Recent software developments for special functions in the Santander-Amsterdam project

Amparo Gil

Departamento de Matemática Aplicada y CC. de la Computación.
ETSI Caminos. Universidad de Cantabria. 39005-Santander, Spain.

Javier Segura

Departamento de Matemáticas, Estadística y Computación,
Univ. de Cantabria, 39005 Santander, Spain.

Nico M. Temme

IAA, 1391 VD 18, Abcoude, The Netherlands[*]

e-mail: amparo.gil@unican.es, javier.segura@unican.es, Nico.Temme@cwi.nl

Abstract

We give an overview of published algorithms by our group and of current activities and future plans. In particular, we give details on methods for computing special functions and discuss in detail two current lines of research. Firstly, we describe the recent developments for the computation of central and non-central χ-square cumulative distributions (also called Marcum $Q-$functions), and we present a new quadrature method for computing them. Secondly, we describe the fourth-order methods for computing zeros of special functions recently developed, and we provide an explicit example for the computation of complex zeros of Bessel functions. We end with an overview of published software by our group for computing special functions.

Keywords & Phrases: special functions; incomplete gamma functions, Marcum's $Q-$function; zeros of special functions; algorithms; numerical software.

---

[*]Former address: CWI, Science Park 123, 1098 XG Amsterdam, The Netherlands

1   Introduction

Our project on software developments for special functions started in 1997. Earlier we published a number of papers in this area, and the plan was to combine all our expertise in order to produce quality software based on a selection of the many methods available for special functions, and by justifying these methods in particular cases with the help of elements of numerical and mathematical analysis, such as recurrence relations, numerical quadrature and asymptotic expansions.

We discuss current activities in two different lines of research. Firstly, we discuss the computation of χ-square cumulative distributions and, in particular, we present a new quadrature method for computing the non-central distribution, also called Marcum's $Q-$function. Secondly, we describe in a unified way two recent methods for computing zeros of special functions (for real and complex zeros), we give an explicit example of computation for complex zeros of Bessel functions and discuss plans for software implementations. We end with an overview of our published software for computing special functions.

First, we briefly outline the basic methods and principles we consider in the construction of special function software.

2   Numerical methods and basic principles

Our book [33] "Numerical methods for special functions" appeared in 2007 and describes four basic methods that we have used in writing software. These methods are: convergent and asymptotic series, Chebyshev expansions, linear recurrence relations and quadrature methods. The book also describes numerical methods for computing continued fractions, methods for computing zeros of special functions and the computation with uniform asymptotic expansions, among other topics. Usually, several of these methods have to be combined for computing a special function.

2.1   Our principles of designing algorithms

Our approach in making software can be described by the following principles.

1. The main objective is to develop Fortran 90 codes which produce reliable double precision values. We use Maple or Mathematica for obtaining coefficients in expansions and for verifying algorithms, but not

usually in the final product. The exceptions are some codes for computing zeros of special functions.

2. A given special function is usually a special case of a more general function. Our approach is bottom-up and when a simple but important function is demanding software, we prefer to start with the "simple" case. For example: Airy functions are special cases of the more general Bessel functions, but we have written codes for the Airy functions themselves.

3. We accept that it is necessary to combine several methods in order to compute a function accurately and efficiently for a wide range of its variables.

4. We accept that a theoretical error analysis is usually impossible for functions with several real or complex variables. We accept more empirical approaches.

5. The accuracy analysis is usually done by using functional relations, such as Wronskian relations or by comparing with an alternative method of computation.

6. The selection of methods in different parameter domains is based on speed and accuracy, where the latter may prevail. For large real or complex parameters scaling of the result is useful to avoid underflow or overflow in our finite arithmetic environment.

3 Incomplete gamma functions, Marcum $Q-$function

As before commented, present interest is focused on distribution functions, in particular on the incomplete gamma functions (see [38]) and generalizations. Incomplete gamma functions are the central χ-square cumulative distributions and are defined by

$$\gamma(a,x) = \int_0^x t^{a-1}e^{-t}\,dt, \quad \Gamma(a,x) = \int_x^\infty t^{a-1}e^{-t}\,dt. \qquad (3.1)$$

We concentrate on the ratios

$$P(a,x) = \frac{1}{\Gamma(a)}\gamma(a,x), \quad Q(a,x) = \frac{1}{\Gamma(a)}\Gamma(a,x), \qquad (3.2)$$

where we assume that a and x are positive. We can use the well-known series expansions, asymptotic expansions, recurrence relations, and a method

based on the uniform asymptotic representation of these ratios in terms of the complementary error function. Apart from the last method, we mainly use the methods considered in [14, 15], although Gautschi used a different set of functions and also negative values of a. For applications in mathematical statistics and probability theory the ratios in (3.2) are more relevant. Furthermore, [38] in we described algorithms for inverting the incomplete gamma ratios, and the algorithm improves the one given in [8, 9].

The results of the algorithms for the incomplete gamma ratios are used in our current project on the the generalized Marcum $Q-$function, which is defined in (3.5) and (3.6). A paper with full details of our numerical computations has been submitted [40]. The relation with the incomplete gamma functions becomes clear when expanding the Bessel function in its power series, which gives

$$Q_\mu(x, y) = e^{-x} \sum_{n=0}^{\infty} \frac{x^n}{n!} Q(\mu + n, y), \tag{3.3}$$

and

$$P_\mu(x, y) = e^{-x} \sum_{n=0}^{\infty} \frac{x^n}{n!} P(\mu + n, y). \tag{3.4}$$

3.1 A new quadrature method for computing the Marcum Q-function

As we have explained in [33, Chapter 5], numerical quadrature can be an important tool for evaluating special functions. In particular, when selecting suitable integral representations for these functions. In Chapter 12 of our book we have given several examples, see also §§5.2, 5.3, 5.4. The quadrature methods are usually based on the simple trapezoidal rule, that is very efficient and accurate for certain integral representations that follow from contour integrals in the complex plane, for example through the saddle point of the integrand. We have shown for many cases that the trapezoidal rule is an effective tool for a large domain of the parameters. Usually we use the method to overlap power series methods near the origin and asymptotic methods for large parameters.

In this section we describe the trapezoidal method for a special case, the Marcum $Q-$function, and we give all analytical details that are needed for including the method in a computer program. In fact, this is generally the way of working: we need quite some analytical preparations before we apply the simple trapezoidal rule, but the extra work is usually very rewarding.

We define the generalized Marcum $Q-$function by using the integral representation

$$Q_\mu(x,y) = x^{\frac{1}{2}(1-\mu)} \int_y^{+\infty} t^{\frac{1}{2}(\mu-1)} e^{-t-x} I_{\mu-1}\left(2\sqrt{xt}\right) dt, \qquad (3.5)$$

where $x \geq 0$, $y \geq 0$, $\mu > 0$ and $I_\mu(z)$ is the modified Bessel function. We also use the complementary function

$$P_\mu(x,y) = x^{\frac{1}{2}(1-\mu)} \int_0^y t^{\frac{1}{2}(\mu-1)} e^{-t-x} I_{\mu-1}\left(2\sqrt{xt}\right) dt, \qquad (3.6)$$

and the complementary relation reads

$$P_\mu(x,y) + Q_\mu(x,y) = 1. \qquad (3.7)$$

The generalized Marcum $Q-$function, which for $\mu = 1$ reduces to the ordinary Marcum function, is used in problems on radar detection and communications; see [43]. In this field, μ is the number of independent samples of the output of a square-law detector. In our analysis μ is not necessarily an integer number. There is an extensive literature regarding this function, also from the areas of statistics and probability theory, where they are called the non central χ-square or the non central gamma distribution. For papers concentrating on numerical aspects we refer to [2, 3, 10, 41, 42, 46, 47, 52]. As explained earlier in this section, the central gamma distribution is in fact the incomplete gamma function.

