodes have to be computed then go to 2.

If rescaling is used, we just have to take into account that the sum of all the weights is $\sqrt{\pi}$.

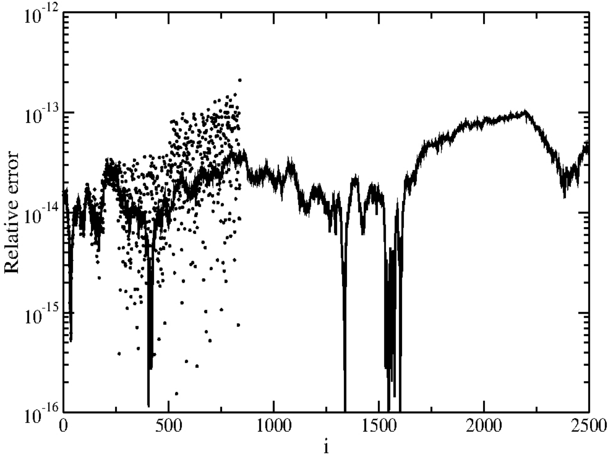
Some (nice) features of the algorithm

1. The cost of local Taylor series does not increase with the degree.
2. The cost of computing each zero decreases as $n \to \infty$ (asymptotic exactness)
3. Because of the fourth order convergence and the error relation, when two succesive iterates are such that $|x^{(n+1)} - x^{(n)}| < 6^{1/4}\, 10^{-p/4}$ then we can estimate that $|x_{n+1}/\alpha - 1| < 10^{-p}$.
4. The so-called scaled weights $\omega_i = 1/y'(x_i)^2$ are well-conditioned as a function of x_i because $\omega_i = W(x_i)$ with $W'(x_i) = 0$.
5. The code is short ($<$ 100 lines), reliable and efficient.

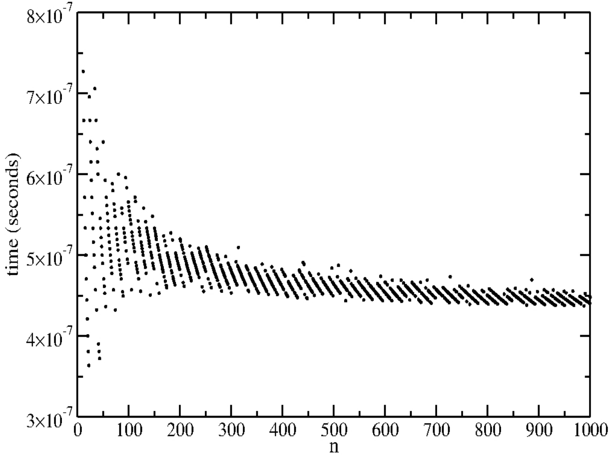
The method does not need initial estimations for the roots. It seems they don't improve significantly the performance (typical running time: 0.5 seconds for 10^6 nodes in my laptop).



Maximum relative error in the computation of the nodes as a function of the degree n



Relative error in the computation of scaled weights (solid line) and unscaled weights (dots) for the 5000-point Gauss Hermite formula as a function of i, with i numbering the positive nodes in increasing order.



Unitary time spent (time per node and corresponding weight) in seconds as a function of the degree

Gauss-Laguerre quadrature

Take $y(z)=z^{\alpha+1/2}e^{-z^2/2}L_n^{(\alpha)}(z^2)$ which satisfies $\ddot{\boldsymbol{y}}(\boldsymbol{z})+\boldsymbol{A}(\boldsymbol{z})\boldsymbol{y}(\boldsymbol{z})=\mathbf{0}$ with

$$\boldsymbol{A(z(x))=-x+2(2n+\alpha+1)+\frac{\frac{1}{4}-\alpha^2}{x},\ x=z^2}$$

$A(x)$ for $x>0$: decreasing if $|\alpha|\leq 1/2$.
with a maximum at $x_e=\sqrt{\alpha^2-1/4}$ if $|\alpha|>1/2$.

Depence of $A(x)$ on $n\rightarrow$ asymptotic exactness.

