Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

An adaptive parallel algorithm for computing zeros of analytic functions

Oswaldo Velásquez Castañón

oswaldo@imca.edu.pe

Universidad Nacional de Ingeniería

Manuel Toribio Cangana

Universidad Nacional de Ingeniería

Victor Galván Oyola

IC-UNICAMP

Research Article

Keywords: zeros of analytic functions, argument principle, exponential polynomials, Dirichlet polynomials, parallel computing, Newton–Kantorovich method

Posted Date: October 1st, 2026

DOI: https://doi.org/10.21203/rs.3.rs-9442949/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

An adaptive parallel algorithm for computing zeros of analytic functions

Oswaldo Velásquez Castañón[1,2*], Manuel Toribio Cangana[3] and Victor Galván Oyola[4]

[1*]Instituto de Matemática y Ciencias Afines, Universidad Nacional de Ingeniería, Calle Los Biólogos 245, Lima 15012, Perú.
[2*]Universidad del Pacífico, Jirón Sánchez Cerro 2141, Lima 15072, Perú.
[3]Facultad de Ciencias, Universidad Nacional de Ingeniería, Av. Túpac Amaru 210, Lima 15333, Perú.
[4]IC-UNICAMP, Av. Albert Einstein 1251, Campinas, 13083-852, Sao Paulo, Brazil.

*Corresponding author(s). E-mail(s): oswaldo@imca.edu.pe;
Contributing authors: mtoribio@uni.edu.pe; v187050@dac.unicamp.br;

Abstract

We develop a robust and efficient algorithm for computing the zeros of analytic functions with simple zeros inside a rectangle. The proposed method combines global information of the function to ensure convergence. It computes the variation of the argument via an adaptive zero-counting procedure inspired by the work of Ying and Katz [1]. The method further incorporates a two-dimensional bisection scheme, an exclusion criterion to discard subregions containing no zeros, and Newton–Kantorovich's method to accelerate computations and certify the results. The algorithm was implemented in parallel using C++, employing mutex-protected stacks for thread-safe processing and Boost libraries to support multiprecision computations. We apply this method to compute the zeros of exponential polynomials, which arise naturally in the study of delay-differential equations, exhibiting all three asymptotic types —retarded, advanced, and neutral— and, in particular, the zeros of the partial sums $\boldsymbol{\zeta_n(s) = \sum_{k=1}^{n} k^{-s}}$ of the Riemann zeta function for $\mathbf{3 \leq n \leq 26}$, computing over **2.5** million certified zeros within a guaranteed tolerance of $\mathbf{9.9 \times 10^{-21}}$.

Keywords: zeros of analytic functions, argument principle, exponential polynomials, Dirichlet polynomials, parallel computing, Newton–Kantorovich method

MSC Classification: 65H20 , 30C15 , 65Y20 , 65G20 , 65Y05

1 Introduction

Several problems in physics, mathematics, engineering, and other domains arise as root-finding problems of non-linear equations.

Exponential polynomials[1], functions written as $f(s) = \sum_{k=0}^{N} p_k(s)e^{-\lambda_k s}$, where $0 = \lambda_0 < \lambda_1 < \cdots < \lambda_N$ are real numbers, and $p_k(s)$, $0 \leq k \leq N$ are complex polynomials, appear naturally as characteristic functions for delay-differential equations [3, §3.5,§4.1], either describing particular exponential solutions to these equations or determining the asymptotic behavior of the general solutions [3, §4.4].

A subclass of these functions, the Dirichlet polynomials $f(s) = \sum_{k=0}^{n} a_k e^{\lambda_k s}$, where $a_0, \ldots, a_n \in \mathbb{C}$ and $0 = \lambda_0 < \lambda_1 < \cdots < \lambda_n$, for $s \in \mathbb{C}$ possess additional properties, such as almost-periodicity. They have applications in areas such as astronomy [4] and number theory, appearing for instance as partial sums of Dirichlet series.

The problem of computing zeros of analytic functions leads us to algorithms that use some combination of the following:

(I) a counting procedure, or some procedure for constructing an equation involving the zeros;
(II) a subdivision strategy, to reduce our computing problem to smaller subproblems,
(III) an exclusion criterion to discard subregions containing no zeros;
(IV) a refining stage of the computed zeros.

We discuss first (I). The *argument principle* is the default counting method. We recall some facts from basic complex analysis; see [5, §5.4]. Let $\gamma : [a, b] \to \mathbb{C}$ be a path (a continuous function with derivative on all the points of $[a, b]$, except for a finite collection of points, where it admits finite one-sided derivatives). A *continuous determination of the argument* of γ is a continuous function $\theta : [a, b] \to \mathbb{C}$ such that $e^{i\theta(t)} = \frac{\gamma(t)}{|\gamma(t)|}$ for all $t \in [a, b]$. Such a function θ is ensured to exist for a path $\gamma : [a, b] \to \mathbb{C} \setminus \{0\}$ without further conditions. With these notations, the *variation of argument* along γ is $\Delta_\gamma \arg = \theta(b) - \theta(a)$.

Let $f : \Omega \to \mathbb{C}$ be a holomorphic function, as meromorphic functions are not of our interest, and $R \subset \Omega$ a region whose boundary $\partial R \subset \Omega$ can be parametrized as a simple, oriented counterclockwise path. As stated by the argument principle [5, Proposition 8.3.2], the number of zeros of $f(s)$ in R, $N(R)$, is given by

$$N(R) = \frac{1}{2\pi} \Delta_{f \circ \partial R} \arg . \tag{1}$$

The methods that compute the zeros with this formula as starting point are also known as *homotopy methods*. The work of Ying and Katz [1, 6] considers a discretization of

[1] or exponential sums, according to [2]

the contour enclosing the zeros; a good enough discretization allows us to compute faithfully the number of zeros inside. Kowalczyk considers what he calls Discretized Cauchy's Argument Principle (DCAP) [7, §II. A. 5] in the same line.

We can also compute $N(R)$ as the integral $N(R) = \frac{1}{2\pi i}\int_{\partial R}\frac{f'(z)}{f(z)}dz$, now depending on quadrature methods. Similar integrals $S_k(R) = \frac{1}{2\pi i}\int_{\partial R} z^k\frac{f'(z)}{f(z)}dz$, called k-th *moments*, were used by Delves and Lyness [8] for computing the zeros; see also the work of Ioakimidis [9] avoiding the computation of the derivative. The error control for the integrals is already difficult, as it involves upper bounding $|f'/f|$ or derivatives of f'/f, which is related to lower bounding of $|f|$, that is, related to determine the non-vanishing of the denominator, so to the problem we are trying to solve. A different approach involves transforming the computation to a generalized eigenvalue problem involving Hankel matrices. In this direction, a Fortran implementation is provided by the ZEAL package [10, 11]; related contributions include the work of Gillan et al. [12] and Chen [13]. The work of Ioakimidis [14] is based on a solution of a homogeneous Riemann-Hilbert boundary problem.

Regarding (II), Delves and Lyness already discuss a subdivision strategy, although it is not good enough for our needs, because it produces overlaps in some cases. As the problem can be divided in smaller regions by its inherent independence, a subdivision process is the natural way to reduce numerical instability and tighten error bounds. Starting from a rectangle, a good (vertical or horizontal) symmetric subdivision strategy is already implemented in ZEAL, which provides smaller rectangles to work on. An adaptive subdivision strategy is described in Dellnitz et al. [15], the QZ-40 algorithm. The result from [15, Proposition 3.2] states its set-wise convergence. Frantsuzov and Artemyev [16] consider discretizing the parametrized curve, by considering the concept of *candidate pairs* [16, §2.2]; they perform also a triangulation of the region, and a corresponding subdivision strategy. A recursive decomposition procedure is discussed in [17], where the authors also address the possilibity of parallelization.