The integrals in (3.5) and (3.6) give stable integral representations, but we prefer a representation in terms of elementary functions. Also, one important point for applying the trapezoidal rule efficiently is the vanishing of the integrand and many (or all) of its derivatives at the finite endpoint of integration.

In the present case we start with the complex integral representation (see [54, Eq. (2.3)])

$$Q_\mu(x,y) = \frac{e^{-x-y}}{2\pi i} \int_{\mathcal{L}_Q} \frac{e^{x/s+ys}}{1-s} \frac{ds}{s^\mu}, \qquad (3.8)$$

where $\mathcal{L}_Q$ is a vertical line that cuts the real axis in a point s_0, with $0 < s_0 < 1$. For the complementary function we have

$$P_\mu(x,y) = \frac{e^{-x-y}}{2\pi i} \int_{\mathcal{L}_P} \frac{e^{x/s+ys}}{s-1} \frac{ds}{s^\mu}, \qquad (3.9)$$

5

now with a vertical line $\mathcal{L}_P$ that cuts the real axis at a point s_0 with $s_0 > 1$. Both representations follow from each other by shifting the contour across the pole at $s = 1$ and picking up the residue.

We use scaled variables x, y and write the representation in (3.8) in the form

$$Q_\mu(\mu x, \mu y) = \frac{e^{-\mu(x+y)}}{2\pi i} \int_{\mathcal{L}_Q} \frac{e^{\mu\phi(s)}}{1-s} \, ds, \qquad (3.10)$$

where $\mathcal{L}_Q$ is as in (3.8), and

$$\phi(s) = \frac{x}{s} + ys - \ln s. \qquad (3.11)$$

The first step in choosing a suitable contour for numerical quadrature is determining the saddle point of $\phi(s)$ and shifting the contour through this saddle point. We have

$$\phi'(s) = -\frac{x}{s^2} + y - \frac{1}{s}, \qquad (3.12)$$

which vanishes at the point (the negative zero is not relevant here)

$$s_0 = \frac{1 + \sqrt{1 + \xi^2}}{2y}, \qquad \xi = 2\sqrt{xy}. \qquad (3.13)$$

We want to evaluate the $Q-$function when $y > x+1$ (in the scaled variables). When $y < x + 1$ we would compute the $P-$function instead and use (3.7).

We continue with $y > x + 1$ with $\mathcal{L}_Q$ through s_0 of (3.13). It is easy to verify that in that case the saddle point s_0 satisfies $0 < s_0 < 1$. The next step is to deform this line into a contour (still through s_0) on which $\Im\phi(s) = \Im\phi(s_0)$. This gives the equation

$$\Im\left(\frac{x}{r}e^{-i\theta} + yre^{i\theta} - \ln r - i\theta\right) = 0, \qquad (3.14)$$

because $\Im\phi(s_0) = 0$. Here we have used polar coordinates

$$s = re^{i\theta}, \quad r > 0, \quad -\pi < \theta < \pi. \qquad (3.15)$$

Solving for r we obtain

$$r(\theta) = \frac{1}{2y}\left(\frac{\theta}{\sin\theta} + \rho(\theta,\xi)\right), \quad \rho(\theta,\xi) = \sqrt{\left(\frac{\theta}{\sin\theta}\right)^2 + \xi^2}. \qquad (3.16)$$

This defines a parabola-shaped contour through s_0 given in (3.13) and extending to $\Re s \to -\infty$ (when $\theta \to \pm\pi$).

6

On this contour we have

$$\phi(s) = \Re\phi(s) = \cos\theta\,\rho(\theta,\xi) - \ln r(\theta). \qquad (3.17)$$

Integrating on the contour with respect to θ gives

$$Q_\mu(\mu x, \mu y) = \frac{e^{-\mu(x+y)}}{2\pi i} \int_{-\pi}^{\pi} e^{\mu\Re\phi(s)} \frac{r'(\theta) + ir}{e^{-i\theta} - r}\, d\theta, \qquad (3.18)$$

and by evaluating the fraction in the integrand we obtain the real representation

$$Q_\mu(\mu x, \mu y) = \frac{e^{-\mu(x+y)}}{2\pi} \int_{-\pi}^{\pi} e^{\mu\Re\phi(s)} f(\theta)\, d\theta, \qquad (3.19)$$

where $f(\theta)$ is an even function (the odd part does not contribute), and is defined by

$$f(\theta) = \frac{\sin\theta\, r'(\theta) + (\cos\theta - r(\theta))\, r(\theta)}{r^2(\theta) - 2r(\theta)\cos\theta + 1}. \qquad (3.20)$$

This representation has the required form: the integrand vanishes with all its derivatives at the endpoints of the interval. But this is not the final representation. The point is, we like to take out the value $\exp(\mu\Re\phi(s_0))$ at $\theta = 0$ in order to avoid numerical instabilities for small values of θ. We write

$$-x - y + \phi(s_0) = -\tfrac{1}{2}\zeta^2 \qquad (3.21)$$

and we obtain our final representation

$$Q_\mu(\mu x, \mu y) = \frac{e^{-\frac{1}{2}\mu\zeta^2}}{2\pi} \int_{-\pi}^{\pi} e^{\mu\psi(\theta)} f(\theta)\, d\theta, \qquad (3.22)$$

where

$$\psi(\theta) = \cos\theta\,\rho(\theta,\xi) - \sqrt{1+\xi^2} - \ln\frac{\dfrac{\theta}{\sin\theta} + \rho(\theta,\xi)}{1 + \sqrt{1+\xi^2}}, \qquad (3.23)$$

with $\rho(\theta,\xi)$ defined in (3.16).

3.1.1 Stable computations

It will be clear that for numerical evaluations we need some preparations for small values of θ. First we observe that

$$\psi(\theta) = -\tfrac{1}{2}\sqrt{1+\xi^2}\,\theta^2 \left(1 + \frac{2 - 3\xi^2}{36(1+\xi^2)}\theta^2 + \mathcal{O}\left(\theta^4\right)\right), \quad \theta \to 0, \qquad (3.24)$$

7

and we can compute more terms in this expansion. However, these terms depend on ξ, and it is better to make first a few analytical steps. First we write

$$\cos\theta\rho(\theta,\xi) - \sqrt{1+\xi^2} = \frac{\left(\frac{\theta}{\sin\theta}\right)^2 - 1 - \theta^2 - \xi^2\sin^2\theta}{\cos\theta\rho(\theta,\xi) + \sqrt{1+\xi^2}}, \qquad (3.25)$$

and for

$$\left(\frac{\theta}{\sin\theta}\right)^2 - 1 = \frac{(\theta-\sin\theta)(\theta+\sin\theta)}{\sin^2\theta} \qquad (3.26)$$

we may use the simple expansion

$$\theta - \sin\theta = \tfrac{1}{6}\theta^3\left(1 + 6\sum_{k=1}^{\infty}(-1)^k\frac{\theta^{2k}}{(2k+3)!}\right). \qquad (3.27)$$

Although the convergence is very fast, in the algorithm we use a Chebyshev expansion for this quantity.

