UPPSALA
UNIVERSITET

U.U.D.M. Project Report 2016:42

Enclosing Zeros for Systems of Two Analytic Functions

Joel Dahne

Examensarbete i matematik, 15 hp
Handledare: Warwick Tucker
Examinator: Jörgen Östensson
Augusti 2016

Department of Mathematics
Uppsala University

Enclosing Zeros for Systems of Two Analytic Functions

Joel Dahne

August 24, 2016

Abstract

We present a method for finding and validating simple zeros to systems of two analytic functions. It can also prove that all zeros have been found by calculating the total number of zeros in the domain. The last part turns out to be very time consuming and not always applicable in practical examples. To illustrate the method we apply it to a non trivial academic example and also on a physical problem for which no effective methods have been found before.

1 Introduction

Finding zeros for analytic functions in one variable is a well studied, and important, problem. For polynomials there exists very fast and accurate methods, see for example [4, 9]. In the case of general analytic functions there are also methods developed. One such method, that also produces validated results, is presented in [5].

For systems of analytic functions the problem of finding approximations of the zeros is not as well studied. In [6] they present a method that can generate approximations of the zeros. They use a multidimensional version of the logarithmic integral together with other methods to do this. However they conclude that proper evaluation of the method is hard, especially evaluation of the integral.

In this paper we will present another method for finding approximations of the zeros. It will also make use of the multidimensional logarithmic integral but combined with a random search for the zeros.

In addition we will make use of validated numerics to achieve rigorous results. The method should generate a number of small boxes which are proved to contain a zero of the function. We also wish to be able to prove that all zeros are found. A short introduction to validated numerics and how this can be achieved will be given.

2 The General Strategy

We will divide the task of finding all the zeros into two parts.

1. Calculate the number of zeros in the domain

2. Perform an exhaustive random search for these zeros

The calculation for the number of zeros will be done using the logarithmic integral and will give us a stop criterion for the random search. For the random search we will make use of Newton's method and also a validated version of Newton's method presented in section 3.4.2. This will give us an increasing number of verified zeros and when the number of zeros is equal to the calculated number of zeros in the domain we can stop.

2.1 Calculating the Number of Zeros

To be able to calculate the number of zeros of the function we will use that it is analytic.

In one complex variable the logarithmic integral gives us a method to find the number of zeros for a function in a domain.

Theorem 1. *Let f be analytic on the closure of a bounded domain $D \subset \mathbb{C}$ with piece wise smooth boundary that does no contain any zeros of f. Then the number of zeros to f, counting multiplicity, is given by*

$$\frac{1}{2\pi i}\int_{\partial D}\frac{f'(z)}{f(z)}dz. \tag{1}$$

This can also be extended to meromorphic functions, but we will make no use of that here.

Since we will work with systems of analytic functions we will need a generalization of this formula. Such a generalization exists in the literature and is given in the following theorem.

Theorem 2. *Let $f = f(z_1, ..., z_n)$ be analytic on the closure of a bounded domain $D \subset \mathbb{C}^n$ with piece wise smooth boundary that does not contain any zeros of f. Then f has only isolated zeros in D, and their number, counting multiplicity, is expressed by the formula*

$$N = \int_{\partial D}\omega(f(z), \overline{f(z)})), \tag{2}$$

where

$$\omega(f, \overline{f}) = \frac{(n-1)!}{(2\pi i)^n}\frac{1}{|f|^{2n}}\sum_{j=1}^{n}(-1)^{j-1}\overline{f}_j d\overline{f}_{[j]} \wedge df. \tag{3}$$

For a proof of this we refer to [1].

The integral is given in differential forms and the notation used is

$$df = df_1 \wedge ... \wedge df_n,$$
$$df_{[k]} = df_1 \wedge ... \wedge df_{k-1} \wedge df_{k+1} \wedge ... \wedge df_n.$$

We want to be able to compute the integral numerically and to prepare it for this we will rewrite it as a Riemann integral which is more suitable for numerical integration. To accomplish this we will follow the procedure in [6], where more details can be found.

With the boundary of D given by

$$\partial D = \partial D_{[1]} \cup ... \cup \partial D_{[n]} \tag{4}$$

where

$$\partial D_{[k]} = D_1 \times \cdots \times D_{k-1} \times \partial D_k \times D_{k+1} \times \cdots \times D_n. \tag{5}$$

We can rewrite the formula as

$$\sum_{k=1}^{n} \int_{\partial D_{[k]}} \omega_k(f, \overline{f}), \tag{6}$$

where

$$\omega_k = (-1)^{k-1} \frac{(n-1)!}{(2\pi i)^n} \frac{\det J_f \overline{\det J_{[k]}}}{|f|^{2n}} d\overline{z}_{[k]} \wedge dz. \tag{7}$$

Here J_f denotes the Jacobian with respect to f and $J_{[k]}$ is the Jacobian matrix with the kth column replaced with $[f_1, ..., f_n]^T$.