Regarding (III), an exclusion criterion determines the absence of zeros in a particular region by a quick test, without the need to perform more costly tests. A bisection-exclusion method is thoroughly analyzed, including the complexity in [18]. The work of Semenov [19] also implements an exclusion criterion based on Taylor's formula.

Finally, as steps (I)–(III) can be slow in general, a refining procedure based on Newton's method or similar can improve the speed of our solver. The work of Delves and Lyness already discuss a refining stage with Newton's method and secant method. Gillan et al. refine the zeros with Halley's method without derivatives. We highlight the work of Semenov [19] whose method is only based on a subdivision strategy either isolating the zeros by using a version of Kantorovich theorem, or discarding the rectangles by using their exclusion criterion, while the counting of the zeros is not considered at all.

In seeking a robust, certified method for computing zeros of analytic functions, we found a result from [1, p. 144–145] which establishes that a finite number of values of a function and its derivatives, themselves, are not enough to determine the number of zeros of an analytic function, or even of a complex polynomial, in a bounded region. The discussion on the difficulty of properly discretizing the domain [7, §II. C.], or even

a given segment, highlights the need for a criterion to ensure robustness. Therefore, the key of a robust algorithm for computing the zeros is the use of *global data* on the given function. This data is easily obtainable when working with special classes of functions, like exponential polynomials. Our proposed method is based on a variation of the argument computing criterion derived from [1]. This algorithm computes the number of zeros inside a given rectangle using a global bound function for the second derivative, which is not a strong condition, at least for the particular functions we are interested in. We derive an adaptive method for computing the variation of the argument of a function over a segment, while avoiding possible zeros over it. Then, we compute the number of zeros of our function inside any given rectangle. A two-dimensional bisection method, similar to the one found in ZEAL, allows us to parallelize the procedure, thereby speeding up the computation and ultimately isolating the zeros. Then, we apply Newton's method to compute the zeros with guaranteed precision. We provide also an exclusion criterion to be combined with the other procedures, thus addressing all four components (I)–(IV) outlined above.

Our method relies on knowledge of the first and second derivatives of a given function. We acknowledge closed-form expressions for these derivatives are not always available, and some authors indeed avoid their computation [7, 12]. Nevertheless, we emphasize that this oor some other form of global information is necessary to ensure the robustness of the computations – and in our case, the derivatives are readily available.

2 Zero counting with the argument principle

We start from formula (1). Let us suppose that we have an argument function $\arg : \mathbb{C}\backslash]-\infty, 0] \to \mathbb{R}$ with values in the interval $]-\pi/2, \pi/2[$. Then, if we have a path with $\left|\Delta_\gamma \arg\right| < \pi$, then $\Delta_\gamma \arg = \arg\left(\frac{\gamma(b)}{\gamma(a)}\right)$. But then, if R is a polygonal region contained in Ω such that $\partial R = \bigcup_{i=0}^{n-1}[z_i, z_{i+1}]$, where $z_0 = s_1$, $z_n = s_2$, and for $i = 0, \ldots, n-1$, the variation of argument is $\left|\Delta_{f\circ[z_i,z_{i+1}]} \arg\right| < \pi$, then

$$N(R) = \frac{1}{2\pi}\sum_{i=0}^{n-1} \Delta_{f\circ[z_i,z_{i+1}]} \arg = \frac{1}{2\pi}\sum_{i=0}^{n-1} \arg\Big(\frac{f(z_{i+1})}{f(z_i)}\Big).$$

The criterion at [1, Theorem 2.1] and written below allows us to determine the points z_i to divide a given curve ∂R and to compute the variation of argument of $f \circ \partial R$.

In what follows, $\langle x, y\rangle = \Re(x\overline{y})$ denotes the (real) scalar product in $\mathbb{C}$.

Theorem 1 (Ying and Katz) *Let $P(s)$ be an affine complex function, $f(s)$ a holomorphic function on the interval $[s_1, s_2]$, and $R(s)$ the function defined by $R(s) = f(s) - P(s)$. If*

$$\min_{s\in[s_1,s_2]} \left|P(s)\right| > \max_{s\in[s_1,s_2]} \left|R(s)\right|, \tag{2}$$

then

$$\left|\Delta_{f\circ[s_1,s_2]} \arg\right| < \pi \quad \text{and} \quad \Delta_{f\circ[s_1,s_2]} \arg = \arg\Big(\frac{f(s_2)}{f(s_1)}\Big).$$

More precisely, if $s_0 \in [s_1, s_2]$ *is such that* $P(s_0) = \min_{s\in[s_1,s_2]}\big|P(s)\big| > 0$, *then for all* $s \in [s_1, s_2]$, $\big\langle f(s), P(s_0)\big\rangle > 0$.

Proof As $P\big([s_1, s_2]\big)$ is a convex set and $P(s_0)$ is the projection from 0 onto this set, the classical geometrical characterization of this point [20, Theorem 5.2] establishes that for all $s \in [s_1, s_2]$ $\big\langle P(s) - P(s_0), P(s_0) - 0\big\rangle \geq 0$. Then

$$\begin{aligned}\big\langle f(s), P(s_0)\big\rangle &= \big\langle P(s), P(s_0)\big\rangle + \big\langle R(s), P(s_0)\big\rangle \geq \big|P(s_0)\big|^2 - \big|R(s)\big| \cdot \big|P(s_0)\big| \\ &\geq \Big(\min_{s\in[s_1,s_2]}\big|P(s)\big| - \max_{s\in[s_1,s_2]}\big|R(s)\big|\Big) \cdot \big|P(s_0)\big| > 0.\end{aligned}$$

□

We restate the following result from [1, Lemma 2.1], with a minor modification and with a corrected proof.[2]

Lemma 2 *Let* $f(s)$ *be a holomorphic function on the interval* $[s_1, s_2]$, $P(s)$ *and* $R(s)$ *given by*

$$P(s) = f(s_1) + \frac{f(s_2) - f(s_1)}{s_2 - s_1}(s - s_1)$$

and $R(s) = f(s) - P(s)$. *If* $M(s_1, s_2)$ *is such that for any* $s \in [s_1, s_2]$, $\big|f''(s)\big| \leq M(s_1, s_2)$, *then*

$$\max_{s\in[s_1,s_2]}\big|R(s)\big| \leq M(s_1, s_2)\frac{|s_1 - s_2|^2}{8}.$$

Proof We fix $u \in \mathbb{C}$ with $|u| = 1$ and define $f_u : [0, 1] \to \mathbb{R}$ by $f_u(t) = \Big\langle f\big(s_1 + t(s_2 - s_1)\big), u\Big\rangle$. Also, we define the interpolating (linear) polynomial $P_u : [0, 1] \to \mathbb{R}$ such that $P_u(0) = f_u(0) = \big\langle f(s_1), u\big\rangle$, $P_u(1) = f_u(1) = \big\langle f(s_2), u\big\rangle$, so that $P_u(t) = \Big\langle P\big(s_1 + t(s_2 - s_1)\big), u\Big\rangle$. for $t \in [0, 1]$. The interpolation error formula in [21, Theorem 2.1.4.1, p. 49] for degree $n = 1$ gives, the existence, for each $t \in]0, 1[$, of $\xi_t \in]0, 1[$ such that $f_u(t) - P_u(t) = \dfrac{f_u''(\xi_t)}{2!}t(t - 1)$, which translates to

$$\langle R(t), u\rangle = \frac{\Big\langle f''\big(s_1 + \xi_t(s_2 - s_1)\big)(s_2 - s_1)^2, u\Big\rangle}{2}t(t - 1).$$