The Gauss-Laguerre weights are

$$w_i=\frac{4\Gamma(n+\alpha+1)}{n!(\dot{y}(z_i))^2}x_i^{\alpha+\frac{1}{2}}e^{-x_i}\equiv \omega_i x_i^{\alpha+\frac{1}{2}}e^{-x_i}.$$

Scaled weights ω_i : **well conditioned as a function of the nodes**
($\omega_i=W(z_i)$ with $\dot{W}(z_i)=0$)

The computation of $y(z)$ for the Laguerre case is not so easy as for Hermite.

Taylor series must be supplemented with an additional starting method.
A good choice is the continued fraction which follows from iterating

$$r^{(\alpha)} = \frac{a_\alpha}{b_\alpha + r^{(\alpha+1)}},$$

where $r^{(\alpha)} = L_n^{(\alpha)}(x)/L_n^{(\alpha-1)}(x)$, $b_\alpha = -(1+\alpha/x)$, $a_\alpha = -(n+\alpha)/x$.
Using the derivative rule

$$xL_n^{(\alpha)\prime}(x) = -\alpha L_n^{(\alpha)}(x) + (\alpha+n)L_n^{(\alpha-1)}(x)$$

we get

$$\frac{\dot{y}(z)}{y(z)} = \frac{1/2-\alpha}{z} - z + \frac{2(n+\alpha)}{zr^{(\alpha)}(z^2)}.$$

with $y(z)$ as defined before.

We only need this for initiating Taylor series (the ratio is enough, because we can rescale with the moment of order zero).

Taylor series

The function $y(z)$ satisfies

$$P(z)y^{(2)}(z) + Q(z)y(z) = 0,$$

with

$$P(z) = z^2, \quad Q(z) = -z^4 + 2Lz^2 + \frac{1}{4} - \alpha^2$$

taking successive derivatives and using that $P^{(n)}(z) = 0$, $n > 2$ and $Q^{(n)}(z) = 0$, $n > 4$, we obtain the following recursion formula for the derivatives with respect to z:

$$\sum_{m=0}^{2} \binom{j}{m} P^{(m)}(z)y^{(j+2-m)}(z) + \sum_{m=0}^{4} \binom{j}{m} Q^{(m)}(z)y^{(j-m)}(z) = 0,$$

where $\binom{j}{m}$ are binomial coefficients.

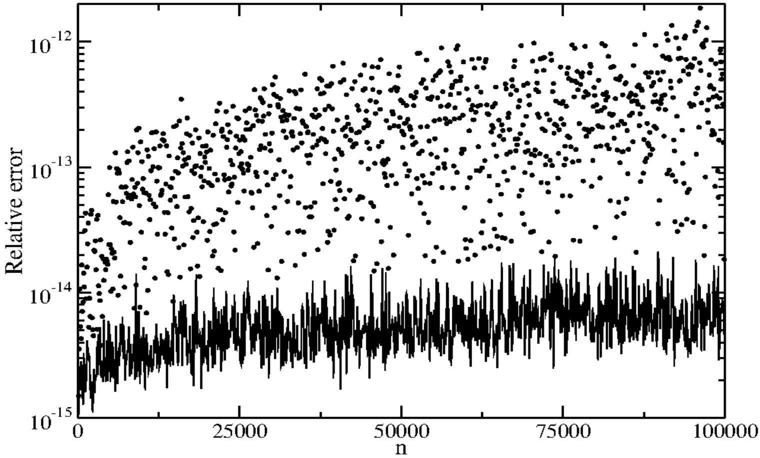
Perron-Kreuser $\Longrightarrow$ well-conditioned recurrence.