Since we will only be working with systems of two analytic functions we can let $n = 2$, which gives us

$$\frac{1}{4\pi^2}\Bigg(\int_{\partial D_{[2]}} \frac{\det J_f \overline{\det J_{[2]}}}{|f|^4} d\overline{z}_1 dz_1 dz_2 - \int_{\partial D_{[1]}} \frac{\det J_f \overline{\det J_{[1]}}}{|f|^4} d\overline{z}_2 dz_1 dz_2\Bigg). \tag{8}$$

When doing the computations it will be beneficial to consider it as a function of 4 real variables instead of 2 complex. By changing variables from z_1 and z_2 to $x_1 + iy_1$ and $x_2 + iy_2$ we get

$$\begin{aligned}\frac{1}{4\pi^2}\Bigg(\int_{\partial D_{[2]}} \frac{\det J_f \overline{\det J_{[2]}}}{|f|^4} (dx_1 - idy_1) \wedge (dx_1 + idy_1) \wedge (dx_2 + idy_2) - \\ - \int_{\partial D_{[1]}} \frac{\det J_f \overline{\det J_{[1]}}}{|f|^4} (dx_2 - idy_2) \wedge (dx_1 + dy_1) \wedge (dx_2 + dy_2)\Bigg).\end{aligned} \tag{9}$$

Expanding the differentials we get

$$\begin{aligned}(dx_1 - idy_1) \wedge (dx_1 + idy_1) \wedge (dx_2 + idy_2) = \\ (dx_1 \wedge dx_1 + i(dx_1 \wedge dy_1) - i(dy_1 \wedge dx_1) + dy_1 \wedge dy_1) \wedge (dx_2 + idy_2) = \\ 2i(dx_1 \wedge dy_1) \wedge (dx_2 + idy_2) = \\ 2i(dx_1 \wedge dy_1 \wedge dx_2 + i(dx_1 \wedge dy_1 \wedge dy_2))\end{aligned} \tag{10}$$

and

$$\begin{aligned}(dx_2 - idy_2) \wedge (dx_1 + idy_1) \wedge (dx_2 + idy_2) = \\ -(dx_2 - idy_2) \wedge (dx_2 + idy_2) \wedge (dx_1 + idy_1) = \\ -(dx_2 \wedge dx_2 + i(dx_2 \wedge dy_2) - i(dy_2 \wedge dx_2) + dy_2 \wedge dy_2) \wedge (dx_1 + idy_1) = \\ -2i(dx_2 \wedge dy_2) \wedge (dx_1 + idy_1) = \\ -2i(dx_2 \wedge dy_2 \wedge dx_1 + i(dx_2 \wedge dy_2 \wedge dy_1)).\end{aligned} \tag{11}$$

This gives us the integral formula

$$\begin{aligned}\frac{i}{2\pi^2}\Bigg(\int_{\partial D_{[2]}} \frac{\det J_f \overline{\det J_{[2]}}}{|f|^4} dx_1 dy_1 dx_2 + i \int_{\partial D_{[2]}} \frac{\det J_f \overline{\det J_{[2]}}}{|f|^4} dx_1 dy_1 dy_2 - \\ - \int_{\partial D_{[1]}} \frac{\det J_f \overline{\det J_{[1]}}}{|f|^4} dx_2 dy_2 dx_1 - i \int_{\partial D_{[1]}} \frac{\det J_f \overline{\det J_{[1]}}}{|f|^4} dx_2 dy_2 dy_1\Bigg).\end{aligned} \tag{12}$$

2.2 Random Search for Zeros

When the number of zeros is known what is left is to find them. As mentioned we will make use of Newton's method and a validated version of it to accomplish this. Newton's method will be used to generate possible zeros, points that are likely to be close to a zero. Then the validated version is applied to check if the points indeed is close to a zero.

For Newton's method we will start by choosing a random point, z_0, in the domain and then generate the sequence

$$z_{n+1} = z_n - J_f(z_n)^{-1} f(z_n) \quad n = 0, 1, 2, ... \tag{13}$$

which normally converges to a zero of f within a few iterations. This convergence is however not guaranteed and we have no information of how good of an approximation to a zero the new point actually is.

Because of this we will need to check and validate the generated point. Using the validated version of Newton's method we can determine if the generated point indeed is a good approximation of a zeros, more specifically we are able to prove that a small box around the point contains a zero of the function.

If the point is determined to be a good approximation of a zero we add it to the list of zeros, if it is not then it is discarded. We can then continue to generate points and try to validate them until the number of zeros in the list is equal to the calculated number of zeros in the domain.