By using the bound $M(s_1, s_2)$ for $|f''(s)|$ and the bound $0 \leq t(1 - t) \leq 1/4$ we obtain

$$\big|\langle R(t), u\rangle\big| \leq M(s_1, s_2)\frac{|s_1 - s_2|^2}{8}.$$

Finally, for $t \in [0, 1]$ it suffices to take u such that $\langle R(t), u\rangle = \big|R(t)\big|$ and reinsert it in the last inequality. □

The above lemma replaces the computation of $\max\big|R(s)\big|$ by a simpler estimate derived from a bound on $\big|f''(s)\big|$. Moreover, the minimum of $\big|P(s)\big|$ can be computed in

Algorithm 1 get_min(a, b)

Input: $a, b \in \mathbb{C}$

Output: The minimum of $\big|(1-t)a + tb\big|$ for $0 \le t \le 1$

```
c ← a·b̄
if ℜ(c) ≥ |a|² then                    ▷ when the minimum is at t ≤ 0
    return |a|
end if
if ℜ(c) ≥ |b|² then                    ▷ when the minimum is at t ≥ 1
    return |b|
end if
return |ℑ(c)|/|b − a|                  ▷ when the minimum is at 0 < t < 1
```

a straightforward manner, as shown in [1, Lemma 2.2] and implemented in Algorithm 1 below.

With all this information, our contour discretization criterion consists in finding smaller segments $[s_1', s_2']$ such that

$$\min_{s \in [s_1', s_2']} \big|P(s)\big| > M(s_1', s_2') \frac{|s_1' - s_2'|^2}{8}. \tag{3}$$

This, together with Lemma 2, ensures that the condition of Theorem 1 is satisfied for $f(s)$ on $[s_1', s_2']$.

The following proposition, of purely theoretical interest, asserts the existence of a convenient discretization of a fixed interval in the complex plane without zero of $f(s)$, which allows us to compute the variation of argument of our function on this segment.

Proposition 3 (Convergence) *Given a function $f(s)$ holomorphic and without zeros over the interval $[s_1, s_2]$, and $M(s_1, s_2) > 0$ such that, for every $s \in [s_1, s_2]$, $M(s_1, s_2) \ge \big|f''(s)\big|$, there exists $\delta > 0$ such that, if $[s_1', s_2'] \subset [s_1, s_2]$ with $|s_1' - s_2'| < \delta$ and*

$$P(s) = f(s_1') + \frac{f(s_2') - f(s_1')}{s_2' - s_1'}(s - s_1'),$$

then

$$\min_{s \in [s_1', s_2']} \big|P(s)\big| > M(s_1, s_2) \frac{|s_1' - s_2'|^2}{8} \ge \max_{s \in [s_1', s_2']} \big|R(s)\big|,$$

where $R(s) = f(s) - P(s)$.

Proof First $m = \min_{s \in [s_1, s_2]} \big|f(s)\big| > 0$. By uniform continuity, there exists $\delta' > 0$ such that, for $x, y \in [s_1, s_2]$, $|x - y| < \delta'$, we have $\big|f(x) - f(y)\big| < m/2$. If $[s_1', s_2'] \subset [s_1, s_2]$ is a segment with $|s_1' - s_2'| < \delta = \min\{\delta', 2\sqrt{m/M(s_1, s_2)}\}$, then $\big|f(s_1') - f(s_2')\big| < m/2$, from which

$$\big|P(s)\big| \ge \big|f(s_1')\big| - \big|f(s_1') - f(s_2')\big| > \frac{m}{2},$$

[2] The original statement applies the Lagrange interpolation error formula to a complex-valued function, yielding a pointwise identity that does not hold in general, as its proof relies on Rolle's theorem, which fails for complex-valued functions.

and

$$\min_{s\in[s_1',s_2']}\big|P(s)\big| \geq \frac{m}{2} > M(s_1,s_2)\frac{|s_1'-s_2'|^2}{8} \geq \max_{s\in[s_1',s_2']}\big|R(s)\big|.$$

This finishes the proof. □

When trying to obtain (3), it may happen that $M(s_1,s_2)$ is too large as a bound, or $\min_{s\in[s_1',s_2']}\big|P(s)\big|$ is too small –for example, if there is a zero of $f(s)$ near the segment $[s_1',s_2']$, and the condition will never be met in practice– because $|s_1'-s_2'|$ will become too small. Thus, a tolerance stop has to be considered in our method. Otherwise, too small values of $|s_1'-s_2'|$ will need too many steps to compute the variation of argument on $[s_1,s_2]$, and they will accumulate too many round-off errors. The best strategy will be to seek large values of $|s_1'-s_2'|$ when possible and, as we consider a smooth variation of the argument on every segment, an adaptive strategy will be our best choice.

Remark If $f(s)$ admits zeros over the interval $[s_1,s_2]$, the condition (2) will not be satisfied, and our method will generate a sequence of steps converging to zero that will not allow the choice of $[s_1',s_2']$ covering this zero. Then the stop criterion will tell us to change our choice of interval.

We implement a backtracking line search method similar to Algorithm 3.1 in [22] with stop criteria (3). We note that if $[s_1',s_2']\subset[s_1,s_2]$, then $\big|f''(s)\big| \leq M(s_1,s_2)$ for $s\in[s_1',s_2']$. If we define $R(s_1,s_2) = M(s_1,s_2)\frac{|s_2-s_1|^2}{8}$ and $\max_{s\in[s_1,s_2]}\big|R(s)\big|$ as in Proposition 3, then

$$\max_{s\in[s_1',s_2']}\big|R(s)\big| \leq R(s_1,s_2)\frac{|s_2'-s_1'|^2}{|s_2-s_1|^2} \tag{4}$$

and we can take the right-hand side of the last inequality as $R(s_1',s_2')$ when needed. This saves us from recomputing this costly upper bound too frequently.

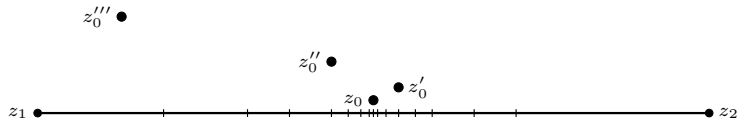


Fig. 1 Discretization of given segment. As a cluster of zeros is approached, the argument varies more rapidly, requiring finer subdivisions

The resulting method, allowing us to compute the variation of the argument of a function along a segment (hence on any polygonal curve) is given in Algorithm 2.

3 Two-dimensional bisection method

Having established a method for counting the zeros inside a rectangle, we now describe a procedure to compute them by first isolating them.