The ingredients for the method for Gauss-Laguerre are:

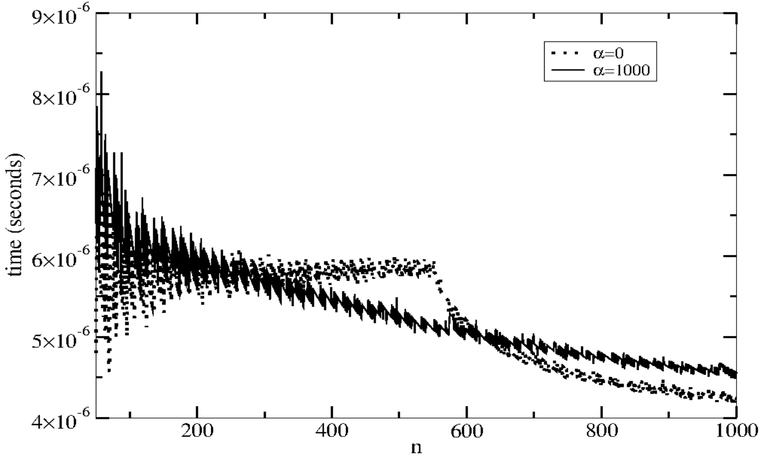
1. The continued fraction as a starting value (in some cases two values are required)
2. The fourth order fixed point method in the z variable
3. Taylor series in the z variable

In our previous notation, this is **□+I+TS/CF**

Some numerical results follow



Maximum relative errors in the computation of the nodes for n-point Gauss–Laguerre quadrature with $\alpha = 0$ (dots). We also show the maximum error when it is evaluated only for the nodes for which the weights are larger than 10^{-30} (solid line); the error in this case is smaller.



Unitary CPU-time spent as a function of the degree n for Gauss-Laguerre with $\alpha = 0$ and $\alpha = 1000$

Gauss-Jacobi quadrature

The Jacobi equation satisfied by $y(x) = P_n^{(\alpha,\beta)}(x)$ reads

$$(1-x^2)y''(x) + [(1+\beta)(1-x) - (1+\alpha)(1+x)]\, y'(x) + n(n+\alpha+\beta+1)y(x) = 0$$

The transformed function $Y_x(x) = (1-x)^{(\alpha+1)/2}(1+x)^{(\beta+1)/2}y(x)$ is a solution of

$$\begin{aligned} &Y_x''(x) + A_x(x)Y_x(x) = 0,\\ &A_x(x) = \frac{1}{4(1-x^2)^2}\left[(L^2-1)(1-x^2) - 2(\alpha^2-1)(1+x) - 2(\beta^2-1)(1-x)\right] \end{aligned}$$

This trivial change is good for computing with Taylor series, but the monotonicity properties of $A_x(x)$ are not easy to handle.

Changes of variable which easen the monotonicity properties are studied Deaño, Gil, JS (2004) and Deaño, JS (2007). Two main changes are considered in the algorithms:

Considering "canonical" change of variable $x = \cos\theta$ and
$Y_\theta(\theta) = \left(\sin\frac{1}{2}\theta\right)^{\alpha+1/2}\left(\cos\frac{1}{2}\theta\right)^{\beta+1/2} y(\cos\theta)$:

$$\frac{d^2 Y_\theta}{d\theta^2} + A_\theta(\theta) Y_\theta = 0,\ A_\theta(\theta) = \frac{1}{4}\left[L^2 + \frac{\frac{1}{4}-\alpha^2}{\sin^2(\theta/2)} + \frac{\frac{1}{4}-\beta^2}{\cos^2(\theta/2)}\right]$$

$L = 2n + \alpha + \beta + 1$

In this variable the method is asymptotically exact as $n \to +\infty$.

1. If $|\alpha| < 1/2$ and $|\beta| < 1/2$, $A(\theta)$ has a minimum at $x_m = (s(\beta) - s(\alpha))/(s(\beta) + s(\alpha))$, $s(z) = \sqrt{|1/4 - z^2|}$.
2. If $|\alpha| > 1/2$ and $|\beta| > 1/2$, $A(\theta)$ has a maximum at x_m.
3. Otherwise $A(\theta)$ is monotonic.

Taylor series in the θ variable are harder, because the derivatives do not satisfy a recurrence relation with a fixed number of terms.