The logarithmic term in (3.23) needs also some care. We avoid the straightforward evaluation because when θ is small the argument tends to unity. In fact, we use a Chebyshev expansion of the function $\ln(1+z)$ that is accurate for small values of $|z|$. For this we write

$$z = \frac{\frac{\theta}{\sin\theta} + \rho(\theta,\xi)}{1 + \sqrt{1+\xi^2}} - 1 = \frac{\frac{\theta-\sin\theta}{\sin\theta} + \rho(\theta,\xi) - \sqrt{1+\xi^2}}{1 + \sqrt{1+\xi^2}}, \qquad (3.28)$$

which can be written as

$$z = \frac{\frac{\theta-\sin\theta}{\sin\theta}}{1 + \sqrt{1+\xi^2}}\left(1 + \frac{\frac{\theta}{\sin\theta} + 1}{\rho(\theta,\xi) + \sqrt{1+\xi^2}}\right). \qquad (3.29)$$

Finally, for the function $f(\theta)$ defined in (3.20) we need a stable representation of $r'(\theta)$. We have from (3.16)

$$r'(\theta) = \frac{\sin\theta - \theta\cos\theta}{2y\sin^2\theta}\left(1 + \frac{\theta}{\rho(\theta,\xi)\sin\theta}\right), \qquad (3.30)$$

and we write $\sin\theta - \theta\cos\theta = \sin\theta - \theta + 2\theta\sin^2\left(\tfrac{1}{2}\theta\right)$.

8

3.1.2 Where to use this quadrature method

The function $f(\theta)$ defined in (3.20) becomes singular when $s = r(\theta)e^{i\theta} = 1$. This happens if $\theta = 0$, and otherwise for imaginary values of θ. If $r(0) = 1$, then $y = x + 1$, and when $|y - x - 1|$ is small we need many function evaluations in the quadrature rule, which we like to avoid. When $|y - x - 1|$ is small we use asymptotic expansions. A safe domain follows from the asymptotic analysis and is given by

$$x + 1 - b\sqrt{2/\mu}\,\sqrt{2x + 1} < y < x + 1 + b\sqrt{2/\mu}\,\sqrt{2x + 1}, \qquad (3.31)$$

for some positive number b, say $b = 1$. Outside this parabolic shaped domain around the line $y = x + 1$ we can use for the representation in (3.22) the trapezoidal rule with great efficiency. It should be observed that, although the selection of the contour is based on asymptotic analysis, the quadrature method does not need necessarily large parameters, although it performs also quite well in that case.

The domain in (3.31) is described in terms of the scaled variables for $Q_\mu(\mu x, \mu y)$. For the unscaled variables in $Q_\mu(x, y)$ we use

$$x + \mu - b\sqrt{4x + 2\mu} < y < x + \mu + b\sqrt{4x + 2\mu}. \qquad (3.32)$$

In [41] the trapezoidal rule is used by including the pole at $s = 1$ in (3.10) in the function $\phi(s)$. That is, by writing (compare (3.11))

$$\widetilde{\phi}(s) = \frac{x}{s} + ys - \ln s - \frac{1}{\mu}\ln(1 - s). \qquad (3.33)$$

In that case the saddle point has to be calculated from a cubic polynomial and the contour follows from $\Im\widetilde{\phi}(s) = 0$ through that saddle point. Helstrom used an approximation of this contour by taking a parabola centered at the saddle point. The tabled results show correct values at the the critical values $y = x + 1$.

It is of interest to see how the method described in [4], in which the trapezoidal rule is used when a pole is close to the saddle point, can be applied to the integral in (3.8).

4 Zeros of special functions

There is ongoing work on software developments for the computation of zeros of special functions. The zeros of special functions are important quantities appearing in numerous scientific applications.

The problem we are addressing is, given a (special) function $f(x)$, develop a program for computing with certainty the solutions of $f(x) = 0$. We consider real zeros, but we also discuss the extension to complex zeros.

We restrict our attention to functions which are solutions of second-order differential equations

$$y''(x) + B(x)y'(x) + A(x)y(x) = 0. \qquad (4.1)$$

Special cases are, for instance, the classical orthogonal polynomials (for which the zeros are the nodes of Gaussian quadrature formulas), and Bessel functions.

Except for some particular cases (like, for instance, the Golub-Welsch algorithm for orthogonal polynomials, see for instance [33, §5.3.2]) the development of software for computing zeros of a function requires that software for computing the function is available; this is also the case of our algorithms. In this sense, the computation of zeros of special functions is secondary to the evaluation of the functions themselves. Most of the software for the evaluation of special functions is written in Fortran and we consider this programming language when special function software is available.

Another possibility is using commercial software with built-in special function commands. Mathematica and Maple are interesting possibilities with a large span of mathematical functions available We use Maple for some of our packages; current activities include the development of software packages for computing real or complex zeros of special functions using the fourth-order methods of [49, 51].

Next, we describe the basic ingredients for the method introduced in [49] for real zeros and the status of the software implementations, both in Fortran and in Maple. Here we motivate the method as a direct consequence of Sturm's comparison theorem [53, page 19]. In the second place, we briefly discuss the extension of the method for the computation of complex zeros presented in [51], we give some specific examples of application in Maple and discuss details of the implementation.

4.1 Computation of real zeros

The method introduced in [49] is a fast fourth-order method which is able to compute all the zeros of any solution of a second-order ODE $y''(x) + A(x)y(x) = 0$ [1] in any real interval where $A(x)$ is continuous, provided the

---

[1] Any differential equation $y''(x) + B(x)y'(x) + A(x)y(x) = 0$ with $B(x)$ differentiable can be transformed to normal form (no first derivative term) with the change of function $w(x) = \exp(\frac{1}{2}\int^x B(\zeta)d\zeta)y(x)$

monotonicity properties of $A(x)$ in this interval are known in advance. This method can be motivated as a direct consequence of the following Sturm theorem:

Theorem 1 (Sturm comparison) *Let $y(x)$ and $w(x)$ be solutions of $y''(x) + A_y(x)y(x) = 0$ and $w''(x) + A_w(x)w(x) = 0$ respectively, with $A_w(x) > A_y(x) > 0$. If $y(x_0)w'(x_0) - y'(x_0)w(x_0) = 0$ and x_y and x_w are the zeros of $y(x)$ and $w(x)$ closest to x_0 and larger (or smaller) than x_0, then $x_w < x_y$ (or $x_w > x_y$).*

The proof is straightforward. An intuitive explanation is as follows. The differential equations of the form $y''(x) + A(x)y(x) = 0$ have solutions which may oscillate if $A(x) > 0$, and the oscillations are more rapid as $A(x)$ is larger. In the theorem, because $A_w(x) > A_y(x) > 0$, the solutions of the second equation oscillate more rapidly and their zeros tend to be closer together. Now, because we have the hypothesis $y(x_0)w'(x_0) - y'(x_0)w(x_0) = 0$ we can consider that $y(x_0) = w(x_0)$ and $y'(x_0) = w'(x_0)$, and there is no loss of generality because a solution of a linear homogeneous ODE can be multiplied by a constant and it remains a solution of the same ODE with the same zeros. Therefore, the solutions $y(x)$ and $w(x)$ have the same initial conditions at x_0; but because $w(x)$ oscillates more rapidly than $y(x)$, then then there is necessarily a zero of $w(x)$ between x_0 and any zero of $y(x)$. This is illustrated graphically in Figure 1, where we consider the simple case of equations with constant coefficients.

