Fast and Reliable Computation of the Zeros of Some Notable Equations.

Javier Segura

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

Third BCAM Workshop on Computational Mathematics

Solving non-linear scalar equations $f(x) = 0$ is an elementary but central problem in numerical analysis.

A number of general purpose methods are studied in general NA courses, from the slow and safe bisection method to the faster but uncertain Newton method.

As we will see, it is possible to construct methods which are both fast and reliable for some families of notable equations.

We focus on particular problems where the function satisfies some functional relation and we exploit these relations (we will call these functions special, and we later specify how special they are).

Computation of the zeros of special functions

The problem: given **a function with several zeros** in an interval, compute all the zeros reliably and efficiently.

Reliability: no zero is missed and the method is convergent without accurate initial approximations.

Efficiency: the total count of iterations is small (and if the order of convergence is high so much the better).

We solve the problem for a wide set of functions including important cases (like computing the zeros of orthogonal polynomials for Gauss quadrature, computing the zeros of Bessel functions and of their derivatives,...).

Most published algorithms rely on first approximations + Newton method.

Reliable?: maybe. There is no proof of convergence.

Efficient?: yes, but can be improved without additional computational cost.

Additionally, one needs particular approximations for each different function and sometimes they are hard.

The initial approximations are difficult to handle for functions depending on several parameters.

By special functions we mean functions satisfying first order linear systems

$$\begin{array}{l} y_n'(x) = a_n(x)y_n(x) + d_n(x)y_{n-1}(x) \\ y_{n-1}'(x) = b_n(x)y_{n-1}(x) + e_n(x)y_n(x) \end{array} \quad (DDE)$$

and/or three-term recurrence relations

$$y_{n+1}(x) + \beta_n(x)y_n(x) + \alpha_n(x)y_{n-1}(x) = 0 \quad (TTRR)$$

and/or second order ODEs

$$y_n''(x) + B_n(x)y_n'(x) + A_n(x)y_n(x) = 0 \quad (ODE)$$

For each of the functional relations described above, specific methods have been developed

An example of special functions

Riccati-Bessel functions $\mathcal{C}_\nu(x) = \sqrt{x}(\cos\alpha J_\nu(x) - \sin\alpha Y_\nu(x))$ satisfy

$$\begin{aligned} \mathcal{C}'_\nu(x) &= -\eta_\nu(x)\mathcal{C}_\nu(x) + \mathcal{C}_{\nu-1}(x), \\ \mathcal{C}'_{\nu-1}(x) &= \eta_\nu(x)\mathcal{C}_{\nu-1}(x) - \mathcal{C}_\nu(x), \end{aligned} \qquad (DDE)$$

where $\eta_\nu(x) = \dfrac{\nu - 1/2}{x}$.
And from this,

$$\mathcal{C}_{\nu+1}(x) - \frac{2\nu}{x}\mathcal{C}_\nu(x) + \mathcal{C}_{\nu-1}(x) = 0 \quad (TTRR)$$

and

$$\mathcal{C}''_\nu(x) + \left(1 - \frac{\nu^2 - 1/4}{x^2}\right)\mathcal{C}_\nu(x) = 0 \quad (ODE)$$

Zeros of Bessel functions pop up in many applications: wave scattering, optical wave guides, quantum physics, quadrature.

Many other relevant functions satisfy these type of relations, and in particular classical orthogonal polynomials.

For these functions, the three methods (TTRR, DDE and ODE) can be applied.

TTRR METHODS

TTRR method

Consider recurrence relations of the form

$$a_n y_{n+1}(x) + b_n y_n(x) + c_n y_{n-1}(x) = g(x) y_n(x), \quad n = 0, 1, \ldots ,$$

Considering the first N relations:

$$\mathbf{J}_N \mathbf{Y}_N(x) + a_{N-1} y_N(x) \mathbf{e}_N + c_0 y_{-1}(x) \mathbf{e}_1 = g(x) \mathbf{Y}_N(x),$$

$\mathbf{e}_1 = (1, 0, \ldots, 0)^T$, $\mathbf{e}_N = (0, \ldots, 0, 1)^T$, $\mathbf{Y}_N(x) = (y_0(x), \ldots, y_{N-1}(x))^T$

$$\mathbf{J}_N = \begin{pmatrix} b_0 & a_0 & 0 & . & . & 0 \\ c_1 & b_1 & a_1 & 0 & . & 0 \\ 0 & c_2 & b_2 & a_2 & . & 0 \\ . & . & . & . & . & 0 \\ . & . & . & . & . & a_{N-2} \\ 0 & 0 & 0 & . & c_{N-1} & b_{N-1} \end{pmatrix}. \quad (1)$$

If for $x = x_0$ $a_{N-1} y_N(x_0) = c_0 y_{-1}(x_0) = 0$ we have the equation for an eigenvalue problem with eigenvalue $g(x_0)$.