With the change $x = \tanh z$ the transformed function

$$Y_z(x) = (1-x)^{\alpha/2}(1+x)^{\beta/2}P_n^{(\alpha,\beta)}(x) \tag{1}$$

satisfies the equation $\ddot{Y}_z(z) + A_z(z)Y_z(z) = 0$ with

$$A_z(z(x)) = \frac{1}{4}\left[(L^2-1)(1-x^2) - 2\alpha^2(1+x) - 2\beta^2(1-x))\right]$$

The coefficient has a maximum at $x_e = \dfrac{\beta^2-\alpha^2}{L^2-1}$

For $\alpha = \beta$ (Gauss-Gegenbauer quadrature, and Gauss-Legendre in particular), the method starts at $z = 0$, and an Hermite-like algorithm is possible.

In the general case, we have a Laguerre-type algorithm in the z-variable, and one initiator method will be required at $z = z(x_e) = \tanh(x_e)$ (we use the recurrence over n for $P_n^{(\alpha,\beta)}(x)$.

Same as for the θ variable, Taylor series in terms of z are not a good idea. So, what then?

We combine computations in the three variables:

1. The backbone of the method is the fixed point method in the variable z: $z^{(k+1)} = T(z^k)$, with

$$T(z) = z - \frac{1}{\sqrt{A_z(z)}} \arctan_j\left(\sqrt{A_z(z)}h_z(z)\right),\ h_z(z) = Y_z(z)/\dot{Y}_z(z).$$

   We however, compute this in the x-variable, using $z = \tanh x$, that is:

$$x^{(k)} = g(x^{(k)}) \equiv \frac{x^{(k)} - \tanh(F(x^{(k)}))}{1 - x^{(k)}\tanh(F(x^{(k)}))},$$

$$F(x) = \frac{1}{\sqrt{A_z(z(x))}} \arctan_j\left(\sqrt{A_z(z(x))}\frac{Y_x(x)}{(1-x^2)Y_x'(x) + xY_x(x)}\right)$$

2. The functions Y_x and Y_x' are computed by Taylor series in the x variable, with an initiation (three-term recurrence) for the nonsymmetric case $\alpha \neq \beta$.
3. Close to the singularities of the Jacobi equation $x = \pm 1$ Taylor series tend to fail, and it is better to use and alternative representation (continued fraction) and to compute with the fixed point method associated to the θ variable.

Some preliminary impressions on this preliminary algorithm:

1. As happened with Hermite and Laguerre, the algorithm appears to have no practical restrictions (compared for instance with chebfun where $\alpha, \beta < 5$ and with GW which is unpractical for moderately high orders).
2. The nonlinear method converges with certainty and with fourth order.
3. The method is based on convergent processes and it can be used in arbitrary accuracy computations.
4. The computations are simple (arctan and tanh are the most complicated functions appearing).
5. It is accurate and very fast.

TIMING

n	α	β	Our method (s)	chebfun (s)	Speed–up
10	0	0	8×10^{-6}	4×10^{-6}	~ 0.5
10	1	1	6×10^{-6}	4×10^{-6}	~ 0.6
10	-0.4	1	8×10^{-6}	8×10^{-6}	~ 1
10	-0.99	5	8×10^{-6}	9×10^{-6}	~ 1
100	-0.7	-0.7	4×10^{-6}	5×10^{-5}	~ 10
100	0	0	4×10^{-6}	2×10^{-6}	~ 0.5
100	1	1	4×10^{-6}	4×10^{-5}	~ 10
100	-0.4	1	4×10^{-6}	1×10^{-4}	~ 25
100	-0.99	5	4×10^{-6}	2×10^{-4}	~ 50
1000	-0.7	-0.7	3×10^{-6}	9×10^{-6}	~ 3
1000	0	0	2×10^{-6}	3×10^{-7}	~ 0.1
1000	1	1	3×10^{-6}	8×10^{-6}	~ 2.5
1000	-0.4	1	3×10^{-6}	2×10^{-5}	~ 1
1000	-0.99	5	3×10^{-6}	3×10^{-5}	~ 10

Time per node in seconds.

THANK YOU!