From Sturm's theorem we can construct a method for the computation of the zeros of solutions of $y''(x) + A(x)y(x) = 0$ when $A(x)$ is monotonic. If $A(x)$ is a decreasing (increasing) function and $A(x) > 0$ we compute the zeros with an increasing (decreasing) sequence; if $A(x) < 0$ the solutions have one zero at most. Given a value x_0, the zero of $y(x)$ closest to x_0 and larger (smaller) than x_0 can be computed with certainty using the following scheme.

Algorithm 1 (Zeros of $y''(x) + A(x)y(x) = 0$, $A(x) > 0$ monotonic) .
Let $x_0 < \alpha$ with $y(\alpha) = 0$ and such that there is no zero of $y(x)$ between x_0 and α.

Assume that $A(x)$ is decreasing (increasing). Starting from x_0, compute x_{n+1} from x_n as follows: find a non-trivial solution of the equation $w''(x) + A(x_n)w(x) = 0$ such that $y(x_n)w'(x_n) - y'(x_n)w(x_n) = 0$. Take as x_{n+1} the zero of $w(x)$ closest to x_n and larger (smaller) than x_n. Then, the sequence $\{x_n\}$ converges to α.
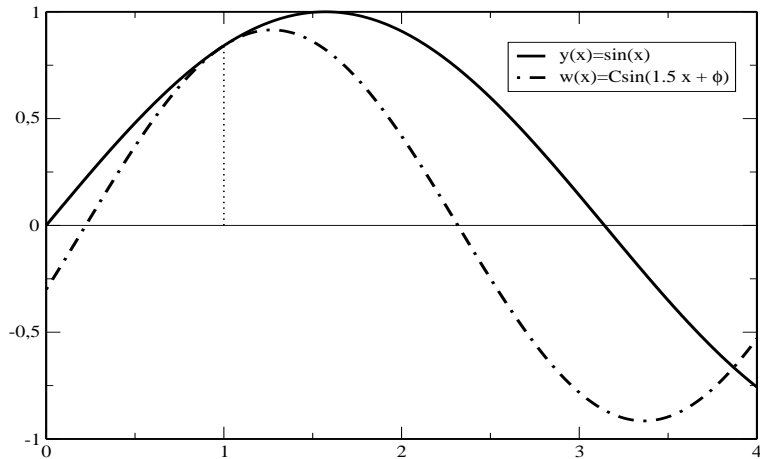
Figure 1: The function $y(x) = \sin(x)$ is a solution of $y''(x) + A_y(x)y(x) = 0$, $A_y(x) = 1$, and $w(x) = C\sin(1.5\,x + \phi)$, $C = 0.9153\ldots$, $\phi = -0.3336\ldots$ is a solution of $w''(x) + A_w(x)w(x) = 0$, $A_w(x) = 2.25$. The curves are tangent at $x_0 = 1$, where $y(x_0)w'(x_0) - y'(x_0)w(x_0) = 0$. Because $A_w(x) > A_y(x) > 0$, the zeros of $w(x)$ are closer to $x_0 = 1$ than the zeros of $y(x)$.

This is a direct consequence of Sturm comparison. Consider, for instance, the case of $A(x)$ increasing; $A(x_n) > A(x)$ for $x > x_n$ and the solutions of $w''(x) + A(x_n)w(x) = 0$ oscillate more rapidly than those of $y''(x) + A(x)y(x) = 0$, and because $y(x_n)w'(x_n) - y'(x_n)w(x_n) = 0$ we have $x_n < x_{n+1} < \alpha$. For increasing $A(x)$ the iterations with the algorithm produce an increasing sequence with upper bound the zero that is computed; therefore it converges, and necessarily to the zero because, as we will see, the only fixed point of the resulting iteration function (Eq. (4.2)) in $(x_n, \alpha]$ is precisely α. The situation is similar to Figure 1 ($x > 1$), where $y(x)$ would be the special function and $w(x)$ would provide the next iteration.

Of course, the algorithm can be applied successive times to generate a sequence of zeros.

Algorithm 2 (Computing a sequence of zeros, $A(x)$ monotonic) .
Let α_1, α_2 be consecutive zeros of $y(x)$, with $\alpha_1 < \alpha_2$.
If $A(x)$ is decreasing and α_1 is known, the zero α_2 can be computed

using Algorithm 1 with starting value $x_0 = \alpha_1$ (the first iteration being $x_1 = \alpha_1 + \pi/\sqrt{A(\alpha_1)}$).

If $A(x)$ is increasing and α_2 is known, the zero α_1 can be computed using Algorithm 1 with starting value $x_0 = \alpha_2$ (the first iteration being $x_1 = \alpha_2 - \pi/\sqrt{A(\alpha_2)}$).

As commented before the sequences generated are increasing (decreasing) if $A(x)$ is decreasing (increasing).

The iteration of Algorithm 1 can be explicitly written as follows:

$$T(x) = x - \frac{1}{\sqrt{A(x)}} \arctan_j(\sqrt{A(x)}h(x)), \qquad (4.2)$$

with $h(x) = y(x)/y'(x)$, $j = \text{sign}(A'(x))$ and

$$\arctan_j(\zeta) = \begin{cases} \arctan(\zeta) \text{ if } j\zeta > 0, \\ \arctan(\zeta) + j\pi \text{ if } j\zeta \leq 0, \\ j\pi/2 \text{ if } \zeta = \pm\infty. \end{cases} \qquad (4.3)$$

Observe that the only fixed points of $T(x)$ are the zeros of $y(x)$.

The method is of order four which means, roughly speaking, that the number of correct digits is multiplied by four in each iteration provided the iterations are close enough to the zero. On the other hand, it has good non-local behavior (see [49, Definition 4.1]), and once a zero is computed, few iterations are generally needed to reach a close estimation of the next zero. The fixed point method depends on an arctangent function but defined with a different range depending on whether $A(x)$ is decreasing or increasing. When an iteration close to the zero that is being computed is reached, it is better to switch to the standard arctangent functions (range $(-\pi/2, \pi/2)$), which gives a fixed point method continuous at the zero and convergent around the zero

The algorithms need some a priori analysis: the monotonicity properties of the coefficient $A(x)$ must be known in advance, because the method has to be applied separately in those subintervals where $A(x)$ is monotonic. This analysis has been completed for hypergeometric functions [5].

An excellent benchmark for these methods is Maple, which permits constructing algorithms for computing zeros of a large number of special functions; in [49], details on the performance of these methods are given for a good number of special functions. It is shown that only with three or four iterations per root, one can attain one hundred digits accuracy. Following these earlier tests, we are currently developing a Maple package for computing zeros of special functions [19]. By using the method that we are

describing in the next section, this package will later be extended to complex zeros.

Recently, we developed Fortran programs for computing zeros of Bessel functions $\mathcal{C}_\nu(x)$ (solutions of $x^2 y''(x) + xy'(x) + (x^2 - \nu^2)y(x) = 0$), of their first derivative and of the combination $x\mathcal{C}_\nu + \gamma\mathcal{C}'_\nu(x)$. In these methods, we considered the Fortran 77 codes of Amos [1] for computing Bessel functions. An update to Fortran 90 will be considered, together with new Fortran algorithms for computing zeros of other functions for which Fortran programs are available, like parabolic cylinder functions [32, 36], Legendre functions [48], conical functions [39], and modified Bessel functions of imaginary argument [27].