For classical OPs of degree N: $y_N = P_N(x)$, $c_0 P_{-1} \equiv 0$ and the zeros of $y_N = P_N$ are exactly the eigenvalues of the Jacobi matrix $\mathbf{J}_N$ (Wilf (1967), Golub-Welsch (1969)).

For minimal solutions of three-term recurrence relations $a_{N-1} y_N(x) \approx 0$ for large enoguh N and the zeros of $y_{-1}(x)$ are obtained from a truncated eigenvalue problem (see Grad and Zakrajšek (1973), Ikebe (1975),..., Ball (2000))

Given a recurrence relation $a_n y_{n+1} + b_n y_n + c_n y_{n-1} = 0$, we say that $\{f_n\}$ is a minimal solution as $n \to +\infty$ if $\lim\limits_{n\to+\infty} \dfrac{f_n}{g_n} = 0$ for any other solution $\{g_n\}$ linearly independent of $\{f_n\}$. Example: the Bessel function $J_\nu(x)$ is minimal as $\nu \to +\infty$.

The pros and cons of using TTRR methods

Pros

1. For the orthogonal polynomial cases, all the zeros can be computed simultaneously.
2. Complex zeros, and not only real zeros, can be computed when they exist.
3. The method only requires the coefficients of the recursion, and no function values need to be computed.

Cons

1. Conditioning is not always good.
2. The type of recurrences is quite restrictive.
3. For minimal solutions: where to truncate?
4. Efficiency?

DDE METHODS

We are not discussing these methods.

They are more general than TTRR in some sense (they can be used for any solution and the equations are less restrictive) and less in a different sense (only real zeros). The method computes with certainty all the zeros in any given interval.

More details in our book “Numerical Methods for Special Functions”, SIAM (2007).

ODE METHODS

First observation: the equation may be used to speed-up a method.
For computing zeros of solutions of

$$w''(x) + B(x)w'(x) + C(x)w(x) = 0 \qquad (2)$$

using the Newton method we have order 2 generally.
But, assuming that $B(x)$ is differentiable we can transform (2) 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 $x_{n+1} = x_n - \frac{y(x_n)}{y'(x_n)}$ is now of third order.

The reason: if α is such that $y(\alpha) = 0$, then $y''(\alpha) = 0$.

Using one of the coefficients of the ODE we have obtained a third order method which uses the function and the first derivative. This means that

$$\epsilon_{n+1} \approx C\epsilon_n^3,\ \epsilon_k = x_k - \alpha$$

where α is a zero.

In the equation

$$y''(x) + A(x)y(x) = 0$$

we have a remaining coefficient we can use. This is our next step.

Construction of a fixed point method of order 4

Taking $h(x) = y(x)/y'(x)$, we have $h'(x) = 1 + A(x)h(x)^2$.

If $y(\alpha) = 0$ and $A(x)$ has slow variation:

$$\int_\alpha^x \frac{h'(\zeta)}{1 + A(x)h(\zeta)^2} d\zeta \approx x - \alpha,$$

and assuming $A(x) > 0$

$$\alpha \approx x - \frac{1}{w(x)} \arctan\left(w(x)h(x)\right), \; w(x) = \sqrt{A(x)}$$

A classroom exercise:

Prove that the fixed point method $x_{n+1} = g(x_n)$,

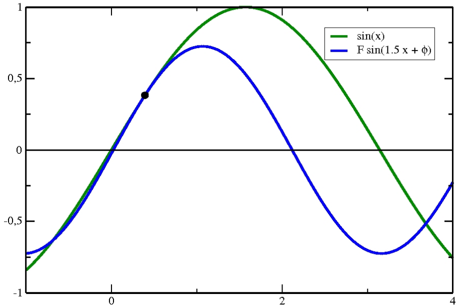
$$g(x) = x - \frac{1}{\sqrt{A(x)}} \arctan\left(\sqrt{A(x)}\frac{y(x)}{y'(x)}\right)$$

with $y''(x) + A(x)y(x) = 0$, $A(x)$ differentiable, has order order of convergence four.

But the goal is to compute all the zeros in a given interval. How to be sure?

Theorem (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 $A_y(x)>A_w(x)>0$. 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 $x_y<x_w$ (or $x_y>x_w$).