Algorithm 2 var_arg(s_1, s_2)

Input: $s_1, s_2 \in \mathbb{C}$

Output: The variation of argument of $f(s)$ over $[s_1, s_2]$

$d \leftarrow s_2 - s_1$
$s_3 \leftarrow s_1$
$t \leftarrow 1$
$t_0 \leftarrow 0$
$ans \leftarrow 0$
while $t_0 + \varepsilon_1 < 1$ **do**
    $s_4 \leftarrow s_3 + td$
    $fs_3 \leftarrow f(s_3)$
    $R_0 \leftarrow M(s_3, s_4)\dfrac{|s_4 - s_3|^2}{8}$
    $minP \leftarrow \text{get_min}\left(fs_3, f(s_4)\right)$
    while $minP \leq R_0 + \varepsilon$ and $t > \varepsilon_2$ **do**
        $t \leftarrow t\rho$
        $R_0 \leftarrow R_0\rho^2$ ▷ because of (4)
        $s_4 \leftarrow s_3 + td$
    end while
    if $t \leq \varepsilon_2$ **then**
        return FAIL
    end if
    $ans \leftarrow ans + \arg\left(f(s_4)/fs_3\right)$
    $t_0 \leftarrow t_0 + t$
    $s_3 \leftarrow s_3 + dt$
    $t \leftarrow \min\{\left|(s_2 - s_3)/d\right|, t/\rho, 1 + \varepsilon_1 - t_0\}$
end while
return ans

3.1 Subdivision strategy

Our main assumption is that $f(s)$ has no zeros on the boundary ∂R of the initial rectangle. We divide this rectangle by computing the mid-points of the longest side. Then, the variation of argument between these midpoints is computed, and the rectangle is subdivided recursively. This last step may fail if the function has zeros on or near the segment used for subdivision. In that case, we recompute the argument along a slight perturbation of the subdivision segment. In this way, we ensure that, at each step, there are no zeros on the boundaries of the resulting rectangles. The process is illustrated in Figure 2.

In each step, we obtain up two rectangles of smaller diameter that can be handled similarly. This particular setup is described in Section 5. Each resulting rectangle can also be discarded directly via the exclusion criterion described below. We are eventually led to a collection of isolated zeros, each enclosed in a size-balanced rectangle whose diameter converges to zero. As this process is slow, we will speed it up by combining it with Newton's method.

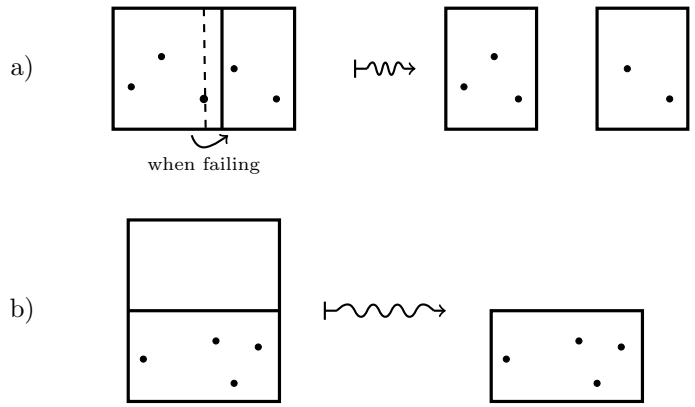


Fig. 2 Bisection and exclusion strategy

3.2 Exclusion criterion

Now we recall the exclusion criterion described in [6, Theorem 2.2], restated as follows. In the following, we require a function $N(z_1, z_2)$ giving a bound for the second derivative of $f(s)$ inside a rectangle or corners z_1 and z_2:

$$\big|f''(s)\big| \leq N(z_1, z_2), \text{ for } s \in R = R(z_1, z_2),$$

the rectangle with opposite corners z_1, z_2.

Theorem 4 *Let $f(s)$ be a holomorphic function on a rectangle R whose lower-left corner is z_1 and its upper-right corner is z_2. Let $z_0 = \frac{1}{2}(z_1 + z_2)$ be the central point of the rectangle, $r = \frac{1}{2}|z_2 - z_1|$ and $N(z_1, z_2)$ such that*

$$\max_{z \in R} \big|f''(z)\big| \leq N(z_1, z_2).$$

Then, for all $z \in R$

$$\big|f(z)\big| \geq \big|f(z_0)\big| - \big|f'(z_0)\big|r - \frac{1}{2}N(z_1, z_2)r^2. \tag{5}$$

Therefore, if the right-hand side of the last inequality is positive, then $f(s)$ does not have any zeros in R.

Proof As in the proof of Lemma 2, we require some tweaking to make the proof work. Let $u \in \mathbb{C}$ with $|u| = 1$, $z \in R$ and $f_u(t) = \big\langle f\big(z_0 + t(z - z_0)\big), u\big\rangle$, $0 \leq t \leq 1$. By Taylor's Theorem with Lagrange remainder, there exists $0 < t_u < 1$ such that $f_u(1) = f_u(0) + f_u'(0)\cdot 1 + \frac{1}{2}f_u''(t_u)$, which translates to

$$\big\langle f(z), u\big\rangle = \big\langle f(z_0), u\big\rangle + \big\langle f'(z_0)(z - z_0), u\big\rangle + \frac{1}{2}\Big\langle f''\big(z_0 + t_u(z - z_0)\big)(z - z_0)^2, u\Big\rangle.$$

Then $\big\langle f(z) - f(z_0) - f'(z_0)(z - z_0), u\big\rangle \leq \frac{1}{2}N(z_1, z_2)|z - z_0|^2$. By conveniently choosing u, we obtain $\big|f(z) - f(z_0) - f'(z_0)(z - z_0)\big| \leq \frac{1}{2}N(z_1, z_2)|z - z_0|^2$, hence (5) as expected. □

We note that the use of our exclusion criterion – namely, verifying the inequality $\big|f(z_0)\big| - \big|f'(z_0)\big|r - \frac{1}{2}N(z_1, z_2)r^2 > 0$ is optional and is intended to reduce the overall running time of the method. The effectiveness of this step depends on how sharp the bound N is.

3.3 Linear convergence

Now we show that the subdivision method is linearly convergent. Suppose that at the n-th iteration we have a rectangle R_n with sides $a \geq b > 0$, containing only one simple zero x^* of $f(s)$. The zero x^* is approximated by any point $x_n \in R_n$ with error $|x^* - x_n| \leq r_n$, where $r_n = \sqrt{a^2 + b^2}$. For the following subdivision $r_{n+1} = \sqrt{\left(\frac{a}{2}\right)^2 + b^2}$ and

$$\frac{r_{n+1}^2}{r_n^2} = \frac{a^2 + 4b^2}{4a^2 + 4b^2} \leq \frac{5}{8},$$

so $r_{n+1} \leq \sqrt{\frac{5}{8}} r_n$. In particular, we have linear convergence for $(r_n)_{n\in\mathbb{N}}$ going to 0. If two consecutive subdivision steps are considered, then regardless of the configuration, we obtain $r_{n+2} \leq \frac{1}{2} r_n$, which yields a sharper bound.

4 Zeros certification and acceleration

The two-dimensional bisection method is convergent, but slow, as it is only of linear convergence. To accelerate our computation, one we have a rectangle R containing exactly one zero, as we can ensure when working with functions with simple zeros, we can combine our main method with some faster methods, such as Newton's method. Starting from the center $z_0 \in R$, we iterate

$$z_{n+1} = z_n - \frac{f(z_n)}{f'(z_n)}, \quad n \geq 0$$

until $\big|f(z_n)\big| < \varepsilon$ or $|z_{n+1} - z_n| < \varepsilon$ for some $\varepsilon > 0$, choosing the appropriate stop criterion. As is well known, Newton's method converges quadratically, that is $|z_{n+1} - z^*| \leq C|z_n - z^*|^2$ for some constant $C > 0$ and $z^* \in R$ with $f(z^*) = 0$. However, Newton's method is only locally convergent, that is, it converges only when initialized sufficiently close to the solution; otherwise it may fail to converge or diverge altogether. For that reason, we add two extra (failing) stop criteria: either $n > n_0$ for some fixed iteration limit n_0 or $z_n \notin R$ for some $n > 0$. When failing, we use again the division method on our rectangle, then we apply Newton's method again. This way, we ensure global convergence for our method.