4.2 Complex zeros

It is possible to extend the previous fourth-order method to zeros in the complex plane, although it is not easy to prove its convergence in full generality. However, the WKB approximation (also named Liouville approximation) [44, Chap. 6] motivates why the method for complex zeros works [51].

Let us start by considering the trivial case of $A(z)$ constant. Then the general solution of $y''(z) + A(z)y(z) = 0$ reads

$$y(z) = C \sin\left(\sqrt{A(z)}\,(z - \psi)\right),$$

and the zeros are over the line

$$z = \psi + e^{-i\frac{\varphi}{2}}\lambda, \ \lambda \in \mathbb{R}, \ \varphi = \arg A(z).$$

In other words, writing $z = u + iv$ we have that the zeros are over an integral line of

$$\frac{dv}{du} = -\tan(\varphi/2). \tag{4.4}$$

Of course, in general $A(z)$ will not be a constant. The method for complex zeros is based on the assumption that the curves where the zeros lie are also given by (4.4), but with variable φ. This assumption is equivalent to considering that the WKB approximation is accurate. The WKB approximation with a zero at $z^{(0)}$ is

$$y(z) \approx CA(z)^{-1/4} \sin\left(\int_{z^{(0)}}^{z} A(\zeta)^{1/2} d\zeta\right).$$

Then, if $z^{(0)}$ is a zero, other zeros lie over the curve such that

$$\Im \int_{z^{(0)}}^{z} A(\zeta)^{1/2} d\zeta = 0, \tag{4.5}$$

14

and those curves are also given by (4.4). These are the so-called anti-Stokes lines (ASLs).

The method for computing complex zeros precisely follows the path of the ASLs and it is similar to the method for real zeros. Given $z^{(0)}$ ($y(z^{(0)}) = 0$) and assuming that $|A(z)|$ decreases for increasing $\Re z$, we consider the following algorithm to compute the next zero $z^{(1)}$:

Algorithm 3 (Basic algorithm for complex zeros; $|A(z)|$ decreasing)
.

1. *Take $z_0 = H^+(z^{(0)}) = z^{(0)} + \pi/\sqrt{A(z^{(0)})}$.*

2. *Iterate $z_{n+1} = T(z_n)$ until $|z_{n+1} - z_n| < \epsilon$, with*

$$T(z) = z - \frac{1}{\sqrt{A(z)}} \arctan\left(\sqrt{A(z)}\frac{y(z)}{y'(z)}\right). \qquad (4.6)$$

3. *Take as approximate zero $z^{(1)} = z_{n+1}$.*

The algorithm can be repeated in order to compute subsequent zeros. If $|A(z)|$ is decreasing, the same algorithm can be considered but with the first line replaced by $z_0 = H^-(z^{(0)}) = z^{(0)} - \pi/\sqrt{A(z^{(0)})}$.

Observations:

1. If $A(z)$ has slow variation, the first step could be a good approximation to $z^{(1)}$. In addition, the step is tangent to the anti-Stokes line (ASL) at $z^{(0)}$ (that is: the straight line joining $z^{(0)}$ and z_0 is tangent at $z^{(0)}$ to the ASL passing through this point).

2. $|z_0 - z^{(0)}| < L$ where L is the length of the anti-Stokes arc between $z^{(0)}$ and the next zero. It is a step by defect, and in the correct direction.

3. $T(z)$ is a fixed point iteration with order of convergence 4. This fact does not depend on the validity of the WKB approximation.

Figure 2, left, shows the complex zeros of the Bessel function $Y_{10.35}(z)$ in the first quadrant, the first estimations provided by the method together with the ASL passing through the zero with largest imaginary part. The algorithm starts with this zero and computes the following zeros (with successively smaller imaginary parts). The zeros are very close to the ASL and the first estimations are very reasonable with one exception: after computing the last zero with positive imaginary part, the estimation for the next

zero (which is on the real line) is not accurate and, furthermore, this zero appears well separated from the ASL. We conclude that the WKB approximation works initially well, but that it is not accurate for computing the last zero (although the iteration finally converges to this zero). The problem with this last zero is that WKB fails as a principal Stokes line is crossed. We next explain the notion of principal lines.

A Stokes line (SL) passing through z_0 for a differential equation $y''(z) + A(z)y(z) = 0$ is given by

$$\Re \int_{z_0}^{z} A(\zeta)^{1/2} d\zeta = 0. \tag{4.7}$$

Compare this with the definition of ASLs (4.5).

Given a point z_0 in the complex plane such that $A(z_0) \neq 0, \infty$, then there is one and only one ASL (or SL) passing through z_0. The situation is different if z_0 is a zero or a singularity; in particular, if z_0 is a zero of $A(z)$ with multiplicity m, there exist $m + 2$ ASLs (and SLs) emerging from z_0. We call the ASLs (or SLs) emerging from the zeros of $A(z)$ principal ASLs (SLs).

For the case of real zeros described in the previous section, the real interval where the zeros are sought has to be divided in different subintervals where $A(x)$ is monotonic and the direction of computation is chosen in such a way that this coefficient decreases. For complex zeros, a similar procedure has to be implemented, where the position with respect to the principal lines has to be analyzed. Summarizing, the following strategy proves to be reliable:

The strategy combines the use of $H^{(\pm)} = z \pm \dfrac{\pi}{\sqrt{A(z)}}$ and $T(z)$ (Eq. (4.6)) following these rules:

1. Divide the complex plane into disjoint domains separated by the principal ASLs and SLs and compute separately in each domain.

2. In each domain, start away from the principal SLs, close to a principal ASL and/or singularity (if any). Iterate $T(z)$ until a first zero is found. If a value outside the domain is reached, stop the search in that domain.

3. Proceed with the basic algorithm, choosing the displacements $H^{(\pm)}(z)$ in the direction of approach to the principal SLs and/or singularity.

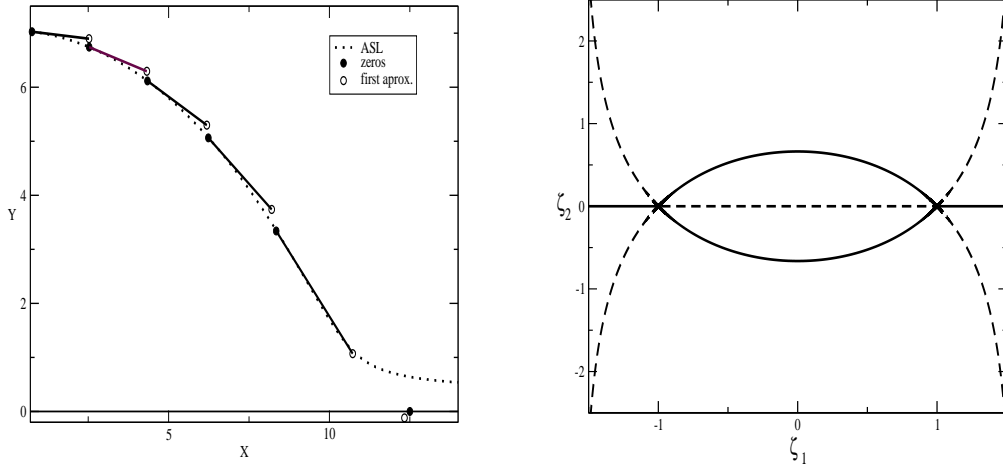4. Stop when a value outside the domain is reached.