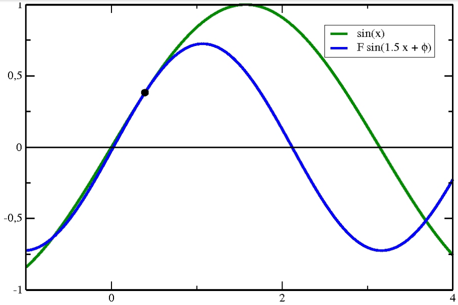
Equations: $y''(x)+y(x)=0$, $y''(x)+2.25y(x)=0$

Algorithm (Zeros of $y''(x)+A(x)y(x)=0$, $A(x)$ monotonic)

Given x_n, the next iterate x_{n+1} is computed as follows: find a solution of the equation

$$w''(x)+A(x_n)w(x)=0$$

such that $y(x_n)w'(x_n)-y'(x_n)w(x_n)=0$. If $A'(x)<0$ ($A'(x)>0$) take as x_{n+1} the zero of $w(x)$ closer to x_n and larger (smaller) than x_n.



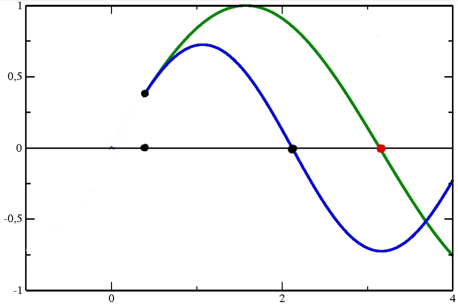
Equations: $y''(x)+y(x)=0$, $y''(x)+2.25y(x)=0$

Algorithm (Zeros of $y''(x)+A(x)y(x)=0$, $A(x)$ monotonic)

Given x_n, the next iterate x_{n+1} is computed as follows: find a solution of the equation

$$w''(x)+A(x_n)w(x)=0$$

such that $y(x_n)w'(x_n)-y'(x_n)w(x_n)=0$. If $A'(x)<0$ ($A'(x)>0$) take as x_{n+1} the zero of $w(x)$ closer to x_n and larger (smaller) than x_n.



Equations: $y''(x)+A(x)y(x)=0$, $w''(x)+A(x_n)w(x)=0$, ($A'(x)<0$)

The method is equivalent to iterating $x_{n+1} = T(x_n)$ with the following fixed point iteration.

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

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

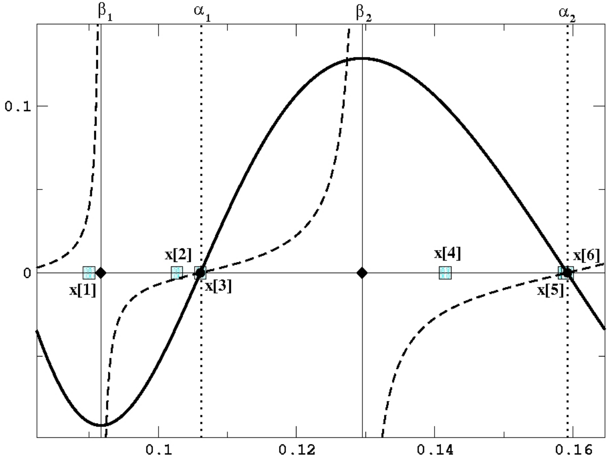
This method converges to α for any x_0 in $[\alpha', \alpha)$ if $A'(x) < 0$, with α' the largest zero smaller than α (analogously for $A'(x) > 0$).

The method has fourth order convergence:

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

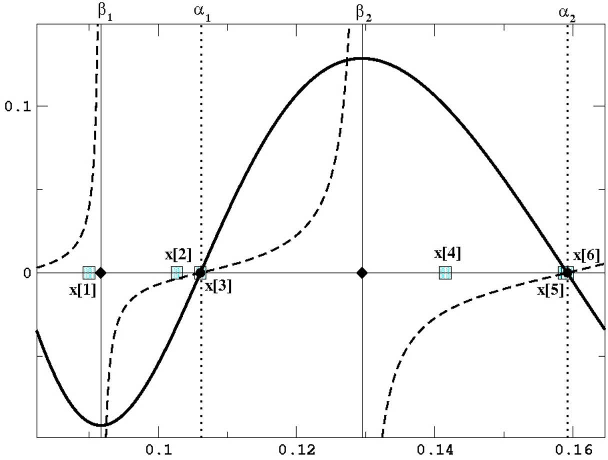
Computing the zeros in an interval where $A(x)$ is monotonic.

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.