3 Validated Numerics

Computers are fast, they can perform billions of operations per second and with them we can accomplish tasks otherwise not possible. But when it comes to mathematics, computers are not always reliable. A computer does not know what a real number is and cannot calculate with real numbers. All the computer gives us is an approximation, often times it is good enough though. But to be able to generate mathematical proofs involving the continuum with the help of a computer we need other tools. For this we will use *validated numerics* and we will here give a short introduction to it, for a more complete one see for example [10].

For simplicity we will in this introduction consider validated numerics only on real numbers. The step from $\mathbb{R}$ to $\mathbb{C}^2$ is not very big and a short discussion of it will be given in the end.

3.1 Floating Points

To start with we will give an introduction to floating point numbers, the computers way of handling real numbers. We will not go into how they are constructed or how the computations are done, but will instead talk about some of their properties and the problems they cause.

The floating point numbers are a finite subset of the real numbers and denoted by $\mathbb{F}$. Around the origin they are rather dense and then become more and more sparse the further from the origin, see figure 1. The exact number of floating points depends on which format is used.

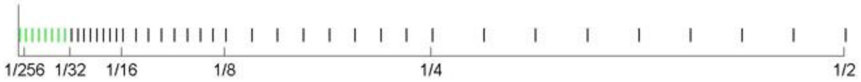


Figure 1: The distribution of floating point numbers.

The problem with computing with floating points is that when the exact result of a computation cannot be represented in the format we have to round off to the closest number that we can represent. Even though the round off for one computation may be small we get problems when doing many computations together, each one introducing a small error.

Example 1. *Consider the first one million terms in the harmonic series. For a mathematician it is clear that*

$$\frac{1}{1}+\frac{1}{2}+\frac{1}{3}+\dots+\frac{1}{10^6}=\frac{1}{10^6}+\dots+\frac{1}{2}+\frac{1}{1}, \tag{14}$$

because addition is associative and commutative. But computing this on a computer using a single precision format we get

$$\begin{aligned}\frac{1}{1}+\frac{1}{2}+\frac{1}{3}+\dots+\frac{1}{10^6} &=14.357357,\\ \frac{1}{10^6}+\dots+\frac{1}{3}+\frac{1}{2}+\frac{1}{1} &=14.392651.\end{aligned}$$

Clearly they are not equal and it is hard to tell how far from the real answer they are. It is obvious that this is not a property that is very good when trying to prove something about the result.

In the following sections we will introduce a method that will help us keep track of the round off error and it will allow us to actually prove things about the result.

3.2 Interval Arithmetic

At the heart of validated numerics lies the concept of intervals and interval arithmetic, essentially the use of intervals will let us keep track of the errors introduced in the computations. We will begin by giving an elementary introduction to interval arithmetic on $\mathbb{R}$.

Let $\mathbb{IR}$ denote the set of all closed intervals on $\mathbb{R}$, i.e. $\mathbb{IR} = \{[a,b] : a \leq b, a, b \in \mathbb{R}\}$. We will use the notation $\mathbf{x} = [\underline{x}, \overline{x}]$ to denote an interval and its endpoints $\underline{x}$, $\overline{x}$ respectively.

Let $\star$ denote one of the standard arithmetic operations, i.e. $\star \in \{+, -, \times, \div\}$. We can then define $\mathbf{x} \star \mathbf{y}$, for $\mathbf{x}, \mathbf{y} \in \mathbb{IR}$, by $\mathbf{x} \star \mathbf{y} = \{x \star y : x \in \mathbf{x}, y \in \mathbf{y}\}$ where we demand that $0 \notin \mathbf{y}$ for division. What might not be immediately clear is that this will always return a new interval. The endpoints for this new interval can be characterized by

$$\begin{aligned}\mathbf{x}+\mathbf{y} =&[\underline{x}+\underline{y},\overline{x}+\overline{y}],\\ \mathbf{x}-\mathbf{y} =&[\underline{x}-\overline{y},\overline{x}-\underline{y}],\\ \mathbf{x}\times\mathbf{y} =&[\min(\underline{x}\underline{y},\underline{x}\overline{y},\overline{x}\underline{y},\overline{x}\overline{y}),\max(\underline{x}\underline{y},\underline{x}\overline{y},\overline{x}\underline{y},\overline{x}\overline{y}),\\ \mathbf{x}\div\mathbf{y} =&\mathbf{x}\times[1/\overline{y},1/\underline{y}].\end{aligned}$$

This gives us an extension of the standard arithmetic operations on $\mathbb{R}$ to $\mathbb{IR}$.

Next we want to be able to extend functions on $\mathbb{R}$ to functions on $\mathbb{IR}$. We want to be able to enclose the range of the function by an interval.