For zero certification, we invoke the following result from [23, Theorem 6, pp. 532–533].

Theorem 5 (Kantorovich and Akilov) *Let $f(z)$ be a holomorphic function on $\Omega \subset \mathbb{C}$, $z_0 \in \Omega$, $\eta > 0$, $K > 0$ and $\Omega_0 \subset \Omega$ such that the following conditions hold:*

1. *$f'(z_0) \neq 0$;*

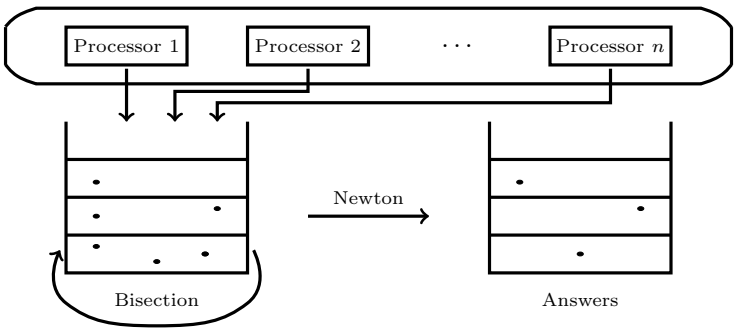


Fig. 3 Mutex stacks and parallel processing

2. $\left|f'(z_0)^{-1}f(z_0)\right| \leq \eta$;
3. $\left|f'(z_0)^{-1}f''(z)\right| \leq K$ *for* $z \in \Omega_0$.

Then, if $h = K\eta < \frac{1}{2}$ *and* $r \geq r_0 = \frac{1-\sqrt{1-2h}}{h}\eta$ *the equation* $f(z) = 0$ *has a solution* z^**, and Newton's method converges to this solution. Furthermore* $|z^* - z_0| \leq r_0$*. Also, if* $r < r_1 = \frac{1+\sqrt{1-2h}}{h}\eta$ *then* z^* *is the unique solution in* Ω_0*. The rate of convergence is given by*

$$|z_n - z^*| \leq \frac{1}{2^n}(2h)^{2^n} \cdot \frac{\eta}{h}, \quad n \geq 0.$$

The convergence of our method is given by the global convergence of the bisection method and the local convergence of Newton's method. By the preceding estimates, with the two-dimensional bisection method, the zeros of $f(s)$ within an initial rectangle R are enclosed in a finite number of rectangles R_n with diameter $\operatorname{diam}(R_n) \leq \sqrt{\frac{5}{8}}^n \operatorname{diam}(R)$, which can be made arbitrarily small. We consider a simple zero $z^* \in R_n$ of $f(s)$. As $f(z^*) = 0$ and $f'(z^*) \neq 0$, the uniform continuity of $f'(s)$ allows us to choose n sufficiently large such that $\left|f'(z)\right| \geq \frac{1}{2}\left|f'(z^*)\right|$. Taking $\Omega_0 = R_n$ and $K = \max_{z\in R_n}\left|f''(z)\right| \cdot \frac{2}{\left|f'(z^*)\right|}$ we obtain conditions 1 and 3 of Theorem 5. Now, by the uniform continuity of $f(s)$, for any fixed $0 < \eta < \frac{1}{2K}$, we can choose n sufficiently large such that $\left|f(z_0)\right| < \frac{2\eta}{\left|f'(z^*)\right|}$ for all $z_0 \in R_n$. This ensures condition 2 of Theorem 5. Finally, choosing $r = r_0$ we obtain the conclusions.

5 Implementation

The subdivision of the initial rectangle yields independent subproblems, which makes the method well suited for parallel implementation. Each resulting rectangle is pushed onto a stack and processed by an available processor. To prevent race conditions that could lead to data corruption, we employ mutex-protected stacks. Once Newton's method converges, the certified zeros are pushed onto the answers stack. The process is illustrated in Figure 3.

To handle multi-precision, we use the Boost C++ libraries [24]. The implementation requires several parameters, including various tolerance values `EPS`, `eps1`, `eps2`, `eps3` to handle round-off errors and inequality tolerances, a thread limit `limit` for parallelization, and the lower-left corner `LD` and upper-right corner `RU` of the initial rectangle.

The method algo requires the following functions: $f(z)$, its first derivative $df(z)$, a function $M(s_1, s_2)$ providing an upper bound for $\big|f''(s)\big|$ on $s \in [s_1, s_2]$, and a function $N(s_1, s_2)$ providing an upper bound for $\big|f''(s)\big|$ on the rectangle R_{s_1,s_2} with opposite corners s_1 and s_2.

6 Running and benchmarking

The implementation of our code with working examples can be found at https://github.com/racsosabe/PCZERO.

6.1 Exponential polynomials

We first describe the implementation strategy for exponential polynomials of the form

$$f(s) = \sum_{k=0}^{N} p_k(s) e^{-\lambda_k s}, \text{ where } 0 = \lambda_0 < \lambda_1 < \cdots < \lambda_N$$

are real numbers, and $p_k(s) = \sum_{j=0}^{d_k} a_{kj} s^j$, $a_{kj} \in \mathbb{C}$ for $0 \le j \le d_k$ and $a_{kd_k} \neq 0$. For the Newton method we will need the derivative $f'(s) = \sum_{k=0}^{N} \big(p_k'(s) - \lambda_k p_k(s)\big) e^{-\lambda_k s}$. We associate to a generic polynomial $q(s) = \sum_{j=0}^{d} \alpha_j s^j$ its derivative $q'(s) = 0$ if $d = 0$, and $q'(s) = \sum_{j=0}^{d-1} (j+1)\alpha_{j+1} s^j$ for $d \ge 1$. We also associate to it the (real) polynomial $|q|(s) = \sum_{j=0}^{d} |\alpha_j| s^j$, so $\big|q(s)\big| \le |q|\big(|s|\big)$. Then, we can choose $M(s_1, s_2) = \sum_{k=0}^{N} q_k\big(\max(|s_1|, |s_2|)\big)$, where $q_k(s) = |p_k'' - 2\lambda_k p_k' + \lambda_k^2 p_k|(s)$. We can also choose $N(s_1, s_2) = M(s_1, s_2)$ for the exclusion criterion.

We recall some facts about the distribution of zeros of exponential polynomials (cf. [3, §12.8]). Given an exponential polynomial $g(s) = \sum_{k=0}^{n} p_k(s) e^{\alpha_k s}$ where each $p_k(s)$ is a complex polynomial of degree m_k, and each α_k is a real number, $0 \le k \le n$, with $\alpha_0 < \alpha_1 < \cdots < \alpha_n$, we place the points (α_k, m_k), $0 \le k \le n$ in the plane, and we draw the polygonal line L corresponding to the upper boundary of the convex hull C of these points. This line is what is called the *distribution diagram* of the zeros of $g(s)$. The $r+1$ extreme points of C on L (α_{l_j}, m_{l_j}), $0 \le j \le r$, generate L in the sense that $L = \bigcup_{j=1}^{r} L_j$, where $L_j = \big[(\alpha_{l_{j-1}}, m_{l_{j-1}}), (\alpha_{l_j}, m_{l_j})\big]$. Here $(\alpha_{l_0}, m_{l_0}) = (\alpha_0, m_0)$ and $(\alpha_{l_r}, m_{l_r}) = (\alpha_n, m_n)$. If we denote by μ_j the slope of L_j, then $\mu_j = (m_{l_j} - m_{l_{j-1}})/(\alpha_{l_j} - \alpha_{l_{j-1}})$, $1 \le j \le r$ and $\mu_1 > \mu_2 > \cdots > \mu_r$.