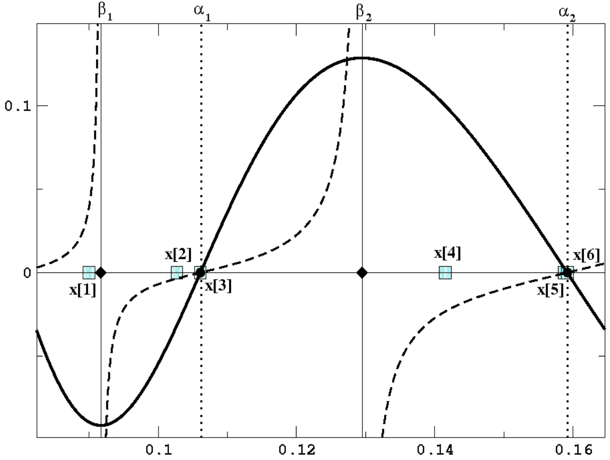
Example: zeros of $y(x)=x\sin(1/x)$, satisfying $y''(x)+x^{-4}y(x)=0$ (4 digits of acc.).



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.

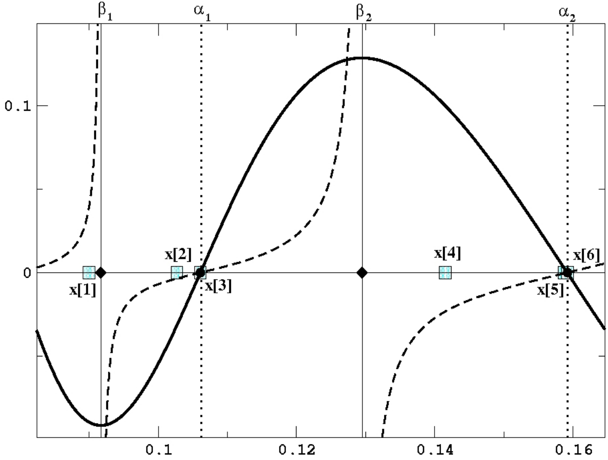
Example: zeros of $y(x) = x\sin(1/x)$, satisfying $y''(x) + x^{-4}y(x) = 0$ (4 digits of acc.).



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.)

The algorithm

The basic algorithm is as simple as this:

Algorithm

Computing zeros for $A'(x) < 0$

1. *Iterate $T(x)$ starting from x_0 until an accuracy target is reached. Let α be the computed zero.*
2. *Take $x_0 = T(\alpha) = \alpha + \pi/\sqrt{A(\alpha)}$ and go to 1.*

Repeat until the interval where the zeros are sought is swept. For $A'(x) > 0$ the same ideas can be applied but the zeros are computed in decreasing order.
See JS, SIAM J. Numer. Anal. (2010).

Requirement: the monotonicity properties of $A(x)$ should be known in advance in order to compute zeros in sub-intervals where $A(x)$ is monotonic.

But we already did that job for Gauss and confluent hypergeometric functions (A. Deaño, A. Gil, JS, JAT (2004))

For 100D accuracy, 3-4 iterations per root are enough and the method has proved reliability.

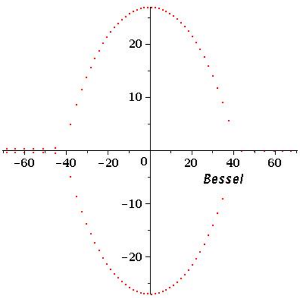
ODE METHOD FOR COMPLEX ZEROS

Computing complex zeros of special functions

The complex zeros of solutions of ODEs

$$y''(z) + A(z)y(z) = 0,$$

with $A(z)$ a complex meromorphic function, lie over certain curves.



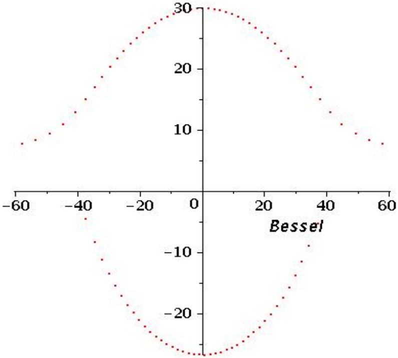
Zeros of the Bessel function $Y_\nu(z)$ of order $\nu = 40.35$

Computing complex zeros of special functions

The complex zeros of solutions of ODEs

$$y''(z) + A(z)y(z) = 0,$$

with $A(z)$ a complex meromorphic function, lie over certain curves.



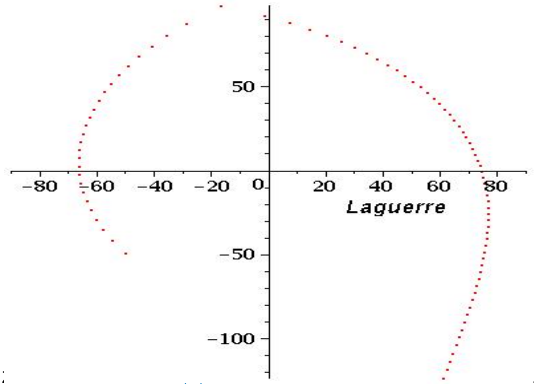
Zeros of the Bessel function of order $\nu = 40.35$ and with a zero at $z = 30i$.