Definition 1. *Let $D \subset \mathbb{R}$, and consider a function $f : D \to \mathbb{R}$. We define the range of f over D to be the set*

$$R(f, D) = \{f(x) : x \in D\}.$$

However, except in very simple cases, finding the exact range of a function is hard. There is an entire branch in mathematics – optimization theory – devoted to finding only the smallest element in that set.

For rational functions, functions on the form $f(x) = p(x)/q(x)$ where p and q are polynomials, we can create an extension by simply substituting all instances of the real variable x with the interval variable $\mathbf{x}$. This produces a rational interval function, $F(\mathbf{x})$ called the *natural* interval extension of f.

We also want find interval extensions of more general functions. For monotonic functions this is easy, for example we may define

$$e^{\mathbf{x}} = \exp\mathbf{x} = [e^{\underline{x}}, e^{\overline{x}}].$$

Other functions can also be extended. Especially the set of standard functions

$$\begin{aligned}\mathfrak{S} =&\{a^x, \log_a x, x, x^{p/q}, \mathit{abs}\ x, \sin x, \cos x, \tan x, ...\\ &..., \sinh x, \cosh x, \tanh x, \arcsin x, \arccos x, \arctan x\}\end{aligned}$$

can be extended to interval functions. By doing this we can find interval extensions to all functions expressed as a finite number of standard functions combined with arithmetic operations, constants and composition. These kind of functions are often called elementary functions and the set of all elementary functions is denoted by $\mathfrak{E}$.

Theorem 3 (**The Fundamental Theorem of Interval Analysis**)**.** *Given an elementary function f and a natural interval extension F such that $F(\mathbf{x})$ is well-defined for some $\mathbf{x} \in \mathbb{IR}$, we have*

$$\begin{aligned}&(1)\mathbf{z} \subseteq \mathbf{z}' \subseteq \mathbf{x} \implies F(\mathbf{z}) \subseteq F(\mathbf{z}')(\textit{inclusion isotonicity})\\ &(2)R(f;\mathbf{x}) \subseteq F(\mathbf{x})(\textit{range enclosure})\end{aligned}$$

While finding the exact range of a function is hard we can, by evaluating the interval extension of function, find an enclosure of the range. With the help of Theorem 2 we can then say something about what is *not* in the range of the function.

Example 2. *Let $f(x) = (\sin x - x^2 + 1)\cos x$. Prove that f has no zeros in* $[0, \frac{1}{2}]$.

By substituting x with $\mathbf{x} = [0, \frac{1}{2}]$ and f with its natural interval extension F we end up with

$$\begin{aligned} F([0, \frac{1}{2}]) &= (\sin[0, \frac{1}{2}] - [0, \frac{1}{2}]^2 + 1)\cos[0, \frac{1}{2}] \\ &= ([0, \sin\frac{1}{2}] - [0, \frac{1}{4}] + [1, 1])[\cos\frac{1}{2}, 1] \\ &= [\frac{3}{4}, 1][\cos\frac{1}{2}, 1] = [\frac{3}{4}\cos\frac{1}{2}, 1 + \sin\frac{1}{2}] \\ &\subseteq [0.65818, 1.4795]. \end{aligned}$$

Since $0 \notin F([0, \frac{1}{2}])$, it follows that $0 \notin R(f, [0, \frac{1}{2}])$ and f has no roots in the domain.

3.3 Directed Rounding

We have now discussed the basics of interval arithmetic using real numbers. However, since we are doing computations on a computer, we will in practice be limited to floating point numbers. Therefor we will discuss how we can limit the domain to floating points and still keep the wanted properties such as *inclusion isotonicity* and *range enclosure*.

Instead of working with real intervals, $\mathbb{IR}$, we will now work with floating point intervals, $\mathbb{IF} = \{[\underline{x}, \overline{x}] : \underline{x} \leq \overline{x},\ \underline{x}, \overline{x} \in \mathbb{F}\}$, recall that $\mathbb{F}$ is the set of floating point numbers. To do this we will use two modes of rounding, *round towards minus infinity* and *round towards infinity* denoted by $\bigtriangledown(x)$ and $\bigtriangleup(x)$. In general the left point of the interval will be rounded towards minus infinity and the right points towards infinity. For example the addition will be implemented as

$$\mathbf{x} + \mathbf{y} = [\bigtriangledown(\underline{x} + \underline{y}), \bigtriangleup(\overline{x} + \overline{y})]. \tag{15}$$

This ensures us that the floating point interval contains the corresponding real interval.