The asymptotic behavior of the zeros is described in [3, Theorem 12.10] as follows: The zeros of $g(s)$ are asymptotically distributed along (curved or ordinary) bands $V_1, \ldots, V_r$ given by $V_j = \big\{s \in \mathbb{C} : a_j < \Re(s + \mu_j \log s) < b_j\big\}$ for suitable constants a_j, b_j, $1 \le j \le r$, where μ_j are the slopes of the segments L_j in the distribution diagram of $g(s)$. On each band V_j, the zeros of $g(s)$ are asymptotic to those of a *comparison*

function $g^*(s) = \sum_{k\in V_j} q_k s^{m_k} e^{\alpha_k s}$, where the sum is taken over the points (α_k, m_k) on the distribution diagram of $g(s)$ on L_j, and where $p_k(s) = q_k s^{m_k}\big(1 + o(1)\big)$. There exists $R > 0$ such that every zero with modulus $> R$ lies in one of these bands, and the number of zeros $\tilde{N}_j(T_1, T_2)$ in the rectangle (or curvilinear rectangle)

$$a_j < \Re(s + \mu_j \log s) < b_j, \quad T_1 < \Im(s + \mu_j \log s) < T_2$$

with $T_2 > T_1 > R$ and within $\{s \in \mathbb{C} : |s| > R\}$, satisfies the inequality

$$\left|\tilde{N}_j(T_1, T_2) - \frac{T_2 - T_1}{2\pi}(\alpha_{l_j} - \alpha_{l_{j-1}})\right| \le n_j - 1,$$

where n_j is the number of points of the distribution diagram on L_j, for $1 \le j \le n$.

If $\mu_j \neq 0$, the zeros on V_j lie on a finite number of curves of the form $se^{\alpha s} = w$, and they are *retarded-type* if $\mu_j > 0$, of *advanced-type* if $\mu_j < 0$, and of *neutral-type* if $\mu_j = 0$.

As a direct application of this result, consider the following. Define

$$f(s) = -1 + (-6 - 5s)e^{-2s} + (-2 + 3s)e^{-\frac{2\pi}{3}s} + 2e^{-\frac{5}{2}s} + se^{-3s} + 3e^{-4s}. \qquad (6)$$

The distribution diagram of this function (Figure 4) reveals zeros of all three types, along with one non-extremal point of the convex hull that does not contribute to the asymptotic distribution.

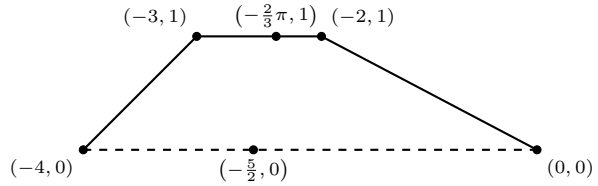


Fig. 4 Distribution diagram for $f(s)$ defined by (6)

6.2 Dirichlet polynomials

Since Dirichlet polynomials are of particular interest to us, we provide a complete implementation of this simple class of functions

$$f(s) = \sum_{k=0}^{n} a_k e^{-\lambda_k s}, \quad a_1, \ldots, a_n \in \mathbb{C}, \ 0 = \lambda_0 < \lambda_1 < \cdots < \lambda_n.$$

Such a function only possesses zeros of neutral type, as its distribution diagram is a horizontal segment. We use $M(s_1, s_2)$ as un upper bound for $\big|f''(s)\big|$ on $s \in [s_1, s_2]$, given

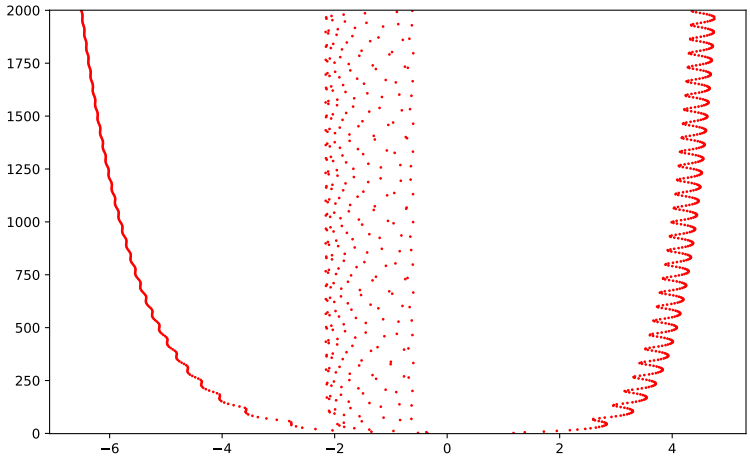


Fig. 5 1 272 Zeros of $f(s)$ with $-7 \leq \Re(s) \leq 4.75$, $0 \leq \Im(z) \leq 2\,000$

by $M(s_1, s_2) = \sum_{k=1}^{n} |a_k| \lambda_k^2 e^{-\min\{\Re(s_1), \Re(s_2)\}}$. We also take $N(s_1, s_2) = M(s_1, s_2)$ as an upper bound for $\left|f''(s)\right|$ on the rectangle with opposite corners s_1 and s_2.

Our program was compiled with Boost C++ Library version 1.83 and G++ version 15.2.1. The compiler flags used were `-O3 -pthread`, to enable multithreading and optimize for speed.

All benchmarks were obtained on a machine running Fedora Linux 43, equipped with an AMD Ryzen 9 7900 (12 cores / 24 threads, up to 5.49 GHz boost), 124 GiB of RAM and 14 GiB of swap. The `limit` parameter for the thread number was set equal to 24 and the `rho` constructor parameter was set to $\rho = 0.5$. We ran our program to compute the zeros of $\zeta_n(s) = \sum_{k=1}^{n} k^{-s}$, the n-th partial sum for the Riemann zeta function, for different values of n and inside different rectangles given, as functions of n and $T > 0$, by $1 - n \leq \Re(s) \leq 1.73, \quad 0 \leq \Im(s) \leq T$. To improve timing accuracy, each configuration was executed five times and the reported value is the mean of those runs. Indeed, all zeros are known to lie inside the vertical strip $1 - n \leq \Re(s) \leq 1.73$; see [25, Theorem 3.1].

Given $n \geq 3$ and $T > 0$, we consider the set of zeros of $\zeta_n(s)$ with $0 < \Im(s) < T$

$$\mathcal{Z}_n(T) = \{s \in \mathbb{C} : \zeta_n(s) = 0, 0 < \Im(s) < T\}$$

and $N_n(T) = \left|\mathcal{Z}_n(T)\right|$ the number of such zeros. We also define $\delta_n(T) = N_n(T) - \dfrac{T}{2\pi}\log(n)$, which satiafies $\left|\delta_n(T)\right| \leq n - 1$ by [3, Theorem 12.5]; the small magnitude of this quantity is directly related to the multiplicity of the zeros and is consistent with [26, Theorem 2]. We also compute $\alpha_n(T) = \max\{\Re(s) : s \in \mathcal{Z}_n(T)\}$, $\beta_n(T) = \min\{\Re(s) : s \in \mathcal{Z}_n(T)\}$ (cf. [27]) and $\mu_n(T) = \frac{1}{N_n(T)} \sum_{s \in \mathcal{Z}_n(T)} \Re(s)$.