Computing complex zeros of special functions

The complex zeros of solutions of ODEs

$$y''(z) + A(z)y(z) = 0,$$

with $A(z)$ a complex meromorphic function, lie over certain curves.



Zeros of $L_n^{(\alpha)}(z)$, $n = 26.2$, $\alpha = -83 + 20i$

Which are those curves?

Consider that two independent solutions of the ODE in a domain D can be written as

$$y_{\pm}(z)=q(z)^{-1/2}\exp\left(\pm i w(z)\right),\; w(z)=\int^{z} q(\zeta)d\zeta$$

If $y(z)$ is a solution such that $y(z^{(0)})=0$ then

$$y(z)=Cq(z)^{-1/2}\sin\left(\int_{z^{(0)}}^{z} q(\zeta)d\zeta\right)$$

Considering the parametric curve $z(\lambda)$, with $z(0)=z^{(0)}$ and satisfying

$$q(z(\lambda))\frac{dz}{d\lambda}=1$$

then $z(k\pi)$ are zeros of $y(z)$ because $\displaystyle\int_{z(0)}^{z(k\pi)} q(\zeta)d\zeta=k\pi,\, k\in\mathbb{Z}$.

Therefore, we have zeros over the integral curve (an exact anti-Stokes line)

$$\frac{dy}{dx}=-\tan(\phi(x,y)),\; q(z)=|q(z)|e^{i\phi(x,y)} \qquad (3)$$

passing through $z(0)=x(0)+iy(0)$

Problem: for computing $q(z)$ we need to solve

$$\frac{1}{2}q(z)\frac{d^2q(z)}{dz^2}-\frac{3}{4}\left(\frac{dq(z)}{dz}\right)^2-q(z)^4-A(z)q(z)^2=0$$

which seems worse than our original problem, which was solving

$$y''(z)+A(z)y(z)=0.$$

A drastic simplification

If $A(z)$ is constant the general solution of $y''(z)+A(z)y(z)=0$ is

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

and the zeros are over the line

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

The zeros lie over the integral lines

$$\frac{dy}{dx}=-\tan(\varphi/2). \qquad (4)$$

Ansatz: the zeros are approximately over (4) even if $A(z)$ is not a constant. This approximation is equivalent to consider $q(z)\approx\sqrt{A(z)}$. This is the WKB (or Liouville-Green) approximation:

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

First step towards an algorithm:

Let $z^{(0)}$ such that $y(z^{(0)})=0$. If $|A(z)|$ decreases for increasing $\Re z$ the next zero can be computed as follows:

1. $z_0=H^{+}(z^{(0)})=z^{(0)}+\pi/\sqrt{A(z^{(0)})}$
2. Iterate $z_{n+1}=T(z_n)$ until $|z_{n+1}-z_n|/|z_n|<\epsilon$, with

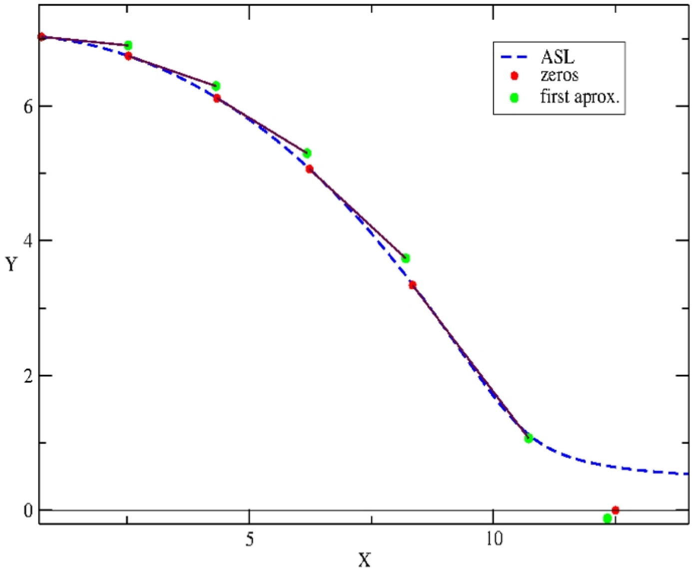
$$T(z)=z-\frac{1}{\sqrt{A(z)}}\arctan\left(\sqrt{A(z)}\frac{y(z)}{y'(z)}\right)$$

a Step 1 depends on WKB and the fact that $A(z)$ has slow variation.

b The straight line joining the points y $z^{(0)}$ y z_0 is tangent to the ASL arc at $z^{(0)}$. It is a step in the right direction and with an appropriate size if $A(z)$ varies slowly enough.

c Step 2 is a fixed point method of order 4, independently of the WKB approx.