Example 3. *Lets, as in example 1, again consider the first one million terms in the harmonic series. This time we add them together using directed rounding instead and get an interval proved to contain the exact result. For the lower part of the interval all roundings were done towards minus infinity, giving us a lower bound for the result, and for the upper part towards infinity, giving us an upper bound.*

$$\begin{aligned} \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{10^6} &\in [13.971489, 14.924348], \\ \frac{1}{10^6} + \cdots + \frac{1}{3} + \frac{1}{2} + \frac{1}{1} &\in [14.350338, 14.436089]. \end{aligned}$$

Both of them give us an interval containing the exact result, but the later is to prefer since it gives a narrower interval.

With this directed rounding together with interval arithmetic we are able have the computer generate a guaranteed enclosure of a function even though we are limited to floating point numbers.

3.4 Validated Methods

Using interval arithmetic we have a method of enclosing the exact value of a computation, with the help of this we can get rigorous results. Most numerical methods however does not only suffer from floating point errors but also discretization errors from the method itself. To truly get rigorous results we need adapt the methods we use to account for the discretization error as well. We will describe two validated methods, one for computing definite integrals and one for finding zeros, that will be used in the coming algorithm.

3.4.1 Computing Integrals

There are several methods developed for computing definite integrals. From the very simple *midpoint* method and *trapezoid* method to the slightly more complex *Simpson's* method. Even tough all of them many times give a good approximation of the integral they are not rigorous. We present a method that can give us a guaranteed enclosure of the integral.

Let f be a real valued function which admits a well-defined interval extension F over the integration domain $[a, b]$. We then have the enclosure of the integral I

$$I \in w([a, b])F([a, b]), \tag{16}$$

where $w([a, b]) = b - a$ denotes the width of the interval. In most cases this will however produce a terrible estimate of the integral. By partitioning the domain $[a, b]$ into several smaller intervals we can produce a much better estimate. Given a partition $a = x_0 \leq x_1 \leq \cdots \leq x_N = b$ we can decompose the domin,

$$[a, b] = [x_0, x_1] \cup [x_1, x_2] \cup \cdots \cup [x_{N-1}, x_N]. \tag{17}$$

Now we get the estimate

$$I \in \sum_{i=1}^{N} w([x_{i-1}, x_i])F([x_{i-1}, x_i]). \tag{18}$$

It can be noted that this method is very similar to the definition of the Riemann-Stieltjes integral.

Even if this method can be used to generate a tight enclosure of the integral it converges very slowly. It can be shown to have linear convergence rate which is very poor compared to, for example, Simpson's method which has a quartic convergence rate. Even though there are validated methods for computing integrals that are much faster we will actually stick with this one in the end. The main reason for this is that it is easily adapted to work for integrals in more than one dimension and does not require that the function is differentiable.

3.4.2 A Validated Newton's method

We have already given a short presentation of Newton's method in section 2.2, it gives us a way of finding approximations of the zeros of a function. Now we will introduce a variant of Newton's method that uses intervals instead of real numbers. It will give us a way of finding guaranteed enclosures of zeros and even proving that a interval contains exactly one zero of the function.

Definition 2 (**The Interval Newton Operator**)**.** *For a continuous and differentiable function $f : \mathbf{x} \to \mathbb{R}$ with the interval extension F we define the interval Newton method*

$$N(\mathbf{x}) = m - \frac{f(m)}{F'(\mathbf{x})}. \tag{19}$$

Where m denotes the midpoint of the interval $\mathbf{x}$.

Much like for the regular Newton method we can define a sequence, only now it will be a sequence of intervals. Starting with an interval $\mathbf{x}_0$ for which $N(\mathbf{x}_0)$ is well-defined, i.e. $0 \notin F'(\mathbf{x}_0)$, we define the sequence

$$\mathbf{x}_{k+1} = N(\mathbf{x}_k) \cap \mathbf{x}_k, \quad k = 0, 1, 2, ... \tag{20}$$

This sequence has several nice properties that will allow us to find zeros to the function.

Theorem 4. *Assume that $N(\mathbf{x}_0)$ is well-defined. If $\mathbf{x}_0$ contains a zero, x^*, of f then so does $\mathbf{x}_k$ for all $k \in \mathbb{N}$. The sequence $\{\mathbf{x}_k\}_{k=1}^{\infty}$ forms a nested sequence converging to x^*.*

For a proof of this we refer to [10]. So like Newton's method this sequence allows us to get better and better approximations of the zero, only this time we get a guaranteed enclosure of the zero. This is however not the only usable property of the sequence. What we will use it for the most is its ability to prove existence of a unique zero in an interval.