With respect to the distribution of the real parts of the zeros, writing $\mathcal{Z}_n(T) = \{s_k\}_{k=1}^{N_n(T)}$ ordered in such a way that $\sigma_k = \Re(s_k)$, $1 \leq k \leq N_n(T)$ satisfies $\sigma_k \leq \sigma_{k+1}$,

$1 \leq k < N_n(T)$, and we consider $\min \delta\sigma_n(T) = \min\{\sigma_{k+1} - \sigma_k : 1 \leq k < N_n(T)\}$ and $\max \delta\sigma_n(T) = \max\{\sigma_{k+1} - \sigma_k : 1 \leq k < N_n(T)\}$. Also, considering the imaginary parts of the zeros, writing $\mathcal{Z}_n(T) = \{s_k\}_{k=1}^{N_n(T)}$ ordered in such a way that $\tau_k = \Im(s_k)$, $1 \leq k \leq N_n(T)$ satisfies $\tau_k \leq \tau_{k+1}$, $1 \leq k < N_n(T)$, we consider $\min \delta\tau_n(T) = \min\{\tau_{k+1} - \tau_k : 1 \leq k < N_n(T)\}$ and $\max \delta\tau_n(T) = \max\{\tau_{k+1} - \tau_k : 1 \leq k < N_n(T)\}$. The given quantities are presented in Table 1.

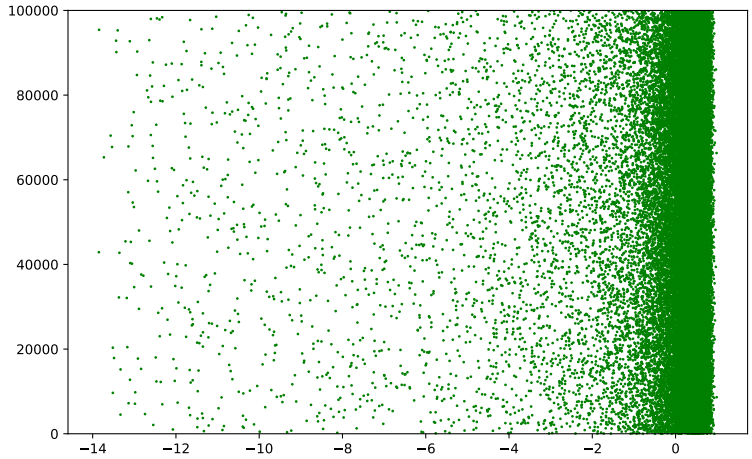


Fig. 6 49 195 zeros of $\zeta_{22}(s)$ with $0 < \Im(z) < 100\,000$

It should be noted that, for Dirichlet functions, the function $M(s_1, s_2)$ depends only on $\Re(s_1)$ and $\Re(s_2)$, so its evaluation on vertical segments can be cached, avoiding redundant computations. It is possible to activate or deactivate the exclusion criterion; doing so does not affect the convergence of the method; the decision depends on how sharp the bound $N(s_1, s_2)$ is. In addition, different choices of ρ may improve or worsen the execution time. An additional simulation was conducted with parameters $n = 3$, $T = 1000$, and $\rho \in [0.01, 0.99]$. The minimum execution time was 11.09 seconds at $\rho = 0.79$, while the maximum execution time was 132.18 seconds at $\rho = 0.02$.

Declarations

Funding

This work was supported by the Facultad de Ciencias of Universidad Nacional de Ingeniería.

Acknowledgments

The authors express their gratitude to the Research Institute of the Faculty of Sciences of Universidad Nacional de Ingeniería for supporting this project. The second

n	T	$N_n(T)$	time(s)	$\delta_n(T)$	$\beta_n(T)$	$\alpha_n(T)$	$\mu_n(T)$	$\min \sigma_n(T)$	$\max \sigma_n(T)$	$\min \tau_n(T)$	$\max \tau_n(T)$
3	1000000	174850	1489.32	0.42371697	−1.00000000	0.78788491	−0.00000091	1.225575×10^{-11}	0.00002310	3.83167301	7.19634299
4	1000000	220636	3011.13	0.39984735	−1.21428532	0.62628810	0.00000115	7.357559×10^{-12}	0.00003177	0.00009870	6.89831506
5	1000000	256150	4415.45	0.00063661	−2.42597696	0.89089957	0.00000493	4.153250×10^{-11}	0.00042293	0.00031971	6.56521252
6	1000000	285168	6608.41	0.62364064	−2.88647664	0.84108861	0.00000052	1.754143×10^{-11}	0.00079533	0.00006418	6.76595803
7	1000000	309702	8717.36	0.78096516	−3.81047760	0.97623623	−0.00000137	1.069589×10^{-12}	0.00320104	0.00004345	6.07532813
8	1000000	330954	11248.18	0.59977102	−4.38110194	0.91943250	0.00000334	3.364309×10^{-11}	0.00505737	0.00004557	6.13969130
9	1000000	349699	14648.34	−0.15256606	−5.01440205	0.89357429	−0.00000349	1.124956×10^{-11}	0.00258518	0.00000639	6.17689097
10	100000	36647	1574.34	0.22005603	−5.68901857	0.90046530	−0.00002288	5.531686×10^{-11}	0.01359666	0.00065393	5.79312944
11	100000	38164	1881.18	0.31143175	−6.55310090	0.96371716	−0.00002377	1.932520×10^{-10}	0.09015391	0.00066592	5.52636615
12	100000	39549	2129.53	0.48235643	−7.18825209	0.93339981	−0.00004664	7.411834×10^{-10}	0.05259597	0.00020812	5.71350868
13	100000	40823	2463.17	0.56309796	−7.86397794	0.98142405	−0.00003630	2.435044×10^{-10}	0.08652529	0.00086489	5.27617061
14	100000	42002	2923.73	0.09808888	−8.49802685	0.99087710	0.00006294	7.324141×10^{-10}	0.11269368	0.00008422	5.15513335
15	100000	43100	3316.53	0.04243536	−9.16079897	0.98360932	0.00003540	1.969671×10^{-9}	0.06517969	0.00010623	5.15571335
16	100000	44127	3667.48	−0.12003053	−9.95720322	0.96013638	0.00006072	1.960611×10^{-9}	0.10104791	0.00072903	5.21827839
17	100000	45092	3938.31	0.00914595	−10.54801160	0.97941850	−0.00000991	5.237668×10^{-10}	0.07162879	0.00040955	5.10735607
18	100000	46002	4269.84	0.30473576	−11.22879418	0.96297858	−0.00007933	6.540674×10^{-11}	0.15401478	0.00039163	5.04663405
19	100000	46862	4552.71	−0.20181668	−11.85128799	0.98976036	0.00005959	5.823875×10^{-11}	0.13461399	0.00002970	5.12549979
20	100000	47679	4912.09	0.44004840	−12.67997300	0.97272832	−0.00000189	5.657981×10^{-10}	0.17934729	0.00057088	5.04506204
21	100000	48455	5303.99	−0.07953179	−13.24835161	0.97892738	0.00000054	5.514315×10^{-10}	0.24514742	0.00017712	4.70492198
22	100000	49195	5699.51	−0.46857589	−13.85057415	0.98781722	0.00007760	6.966938×10^{-11}	0.16152972	0.00049236	4.78645566
23	100000	49903	6143.82	0.05964988	−14.43751957	1.00849693	−0.00000507	2.293833×10^{-10}	0.14731883	0.00031653	4.64994312
24	100000	50580	6024.99	−0.29765120	−15.28301058	0.98948294	0.00006180	3.566710×10^{-10}	0.22492063	0.00020253	4.71556140
25	100000	51230	6324.70	0.00012732	−15.96402456	0.98645703	−0.00005351	1.068932×10^{-9}	0.20469599	0.00002547	4.55774331
26	100000	51854	6711.55	−0.21690967	−16.61484182	0.98120460	0.00006083	3.074083×10^{-10}	0.24814251	0.00022288	4.67511426