Numerical example for $Y_{10.35}(z)$.



Things to consider before constructing an algorithm:

1. Where to start the iterations for computing a first zero
2. How to choose the appropriate direction
3. When to stop
4. How many ASLs do we need to consider

It is important to determine the structure of anti-Stokes and Stokes lines.

Stokes line through z_0: the curve

$$\Re \int_{z_0}^{z} \sqrt{A(\zeta)}d\zeta = 0,$$

Anti-Stokes line through z_0: the curve

$$\Im \int_{z_0}^{z} \sqrt{A(\zeta)}d\zeta = 0.$$

Some properties:

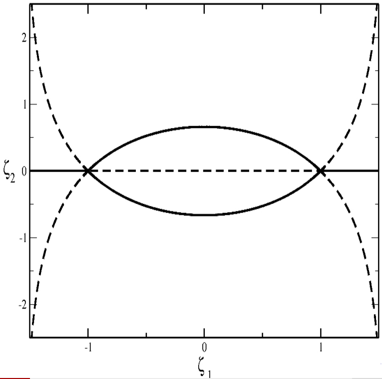
1. If $z_0 \in \mathbb{C}$ is not a zero or a singularity of $A(z)$ there is one and only one ASL passing through that point. The same is true for the Stokes lines.
2. If z_0 is not a zero or a singularity of $A(z)$ the ASL and the SL passing through that point intersect perpendicularly at z_0.
3. If z_0 is a zero of $A(z)$ of multiplicity m, $m+2$ ASLs (and SLs) emerge from z_0

Example: Bessel functions

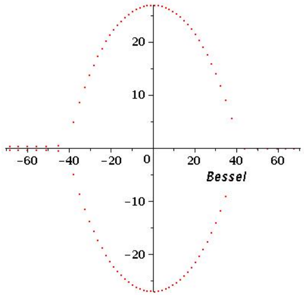
Principal Stokes (dashed line) and anti-Stokes lines (solid) for the equation

$$\frac{d^2y}{d\zeta^2} + (1 - \zeta^{-2})y = 0$$

(Bessel equation of orders $|\nu| > 1/2$ with the change $z = \zeta\sqrt{\nu^2 - 1/2}$).
Principal lines of $y''(z) + A(z)y(z) = 0$ are those emerging from the zeros of $A(z)$.

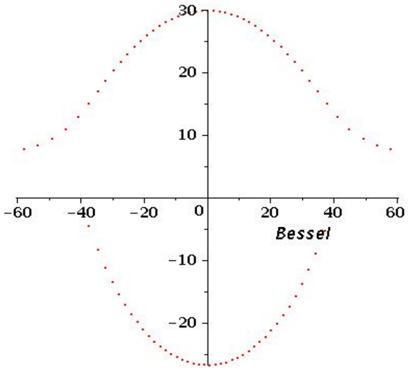


This explains the different patterns of zeros shown before.



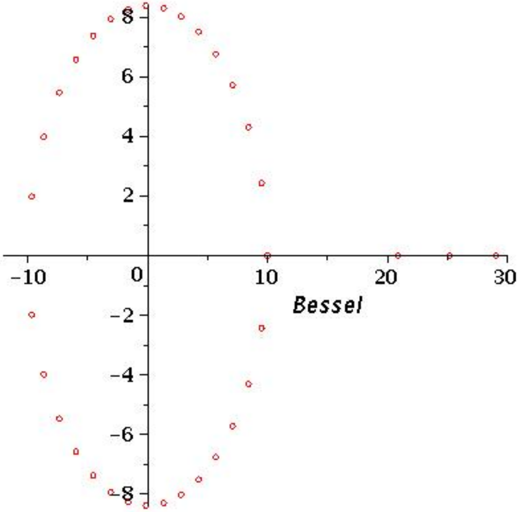
Zeros of the Bessel function $Y_\nu(z)$ of order $\nu = 40.35$

This explains the different patterns of zeros shown before.

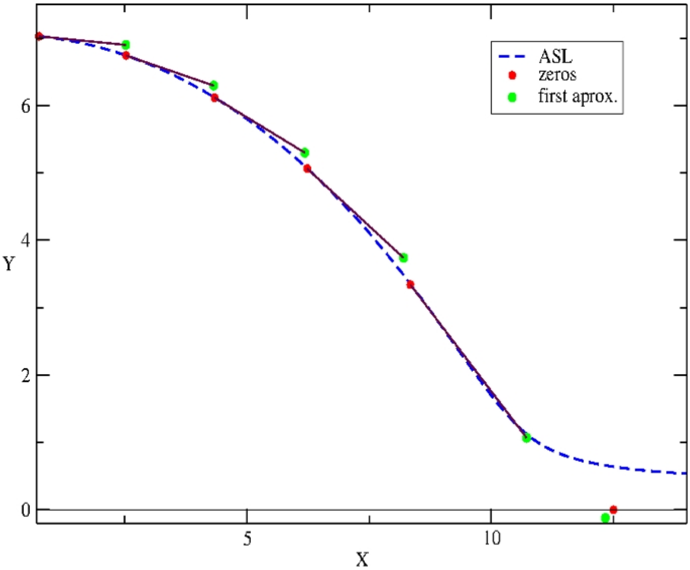


Zeros of the Bessel function of order $\nu = 40.35$ and with a zero at $30i$.