Theorem 5. *Assume that f is twice continuous differentiable, and that $N(\mathbf{x})$ is well-defined. Then the following statements hold*

1. *if $N(\mathbf{x}) \cap \mathbf{x} = \emptyset$, then $\mathbf{x}$ contains no zeros of f;*
2. *if $N(\mathbf{x}) \subseteq \mathbf{x}$, then $\mathbf{x}$ contains exactly one zero of f.*

The first statement clearly follows from Theorem 3. For a proof of the second statement we again refer to [10].

One thing to note here is that the method can never prove the existence of a root of higher multiplicity. This is simply because by definition the derivative at these roots is equal to zero and the method requires that $0 \notin F'(\mathbf{x})$. Thus we will only be able to locate simple zeros to the function.

We at least have a method for proving the existence of a unique simple zero in an interval and in the algorithm we will use an higher order version of this for validating the zeros.

3.5 The Step to $\mathbb{C}^2$

We will here give a short description of the step from handling real intervals to working with intervals on $\mathbb{C}^2$. For the extension to $\mathbb{C}$ we will represent a complex interval with two real intervals as

$$\mathbf{z} = \mathbf{x} + i\mathbf{y} = \{x + iy | x \in \mathbf{x}, y \in \mathbf{y}\}, \tag{21}$$

so an interval in $\mathbb{C}$ will be a rectangle. When taking the step from $\mathbb{C}$ to $\mathbb{C}^2$ we no longer talk about intervals but instead about boxes, a box will be represented by a product of intervals. So the boxes in $\mathbb{C}^2$ will be given by

$$\mathbf{z}_1 \times \mathbf{z}_2 = \mathbf{x}_1 + i\mathbf{y}_1 \times \mathbf{x}_2 + i\mathbf{y}_2 = \{(x_1+iy_1, x_2+iy_2) | x_1 \in \mathbf{x}_1, y_1 \in \mathbf{y}_1, x_2 \in \mathbf{x}_2, y_2 \in \mathbf{y}_2\}. \tag{22}$$

Both the methods described above extends naturally to these kind of boxes. For calculating the integral all we have to change is that instead of using the width in equation 16 we use the volume of the box. In the case of the validated Newton's method we substitute the derivative with the Jacobian in equation 19 and get

$$N(\mathbf{z}) = m - J_F(\mathbf{z})^{-1} f(m), \tag{23}$$

here $\mathbf{z}$ is a box in $\mathbb{C}^2$ and m denotes the midpoint of it.

4 The Algorithm

We are now ready to go through the algorithm for finding zeros to a system of two analytic functions.

For a domain $D \subset \mathbb{C}^2$ let $f = (f_1, f_2) : \overline{D} \to \mathbb{C}^2$ be an analytic function with no zeros on the boundary ∂D. The methods will require that we can find an enclosure to the range of both f and its first order partial derivatives $\frac{\partial f_1}{\partial z_1}, \frac{\partial f_1}{\partial z_2}, \frac{\partial f_2}{\partial z_1}, \frac{\partial f_2}{\partial z_2}$. Because of this we will also require that we can find an interval extension to all of these. In practice this will be automated by the use of different libraries.

Since we will make use of intervals we will also require that the domain D is a box in $\mathbb{C}^2$, i.e. given by a product of intervals. So it will be on the form

$$D = \mathbf{z}_1 \times \mathbf{z}_2, \tag{24}$$

where $\mathbf{z}_1 = \mathbf{x}_1 + i\mathbf{y}_1$ and $\mathbf{z}_2 = \mathbf{x}_2 + i\mathbf{y}_2$ with $\mathbf{x}_1, \mathbf{y}_1, \mathbf{x}_2, \mathbf{y}_2 \in \mathbb{IR}$.

4.1 Integration

We want to compute the integral given in equation 12,

$$\frac{i}{2\pi^2}\Bigg(\int_{\partial D_{[2]}} \frac{\det J_f \overline{\det J_{[2]}}}{|f|^4} dx_1 dy_1 dx_2 + i \int_{\partial D_{[2]}} \frac{\det J_f \overline{\det J_{[2]}}}{|f|^4} dx_1 dy_1 dy_2 - \\ - \int_{\partial D_{[1]}} \frac{\det J_f \overline{\det J_{[1]}}}{|f|^4} dx_2 dy_2 dx_1 + i \int_{\partial D_{[1]}} \frac{\det J_f \overline{\det J_{[1]}}}{|f|^4} dx_2 dy_2 dy_1 \Bigg).$$