Table 1 Some statistics of the zeros of $\zeta_n(s)$, $3 \leq n \leq 26$ with $0 < \Im(s) < T$. These zeros were computed within a guaranteed tolerance of 9.9×10^{-21} and $\rho = 0.5$

author wishes to thank the Postgraduate School of Universidad Nacional del Santa for providing excellent conditions for the development of his Ph.D. thesis. The third author gratefully acknowledges the financial support provided by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Fundação de Desenvolvimento da Unicamp (Funcamp), and BTG Pactual for funding his research during the course of his Master's studies at Universidade Estadual de Campinas (Unicamp). The authors also thank Daniel Saad Nogueira Nunes from the Instituto Federal de Educação, Ciência e Tecnologia de Brasília for kindly providing access to the server on which the computational experiments for this work were carried out.

Competing interests

The authors have no competing interests to declare that are relevant to the content of this article.

Data availability

The code supporting the results of this article is openly available at https://github.com/racsosabe/PCZERO.

Author contributions

All authors contributed to the conceptualization and methodology of the work. O. Velásquez Castañón led the development of the mathematical theory and the writing of the manuscript. M. Toribio Cangana and V. Galván Oyola contributed to the formal analysis and validation of the results. The software implementation was primarily carried out by V. Galván Oyola. All authors reviewed and approved the final manuscript.

References

[1] Ying, X., Katz, I.N.: A reliable argument principle algorithm to find the number of zeros of an analytic function in a bounded domain. Numer. Math. **53**(1-2), 143–163 (1988) https://doi.org/10.1007/BF01395882

[2] Heittokangas, J., Ishizaki, K., Tohge, K., Wen, Z.-T.: Value distribution of exponential polynomials and their role in the theories of complex differential equations and oscillation theory. Bull. Lond. Math. Soc. **55**(1), 1–77 (2023) https://doi.org/10.1112/blms.12719

[3] Bellman, R., Cooke, K.L.: Differential-difference Equations, p. 462. Academic Press, New York (1963)

[4] Jessen, B., Tornehave, H.: Mean motions and zeros of almost periodic functions. Acta Math. **77**, 137–279 (1945) https://doi.org/10.1007/BF02392225

[5] Amar, E., Matheron, É.: Analyse Complexe. Enseignement des mathématiques. Cassini, Paris (2004)

[6] Ying, X., Katz, I.N.: A simple reliable solver for all the roots of a nonlinear function in a given domain. Computing **41**(4), 317–333 (1989) https://doi.org/10.1007/BF02241221

[7] Kowalczyk, P.: Global complex roots and poles finding algorithm based on phase analysis for propagation and radiation problems. IEEE Transactions on Antennas and Propagation **66**(12), 7198–7205 (2018) https://doi.org/10.1109/TAP.2018.2869213

[8] Delves, L.M., Lyness, J.N.: A numerical method for locating the zeros of an analytic function. Math. Comput. **21**, 543–560 (1967) https://doi.org/10.2307/2004999

[9] Ioakimidis, N.I., Anastasselou, E.G.: A modification of the Delves-Lyness method for locating the zeros of analytic functions. J. Comput. Phys. **59**, 490–492 (1985) https://doi.org/10.1016/0021-9991(85)90125-1

[10] Kravanja, P., Van Barel, M., Ragos, O., Vrahatis, M.N., Zafiropoulos, F.A.: ZEAL: A mathematical software package for computing zeros of analytic functions. Comput. Phys. Commun. **124**(2-3), 212–232 (2000) https://doi.org/10.1016/S0010-4655(99)00429-4

[11] Kravanja, P., Van Barel, M.: Computing the Zeros of Analytic Functions. Lecture Notes in Mathematics, vol. 1727. Springer, Berlin (2000). https://doi.org/10.1007/BFb0103927

[12] Gillan, C.J., Schuchinsky, A., Spence, I.: Computing zeros of analytic functions in the complex plane without using derivatives. Comput. Phys. Commun. **175**(4), 304–313 (2006) https://doi.org/10.1016/j.cpc.2006.04.007

[13] Chen, H.: On locating the zeros and poles of a meromorphic function. J. Comput. Appl. Math. **402**, 11 (2022) https://doi.org/10.1016/j.cam.2021.113796 . Id/No 113796

[14] Ioakimidis, N.I.: Application of the generalized Siewert-Burniston method to locating zeros and poles of meromorphic functions. Z. Angew. Math. Phys. **36**, 733–742 (1985) https://doi.org/10.1007/BF00960384

[15] Dellnitz, M., Schütze, O., Zheng, Q.: Locating all the zeros of an analytic function in one complex variable. J. Comput. Appl. Math. **138**(2), 325–333 (2002) https://doi.org/10.1016/S0377-0427(01)00371-5

[16] Frantsuzov, V.A., Artemyev, A.V.: A global argument-based algorithm for finding complex zeros and poles to investigate plasma kinetic instabilities. J. Comput. Appl. Math. **456**, 19 (2025) https://doi.org/10.1016/j.cam.2024.116217 . Id/No 116217

[17] García-Zapata, J.L., Díaz Martín, J.C., Cortés Fácila, A.: An adaptive subdivision method for root finding of univariate polynomials. J. Comput. Appl. Math. **352**, 146–164 (2019) https://doi.org/10.1016/j.cam.2018.11.023

[18] Yakoubsohn, J.-C.: Numerical analysis of a bisection-exclusion method to find zeros of univariate analytic functions. J. Complexity **21**(5), 652–690 (2005) https://doi.org/10.1016/j.jco.2005.06.007

[19] Semenov, V.: Method for the calculation of all zeros of an analytic function based on the Kantorovich theorem. Computational Methods in Applied Mathematics **14**(3), 385–392 (2014) https://doi.org/10.1515/cmam-2014-0011

[20] Brézis, H.: Functional Analysis, Sobolev Spaces and Partial Differential Equations vol. 2. Springer, New York (2011)

[21] Stoer, J.., Bulirsch, R.: Introduction to Numerical Analysis. Texts in applied mathematics. Springer, New York (2002)

[22] Nocedal, J., Wright, S.J.: Numerical Optimization. Springer, New York (2006)

[23] Kantorovich, L.V., Akilov, G.P.: Functional Analysis (translated by Howard L. Silcock). Pergamon Press, New York (1982)

[24] Schäling, B.: The Boost C++ Libraries. XML Press, Santa Clara, CA (2014)

[25] Borwein, P., Fee, G., Ferguson, R., Waall, A.: Zeros of partial sums of the Riemann zeta function. Exp. Math. **16**(1), 21–39 (2007) https://doi.org/10.1080/10586458.2007.10128981

[26] Mora, G., Sepulcre, J.M.: Computing the zeros of the partial sums of the Riemann zeta function. Ann. Mat. Pura Appl. (4) **194**(5), 1499–1504 (2015) https://doi.org/10.1007/s10231-014-0430-6

[27] Balazard, M., Velásquez Castañón, O.: About the infimum of the real parts of the zeros of the partial sums of the Riemann zeta-function. C. R., Math., Acad. Sci. Paris **347**(7-8), 343–346 (2009) https://doi.org/10.1016/j.crma.2009.02.008