This explains the different patterns of zeros shown before.



Zeros of the Bessel function of order $\nu = 15.8$ and with a zero at $z = 10$.


ASL
zeros
first aprox.
6
4
2
0
Y
5
10
X

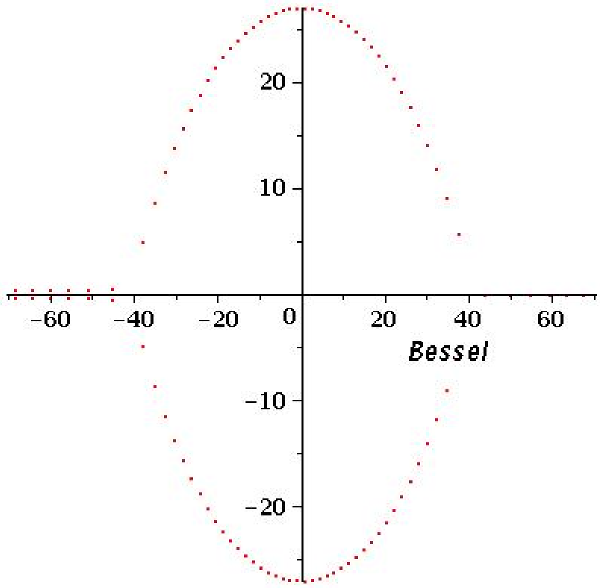
The strategy combines the use of $H^{(\pm)} = z \pm \dfrac{\pi}{\sqrt{A(z)}}$ and

$T(z) = z + \dfrac{1}{\sqrt{A(z)}} \arctan\left(\sqrt{A(z)}\dfrac{y(z)}{y'(z)}\right)$, following these rules:

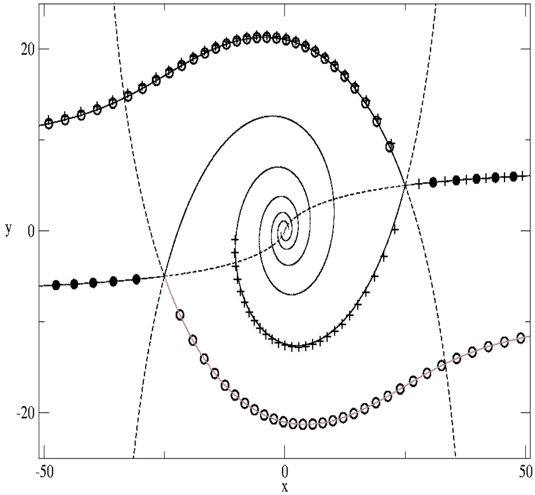
1. Divide the complex plane in disjoint domains separated by the principal ASLs and SLs and compute separately in each domain.
2. In each domain, start away of the principal SLs, close to a principal ASL and/or singularity (if any). Iterate $T(z)$ until a first zero is found. If a value outside the domain 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.

No exception has been found (so far tested for Bessel functions, PCFs and Bessel polynomials). See JS, Numerische Mathematik, 2013.
The method has fourth order convergence.

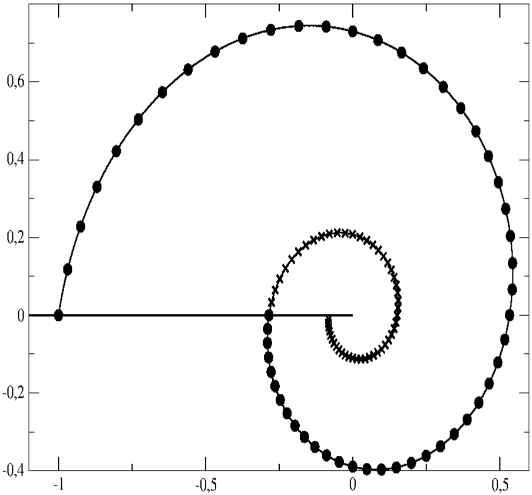
Some graphical examples follow, with zeros computed using our algorithm in Maple.