Figure 2: **Left:** Zeros of the Bessel function $Y_{10.35}(z)$ in the quadrant $\Re z > 0$, $\Im z > 0$ (black circles), and first estimations to the zeros (white circles); the dotted line is the anti-Stokes line passing through the zero with larger imaginary part. **Right:** Principal Stokes (dashed lines) and anti-Stokes lines (solid lines) for the scaled Bessel equation $d^2y/d\zeta^2 + \lambda^2(1 - \zeta^{-2})y = 0$, $\lambda = \sqrt{\nu^2 - 1/2}$, $|\nu| > 1/2$. The lines for Bessel functions of order $|\nu| > 1/2$ correspond to the variable $z = \lambda\zeta$, $\lambda = \sqrt{\nu^2 - 1/2}$.

No exception has been found (so far tested for Bessel functions, parabolic cylinder functions and Bessel polynomials).

As an illustration, let us consider the computation of the complex zeros of Bessel functions, and particularly, of the Bessel function of the second kind $Y_\nu(z)$. The principal Stokes and ASLs are shown in Fig. 2, right. Because the zeros follow ASLs, we expect that the zeros could lie over the principal ASLs (the eye-shaped region and part of the real axis) or inside each of the domains separated by these lines. The eye-shaped region cuts the imaginary axis at $\zeta = ic$ where c are the real roots of $s - \frac{1}{2}\log((s+1)/(s-1)) = 0$, $s = \sqrt{1 + c^2}$, that is $c = \pm 0.66274321....$ In the particular case of $Y_\nu(z)$, $|\nu| > 1/2$, there are zeros very close to the eye-shaped region, in part of the positive real axis and close to the negative real axis.

First we discuss the computation of the zeros of $Y_\nu(z)$ for real orders $\nu > 1/2$; for other Bessel functions and real orders the algorithm will be essentially the same. For computing the zeros in the region satisfying $\Im z \geq 0$

and $|\Re z| < L$, with L a given positive number, we can consider the following steps:

1. Zeros on the positive real line or above the line: take the starting value $z = L + i$, and compute a first zero by iterating $T(z)$. Then, proceed using Algorithm 3 with H^- until a value z with $\Re z < \sqrt{\nu^2 - 1/4}$ or $\Im z < -\epsilon$, with ϵ a small positive number, is reached.[2]

2. Zeros above the negative real axis: take the starting value $z = -L + i$ and proceed similarly as for the positive real zeros, but with H^+.

3. Zeros on, above or inside the eye-shaped region: start with the estimation $z = 0.663\sqrt{\nu^2 - 1/4}$, compute a first zero and move to the right with H^+ until a value z such that $\Re z > \sqrt{\nu^2 - 1/4}$ is reached. Similarly, with the same starting zero, move to the left with H^- until a value until $\Re z < -\sqrt{\nu^2 - 1/4}$ is reached. It is also convenient to have, as for the real zeros, the stopping criterion $\Im z < -\epsilon$.

For the zeros with $\Im z < 0$, the same strategy as for $\Im z > 0$ can be used, mutatis mutandis. This strategy, in fact, works for any Bessel function of real order, and not only for $Y_\nu(z)$.

As an explicit example, we give some explicit Maple instructions to compute some of these zeros. Let us recall that the coefficient for Bessel functions is $A(z) = 1 - (\nu^2 - 1/4)/z^2$, for $y(z) = z^{1/2}Y_\nu(z)$ (or $y(z) = z^{1/2}J_\nu(z)$ and any linear combination). For computing the zeros of $Y_\nu(z)$ over the eye-shaped region in the first quadrant, we consider the following set of instructions.

```
    %% Definition of function and coefficients
    %% Parameters: NI, number of iterations;
    %% NI:=3 gives more than 20 correct digits
    %% a, order of the Bessel function
> restart:NI:=3:Digits:=30:f:=sqrt(z)*BesselY(a,z):
> h:=f/diff(f,z):coef:=sqrt(1-(a^2-0.25)/z^2):


    %% Definition of the iteration T and the displacements H+ and H-
> T:=z-1/coef*arctan(coef*h):Hplus:=z+Pi/coef:Hminus:=z-Pi/coef:a:=10.35:

    %% Computation of a first real zero close to x=40
> x:=40:
```

---

[2]The initial value $z = L$ also works in this case, but for functions different from $Y_\nu(z)$ it is preferable to initiate with $z = L + id$, d a positive number, in order to compute zeros above the positive real line.

```
> for i from 1 to NI do;
>     x:=evalf(subs(z=x,T));
> end do:
> k:=0:


     %% Computing the real zeros in decreasing order.
     %% The algorithm stops when abs(Re(x))<ll
> ll:=evalf(sqrt(a^2-1/4)):
> while abs(Re(x))>ll do;
>     xc[k+1]:=x;k:=k+1;x:=evalf(subs(z=xc[k],Hminus));
>     for i from 1 to NI do;
>          x:=evalf(subs(z=x,T));
>     end do;
> end do:

     %% Computing the first zero close to the eye-shaped region
> x:=0.662*sqrt(a^2-0.25)*I:
> for i from 1 to NI do;
>     x:=evalf(subs(z=x,T));
> end do:

     %% Computing the rest of zeros close to the eye-shaped region
     %% The algorithm stops when abs(Re(x))<ll
> while Re(x)<abs(ll) do;
>     xc[k+1]:=x;k:=k+1;x:=evalf(subs(z=xc[k],Hplus));
>     for i from 1 to NI do;
>          x:=evalf(subs(z=x,T));
>     end do;
> end do:

     %% Values of the first 9 real zeros and zeros on the first quadrant
> for k from 1 to k do;xc[k];end do;
                    40.861454343105291494227866820
                    37.604688987113844219164820306
                    34.324552547227846446332522063
                    31.012589590170552589468827474
                    27.655331475794737730108309198
                    24.229515208783561089174159817
                    20.689742504400913314605497407
                    16.926537747600538917549047798
                    12.500664303401789273444897836
```

   0.75171416445457988203833416828 + 7.02610400031859555166912384150 I
   2.52972140044497327028369512294 + 6.74083920492317316987229531276 I
```

19