With D being a box in $\mathbb{C}^2$ the boundary will be given by 8 cuboids, $D_{[1]}$ and $D_{[2]}$ will represent 4 cuboids each. These will be given by

$$\begin{aligned}
D_1 &= \underline{x}_1 + i\mathbf{y}_1 \times \mathbf{x}_2 + i\mathbf{y}_2,\\
D_2 &= \overline{x}_1 + i\mathbf{y}_1 \times \mathbf{x}_2 + i\mathbf{y}_2,\\
D_3 &= \mathbf{x}_1 + i\underline{y}_1 \times \mathbf{x}_2 + i\mathbf{y}_2,\\
D_4 &= \mathbf{x}_1 + i\overline{y}_1 \times \mathbf{x}_2 + i\mathbf{y}_2,\\
D_5 &= \mathbf{x}_1 + i\mathbf{y}_1 \times \underline{x}_2 + i\mathbf{y}_2,\\
D_6 &= \mathbf{x}_1 + i\mathbf{y}_1 \times \overline{x}_2 + i\mathbf{y}_2,\\
D_7 &= \mathbf{x}_1 + i\mathbf{y}_1 \times \mathbf{x}_2 + i\underline{y}_2,\\
D_8 &= \mathbf{x}_1 + i\mathbf{y}_1 \times \mathbf{x}_2 + i\overline{y}_2.
\end{aligned}$$

The integration will be computed for each of these independently and then summed up. Every domain D_i will be part of two integrals. For one of these integrals the variable held constant will also be one of the variables integrated over and the integral will therefor evaluate to zero. So in the end we get only 8 integrals to compute, one for each D_i.

For computing the integral we will use the method described in section 3.4.1. Since we are now dealing with 3-dimensional integrals instead of 1-dimensional we will as mention have to adjust the method slightly. We will instead of multiplying with the width of the interval in equation 16 multiply with the volume of the cuboid, $V(D_i)$. For finding an interval extension of the integrand we simply use

$$\frac{\det J_F \overline{\det J_{[k]}}}{|F|^4}, \tag{25}$$

where $k = 1$ for $D_1, ..., D_4$ and $k = 2$ for $D_5, ..., D_8$. Note that we here need the interval extension of f and its first order partial derivatives to compute the Jacobian.

This will give us an enclosure of the integral over the cuboid, the whole integral will then be enclosed by the sum of the enclosures for the 8 cuboids. Since the integral represents the number of zeros for f in D we know that the value should be a natural number. So if the enclosure of the integral ever contain only one natural number we know that this must be the exact value. Since every cuboid represents one eighth of the integral we will require that the width of the enclosure should be no larger than 1/8, so that the sum has a width lower than 1 and therefor contains exactly one integer, the number of zeros. To achieve this we will have to split each cuboid several times so that the enclosure can become sufficiently tight.

Since we require that there are no zeros of f on the boundary of D we will never have the problem that the denominator $|f|^4$ is equal to zero. It might however happen that the enclosure $|F|^4$ does contain zero, especially if a zero of f is close to the boundary. Since we have no definition for division by an interval containing zero our only choice is to split the domain further and hope that with sufficiently small parts this will not occur. Note though that this can never give us a false answer, we simply do not get any answer at all until the partitioning is sufficiently small.

4.2 Exhaustive Random Search

To find the zeros we will use a combination of the normal Newton's method and the validated Newton's method as described in section 2.2.

To generate possible zeros we will take a set of random points, P_0 in the domain D to which we apply Newton's method for a fixed number of steps. This will give us a new set of points, P which are possible zeros to the function. Here we do not need to use interval arithmetic since the method itself is not rigorous, instead we use ordinary floating points.

For the points in P we first check if they are inside D. If they are, we check for zeros close to them using the interval valued Newton's method. To do this we create a small box around each point to which we apply the method. Since we are now dealing with boxes in $\mathbb{C}^2$ we will use the algorithm in equation 23. We iterate this a couple of times and if for some k, $N(\mathbf{z}_k) \subseteq \mathbf{z}_k$ we know that the box contains exactly one zero.

Much like with division by zero in the integration we might run into the problem that $0 \in \det J_F$. What we can do then is to try and enclose the original point in a smaller box and hope that this solves the problem.

5 Examples

We will here demonstrate the algorithm by using it to find zeros of different functions. The first examples will be primarily academic in nature but we will also consider some functions originating from *strong fields approximations* in physics were we will be able to generate some new results.

The implementation of the algorithm makes use the C-XSC library for interval computations [3], which has good support for complex intervals. To find interval enclosures to the partial derivatives of the function we make use of interval Taylor arithmetic. This allows us to find and enclosure to the Taylor polynomial of the function of any degree. An implementation of this was found in [2].

Since the boundary of the domain is divided into 8 cuboids, and we calculate them separately, we can easily parallelize this process. This allows us to make use of up to 8 cores at once. However for some of the integrals we take this one step further and divide each cuboid into several parts, adjusting the required tolerance thereafter. Doing this we can use any number of cores, and even distribute the computations on several CPUs.