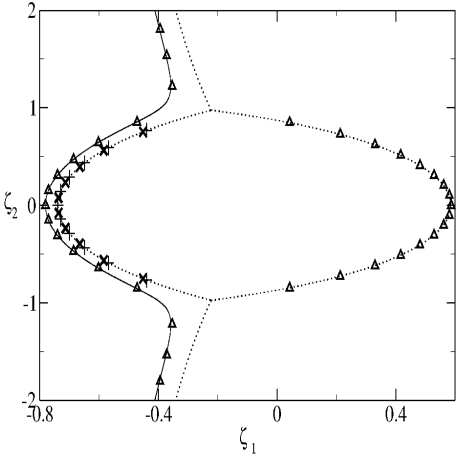
Zeros of the Bessel function $Y_\nu(z)$ of order $\nu = 40.35$



Stokes and anti-Stokes lines for the Bessel equation of order $\nu = 25 + 5i$, together with the zeros of $Y_\nu(x)$ $(+)$, $J_\nu(x)$ $(\bullet)$ and $J_{-\nu}(x)$ $(\circ)$.



Zeros of the two Bessel functions $y_{\nu}^{(k)} = e^{i\phi_k}\alpha J_{\nu}(z) + e^{-i\phi_k}\beta J_{-\nu}(z),\ \phi_k = 2\nu k\pi$, $\boldsymbol{k = 0, -1}$, with α and β such that $y_{\nu}^{(0)}(-1+0^{+}i) = 0$. $\nu = 25 + 5i$.



Zeros of the Bessel polynomials $L_n^{(1-2n-a)}(2z)$ in the variable $\zeta = z/|\gamma|$, $|\gamma| = \sqrt{(n+a/2)(n+a/2-1)}$. All the cases shown share the same ASLs and SLs in the variable ζ. The symbols $\times$ and $+$ correspond to the polynomial solutions $n = 10$, $a = 8$ and $n = 11$, $a = 8.572394\ldots$ Triangles correspond to the non-polynomial case $n = 10.6$, $a = 8.343446..$

Currently, we are developing sofware for computing zeros of specific functions. Particularly, the case of general Airy and Bessel functions is being considered.

The algorithms can be improved in several ways:

1. Using asymptotic information, it is possible to determine a priori how many ASLs containing zeros there exists.
2. Starting values for computing efficiently a first zero in each ASL can be found.
3. The zeros of the first derivative can also be included in the computation.
4. Use of local Taylor series

As example of this additional information, we mention some results we have recently proved concerning the complex zeros of Bessel functions (A. Gil, JS (2014)).

Theorem

Hankel functions $H^{(1)}(z) = J_\nu(z) + iY_\nu(z)$ and $H^{(2)}(z) = J_\nu(z) - iY_\nu(z)$ are the only pair of independent solutions of the Bessel equation (up to constant multiplicative factors) which do not have zeros for large and positive $\Re z$.

Theorem

The function $\mathcal{C}_\nu(\alpha, z) = \cos\alpha J_\nu(z) - \sin\alpha Y_\nu(z)$ with $\alpha \in [0, \pi)$ has a string of zeros over the negative real axis and a conjugated string below this axis if and only if one of these conditions is met:

1. $\alpha \neq 0$ *and* $\nu \in \mathbb{Z}$,
2. $\nu \in (0, 1/2)$ *and* $\alpha > \{\nu\}\pi$,
3. $\nu \in (1/2, 1)$ *and* $\alpha > \{\nu\}\pi$,

where $\{\nu\}$ is the fractional part of ν.

Theorem

Let ν be real and positive and sufficiently large. The functions $J_\nu(z)$, $H_\nu^{(1)}(z)$ and $H_\nu^{(2)}(z)$ are, up to constant factors, the only three solutions of the Bessel equation which do not have zeros for $\Re\tilde{z} \in (-1,1)$, $\tilde{z} = z/\nu$, both for $\Im z > 0$ and $\Im z < 0$ simultaneously.

Theorem

The intersection of the eye-shaped curve containing zeros of $\mathcal{C}_\nu(\alpha, z)$ with the imaginary axis takes place approximatelly at $ji\nu\tilde{y}_j$, $j = \pm 1$, with $\tilde{y}_j$ the (positive) solution of

$$g(\tilde{y}_j) + \frac{1}{2\nu}\log|1 - e^{2ji\alpha}| = 0,\ g(y) = \log\left(\frac{1+\sqrt{1+y^2}}{y}\right) - \sqrt{1+y^2}$$

Software plans:

1. Real cases for hypergeometric functions, including orthogonal polinomials, Bessel functions, Airy functions, Parabolic Cylinder functions, ... (Maple/Mathematica implementation).
2. Bessel functions, real and complex (Maple/Mathematica and Fortran).
3. Other complex functions (Maple/Mathematica).

THANK YOU!