```
4.3422234951299065821228793461O + 6.1157206477194742380481402678G I
6.2355302554244026119277788813T + 5.0629442771080G1019282O4374876 I
8.3441432951O4008518747O7241949 + 3.33927403881636773521569O9007 I
```

5   Published algorithms and related papers

In this section we give an overview of our published algorithms and we mention the papers that explain in detail the methods of computation for these codes.

A combination of numerical methods is needed for computing function values. Apart from the use of convergent and asymptotic series, the most frequent methods are those based on the use of linear recurrence relations and of numerical quadrature.

The classical reference for the computation using three-term recurrence relations is [13] and we also describe different computation methods in [33, Chapter 4]. In the use of recurrences, it is crucial to study the conditioning of the recursion (depending on the direction) and we have studied this problem in a series of papers, both in the case of Gauss hypergeometric functions and in the Kummer case. See [6, 7, 30, 34, 50].

Quadrature methods have been used for Airy and Scorer functions, certain Bessel and Legendre functions, and parabolic cylinder functions. The trapezoidal rule is particularly efficient for computing numerically many special function integral representations, particularly those arising from saddle point methods. In §3.1 we have given an example for a finite interval, and we have given more details in [25], [33, Chapter 5], and in earlier cited papers.

5.1   Airy and related functions

We started this topic with the related functions, also called inhomogeneous Airy functions or Scorer functions, which are solutions of the differential equations

$$\frac{d^2}{dz^2} w(z) - z\, w(z) = \pm \frac{1}{\pi}. \tag{5.1}$$

With the $+$ sign the standard solution is denoted by $\mathrm{Gi}(z)$, with $-$ sign by $\mathrm{Hi}(z)$. Standard solutions of the homogeneous equation are denoted by $\mathrm{Ai}(z)$ and $\mathrm{Bi}(z)$. For details on these functions we refer to [45].

For the Scorer functions we have described a number of contour integrals in the complex plane [20] and the corresponding algorithms can be found in [22]. For the Airy functions we have also used quadrature methods, see for the analysis [23] and for the algorithms [21]. We have also investigated the

zeros of the Scorer functions [26] and we have derived asymptotic expansions of these zeros.

In the same period other publications appeared on using asymptotic expansions of the Airy functions and the differential equation [12] with algorithms in [11].

5.2 Modified Bessel functions of imaginary order

The modified Bessel functions of the third kind of purely imaginary order $K_{ia}(x)$ is used in a number of problems from physics, and it is also the kernel of the Kantorovich-Lebedev transform. The function $I_{ia}(x)$ is not real, but the function

$$L_{ia}(x) = \tfrac{1}{2}\left[I_{-ia}(x) + I_{ia}(x)\right], \tag{5.2}$$

is a real valued numerically satisfactory companion to $K_{ia}(x)$ in the sense considered in [44, pp. 154–155]. The Wronskian relation for these functions is $\mathcal{W}\left[K_{ia}(x), L_{ia}(x)\right] = 1/x$.

In [24, 28] we have described the analytical details of the algorithms, and the codes are given in [27]. We have used power series representations for small values of x, asymptotic expansions for large x, Airy-type asymptotic expansions for large x near the turning point $x = a$, and numerical quadrature. Several non-oscillating integral representations have been used that can be obtained from contour integrals and by using saddle point methods.

5.3 Parabolic cylinder functions

These are solutions of the differential equations

$$\frac{d^2}{dx^2}w(x) - \left(\pm\tfrac{1}{4}x^2 + a\right)w(x) = 0. \tag{5.3}$$

The equation with the plus sign, with $U(a, x)$, $V(a, x)$ as two independent solutions, has for $a < 0$ two turning points at $z = \pm\sqrt{-2a}$; for large negative values of a uniform asymptotic representations in terms of Airy functions are available. Oscillations occur between the turning points. Compare this with the Hermite polynomial case when $a = -n - \tfrac{1}{2}$ with $n = 0, 1, 2, \ldots$. We have investigated many stable integral representations of these functions in [29] for using numerical quadrature. For the corresponding algorithms for $U(a, x)$, $V(a, x)$ and their derivatives for real parameters we refer to [31, 32].

In [37] we have considered the functions $W(a, x)$ and $W(a, -x)$, which are two linearly independent real solutions of the differential equation (5.3)

with the minus sign. In this case the oscillations occur outside the interval $[-\sqrt{2a}, \sqrt{2a}]$. In [36] we have given the algorithms for computing the $W-$functions and their derivatives.

In the algorithms for the parabolic cylinder functions we have used recursion, quadrature, and series expansions, including Maclaurin, local Taylor, Chebyshev and Airy-type asymptotic expansions. By factoring the dominant exponential factor, scaled functions could be computed to avoid overflow/underflow limitations. In this way rather large parameter domains in the (a, x) plane could be covered.

5.4   Legendre functions: toroidal and conical

The toroidal functions are $P^m_{n-\frac{1}{2}}(x)$ and $Q^m_{n-\frac{1}{2}}(x)$ and appear in the solution of Dirichlet problems with toroidal symmetry. For these functions we have used recurrence relations. For the backward recursions we have used starting values from uniform asymptotic expansions valid for fixed n and large m; the expansion is uniformly valid for large positive x. For the codes and related papers we refer to [16, 17, 18].

The conical function $P^\mu_{-1/2+i\tau}(x)$ is also an element of the class of associated Legendre functions, and the combination of the parameters makes it real for real values of x. This function is the kernel of the Mehler-Fock transform, which has numerous applications, and this function is also used in quantum physics, in particular describing the amplitude for Yukawa potential scattering. We have used recurrence relations with respect to $\mu = m$ (integer) and related continued fractions, and we have discussed the use of forward and backward recursions. For $x \in (-1, 1)$ we have used quadrature methods and uniform expansions for large μ in terms of elementary functions. For $x \geq 1$ we have used uniform expansions in terms of modified $K-$Bessel functions with purely imaginary order, in particular for describing the behavior of $P^\mu_{-1/2+i\tau}(x)$ near the turning point $x = \sqrt{\tau^2 + \mu^2}/\tau$. The methods of computation and the algorithms can be found in [35, 39].

Acknowledgements

References

[1] D. E. Amos. Algorithm 644: a portable package for Bessel functions of a complex argument and nonnegative order. *ACM Trans. Math. Software*, 12(3):265–273, 1986.

[2] S. András, Á. Baricz, and Y. Sun. The generalized Marcum Q-function: an orthogonal polynomial approach. *Acta Univ. Sapientiae Math.*, 3(1):60–76, 2011.

[3] S. K. Ashour and A. I. Abdel-Samad. On the computation of noncentral chi-square distribution function. *Comm. Statist. Simulation Comput.*, 19(4):1279–1291, 1990.

[4] C. Chiarella and A. Reichel. On the evaluation of integrals related to the error function. *Math. Comp.*, 22:137–143, 1968.

[5] A. Deaño, A. Gil, and J. Segura. New inequalities from classical Sturm theorems. *J. Approx. Theory*, 131(2):208–230, 2004.

[6] A. Deaño, J. Segura, and N. M. Temme. Identifying minimal and dominant solutions for Kummer recursions. *Math. Comp.*, 77(264):2277–2293, 2008.

[7] A. Deaño, J. Segura, and N. M. Temme. Computational properties of three-term recurrence relations for Kummer functions. *J. Comput. Appl. Math.*, 233(6):1505–1510, 2010.

[8] A.R. DiDonato and A. H. Morris, Jr. Computation of the incomplete gamma function ratios and their inverse. *ACM Trans. Math. Software*, 12(4):377–393, December 1986.

[9] A.R. DiDonato and A. H. Morris, Jr. Algorithm 654: FORTRAN subroutines for computing the incomplete gamma function ratios and their inverse. *ACM Trans. Math. Software*, 13(3):318–319, 1987.

[10] S. Dyrting. Evaluating the noncentral chi-square distribution for the Cox-Ingersoll-Ross process. *Computational Economics*, 24:35–50, 2004.

[11] B. R. Fabijonas. Algorithm 838: Airy functions. *ACM Trans. Math. Software*, 30(4):491–501, 2004.

[12] B. R. Fabijonas, D. W. Lozier, and F. W. J. Olver. Computation of complex Airy functions and their zeros using asymptotics and the differential equation. *ACM Trans. Math. Softw.*, 30(4):471–490, 2004.

[13] W. Gautschi. Computational aspects of three-term recurrence relations. *SIAM Rev.*, 9:24–82, 1967.

[14] W. Gautschi. Algorithm 542: Incomplete gamma functions. *ACM Trans. Math. Software*, 5(4):482–489, 1979.

[15] W. Gautschi. A computational procedure for incomplete gamma functions. *ACM Trans. Math. Software*, 5(4):466–481, 1979.

[16] A. Gil and J. Segura. A code to evaluate prolate and oblate spheroidal harmonics. *Comput. Phys. Commun.*, 108(2-3):267–278, 1998.

[17] A. Gil and J. Segura. Evaluation of toroidal harmonics. *Comput. Phys. Comm.*, 124:104–122, 2000.

[18] A. Gil and J. Segura. DTORH3 2.0: A new version of a computer program for the evaluation of toroidal harmonics. *Comput. Phys. Commun.*, 139(2):186–191, 2001.

[19] A. Gil and J. Segura. A Maple package for computing the zeros of special functions. *In preparation*, 2012.

[20] A. Gil, J. Segura, and N. M. Temme. On nonoscillating integrals for computing inhomogeneous Airy functions. *Math. Comp.*, 70(235):1183–1194, 2001.

[21] A. Gil, J. Segura, and N. M. Temme. Algorithm 819: AIZ, BIZ: two Fortran 77 routines for the computation of complex Airy functions. *ACM Trans. Math. Software*, 28(3):325–336, 2002.

[22] A. Gil, J. Segura, and N. M. Temme. Algorithm 822: GIZ, HIZ: two Fortran 77 routines for the computation of complex Scorer functions. *ACM Trans. Math. Software*, 28(4):436–447, 2002.

[23] A. Gil, J. Segura, and N. M. Temme. Computing complex Airy functions by numerical quadrature. *Numer. Algorithms*, 30(1):11–23, 2002.

[24] A. Gil, J. Segura, and N. M. Temme. Computation of the modified Bessel function of the third kind of imaginary orders: uniform Airy-type asymptotic expansion. *J. Comput. Appl. Math.*, 153(1-2):225–234, 2003. Proceedings of the Sixth International Symposium on Orthogonal Polynomials, Special Functions and their Applications (Rome, 2001).

[25] A. Gil, J. Segura, and N. M. Temme. Computing special functions by using quadrature rules. *Numer. Algorithms*, 33(1-4):265–275, 2003. International Conference on Numerical Algorithms, Vol. I (Marrakesh, 2001).

[26] A. Gil, J. Segura, and N. M. Temme. On the zeros of the Scorer functions. *J. Approx. Theory*, 120(2):253–266, 2003.

[27] A. Gil, J. Segura, and N. M. Temme. Algorithm 831: Modified Bessel functions of imaginary order and positive argument. *ACM Trans. Math. Software*, 30(2):159–164, 2004.

[28] A. Gil, J. Segura, and N. M. Temme. Computing solutions of the modified Bessel differential equation for imaginary orders and positive arguments. *ACM Trans. Math. Software*, 30(2):145–158, 2004.

[29] A. Gil, J. Segura, and N. M. Temme. Integral representations for computing real parabolic cylinder functions. *Numer. Math.*, 98(1):105–134, 2004.

[30] A. Gil, J. Segura, and N. M. Temme. The ABC of hyper recursions. *J. Comput. Appl. Math.*, 190(1–2):270–286, 2006.

[31] A. Gil, J. Segura, and N. M. Temme. Algorithm 850: real parabolic cylinder functions $U(a,x)$, $V(a,x)$. *ACM Trans. Math. Software*, 32(1):102–112, 2006.

[32] A. Gil, J. Segura, and N. M. Temme. Computing the real parabolic cylinder functions $U(a,x)$, $V(a,x)$. *ACM Trans. Math. Software*, 32(1):70–101, 2006.

[33] A. Gil, J. Segura, and N. M. Temme. *Numerical methods for special functions*. SIAM, Philadelphia, PA, 2007.

[34] A. Gil, J. Segura, and N. M. Temme. Numerically satisfactory solutions of hypergeometric recursions. *Math. Comp.*, 76(259):1449–1468, 2007.

[35] A. Gil, J. Segura, and N. M. Temme. Computing the conical function $P^{\mu}_{-1/2+i\tau}(x)$. *SIAM J. Sci. Comput.*, 31(3):1716–1741, 2009.

[36] A. Gil, J. Segura, and N. M. Temme. Algorithm 914: parabolic cylinder function $W(a,x)$ and its derivative. *ACM Trans. Math. Software*, 38(1):Art. 6, 5, 2011.

[37] A. Gil, J. Segura, and N. M. Temme. Fast and accurate computation of the Weber parabolic cylinder function $W(a, x)$. *IMA J. Numer. Anal.*, 31(3):1194–1216, 2011.

[38] A. Gil, J. Segura, and N. M. Temme. Efficient and accurate algorithms for the computation and inversion of the incomplete gamma function ratios. *SIAM J. Sci. Comput.*, 34(6):A2965–A2981, 2012.

[39] A. Gil, J. Segura, and N. M. Temme. An improved algorithm and a Fortran 90 module for computing the conical function $P_{-1/2+i\tau}^m(x)$. *Comput. Phys. Commun.*, 183(3):794–799, 2012.

[40] A. Gil, J. Segura, and N. M. Temme. Algoritm 939:computation of the Marcum $Q-$function. *ACM Trans. Math. Softw.*, 40(3):to appear, 2013.

[41] C. W. Helstrom. Computing the generalized Marcum $Q-$function. *IEEE Trans. Information Theory*, 38(4):1422–1428, 1992.

[42] L. Knüsel and B. Bablok. Computation of the noncentral gamma distribution. *SIAM J. Sci. Comput.*, 17(5):1224–1231, 1996.

[43] J. I. Marcum. A statistical theory of target detection by pulsed radar. *Trans. IRE*, IT-6:59–267, 1960.

[44] F. W. J. Olver. *Asymptotics and special functions*. AKP Classics. A K Peters Ltd., Wellesley, MA, 1997. Reprint of the 1974 original [Academic Press, New York].

[45] F. W. J. Olver. Airy and related functions. In *NIST handbook of mathematical functions*, pages 193–213. U.S. Dept. Commerce, Washington, DC, 2010.

[46] G. H. Robertson. Computation of the noncentral F distribution (CFAR) detection. *IEEE Trans. Aerospace and Electron. Systems*, AES-12(5):568–571, 1976.

[47] A. H. M. Ross. Algorithm for calculating the noncentral chi-square distribution. *IEEE Trans. Inform. Theory*, 45(4):1327–1333, 1999.

[48] B. I. Schneider, J. Segura, A. Gil, X. Guan, and K. Bartschat. A new Fortran 90 program to compute regular and irregular associated Legendre functions. *Comput. Phys. Comm.*, 181(12):2091–2097, 2010.

[49] J. Segura. Reliable computation of the zeros of solutions of second order linear ODEs using a fourth order method. *SIAM J. Numer. Anal.*, 48(2):452–469, 2010.

[50] J. Segura and N. M. Temme. Numerically satisfactory solutions of Kummer recurrence relations. *Numer. Math.*, 111(1):109–119, 2008.

[51] Javier Segura. Computing the complex zeros of special functions. *Numer. Math.*, 124(4):723–752, 2013.

[52] D. A. Shnidman. The calculation of the probability of detection and the generalized Marcum $Q-$function. *IEEE Trans. Information Theory*, 35(2):389–400, 1989.

[53] G. Szegő. *Orthogonal polynomials.* American Mathematical Society, Providence, RI, 1975.

[54] N. M. Temme. Asymptotic and numerical aspects of the noncentral chi-square distribution. *Comput. Math. Appl.*, 25(5):55–63, 1993.