5.1 Example 1

We begin by considering the function $f(z_1, z_2) = (z_1, z_2)$. Finding the unique zero $(0, 0)$ is of course trivial, but we will use this example to demonstrate the use of the integration. Even for this simple function we will see that evaluating the integral is sometimes hard.

The computations were done using an i7-4700 and 8 GB of RAM. We calculate the integral for each cuboid separately but do not parallelize them further.

We first start with computing the number of zeros in the domain

$$D = [-1, 1] + i[-1, 1] \times [-1, 1] + i[-1, 1], \tag{26}$$

i.e. the unit cube in $\mathbb{C}^2$. The first cuboid is given by $D_1 = -1+i[-1,1]\times[-1,1]+i[-1,1]$ and calculating the integral we get the enclosure $[0.080801, 0.187438] + i[-0.025834, 0.025834]]$ and the computations take 16 msec. The remaining 7 cuboids all give the exact same result and summing them up we get the enclosure $I \in [0.646407, 1.499505] + i[-0.206825, 0.206825]$ for the whole integral. Having access to 8 cores means they can all be done in parallel. Since we know that the real part of the integral should be an integer and the imaginary part zero we must have $I = 1$ and the domain contains exactly one zero, which we of course already knew.

Next we consider the domain

$$D = [-1, 1] + i[-5, 5] \times [-5, 5] + i[-5, 5], \tag{27}$$

so the only change is that some of the intervals are wider. This time the first cuboid is given by $D_1 = -1 + i[-5, 5] \times [-5, 5] + i[-5, 5]$ and calculating the integral we get the enclosure $[0.351591, 0.476418]+i[-0.190240, 0.190240]$. This time though the computations take 887567 msec, about 15 minutes. We get the same result for D_2, but for $D_3, ..., D_8$ we get the enclosure $[0.008770, 0.081057]+i[-0.081057, 0.081057]$ and it takes less then 1 msec. So while most of the integrals actually go faster, two of them takes much longer. Summing all these integrals up we get $[0.755801, 1.439179]+i[-0.737133, 0.737133]$ as an enclosure for the whole integral, and again we can conclude that the function has one zero in the domain.

To understand why some integrals where faster to compute and others slower we have to look at the domain D and its boundary compared to the zeros of f. The integrand involves division with $|f|$ and thus moving close to a zero of f will in general make the integrand very large. When widening some of the sides of D the boundary, and thus the domain of integration, moves further away from the zero of f. This will make the integrand smaller and thus compensate for the larger domain. When we integrate over the only part of the boundary still close to the zero we have both the problem of being close to the zero and suffer from the larger domain, giving a large increase in computational time.

From this we can conclude that even for trivial functions the integral can sometimes be hard to evaluate and it is sensitive to changes in the domain. An evaluation time of 15 minutes is acceptable but this still for a trivial function and a rather small domain.

5.2 Example 2

We now move over to a more complex example were the function is given by $f(z_1, z_2) = (\sin(z_1) + z_1^2 + e^{z_2} - \cos(2z_2), \cos(z_1) + z_2^3 + e^{2z_2} - 2)$ and the domain by $D = [-1, 1] + i[-1, 1] \times [-1, 1] + i[-1, 1]$. In [6] they consider the same function, but then with the domain given by the unit sphere in $\mathbb{C}^2$.

The computations were done using an Intel Xeon E5-2620 processor and 32 GB of RAM.

First we want to find the number of zeros by evaluating the integral. This time we split each cuboid into 32 parts, and evaluate the integral on each part separately. This allows us to make use of all 12 cores of the processor when evaluating one cuboid. By also using 8 identical computers we could also do all cuboids in parallel.

The time it took to reach the required tolerance varied very much between the different parts. About a third of them finished within a minute. Whereas some of them was running for 3.5 hours. After 3.5 hours we at least had the enclosure $[3.395191, 4.766817] + i[-0.583617, 0.583607]$ for the integral. Note that the width is larger than 1, this is because not all of the parts were finished. But the enclosure was small enough to only contain 1 integer and thus we can conclude that the exact result must be 4. Therefor the number of zeros in the domain is 4.

Next we want to do an exhaustive search for the zeros. We start by generating random points in D and perform 15 iterations of Newton's method on these, this continues until we have found 50 points that are still in D after the 15 iterations which takes about 0.1 seconds. Now we have 50 points that we want to check if they are zeros to the function.

For each of these points we create a small box, having a width of 10^{-10}, containing it. On these boxes we then perform Newton's interval method, this whole procedure takes less than 0.1 seconds. In the end this results in a total of 4 disjoint boxes proved to contain a zero of the function. Since we know that the function has no more than 4 zeros in the domain these must be all zeros.

5.3 A physical example

Finally we will consider an example from physics. The functions we will consider originates from the area in physics that deals with *Strong Field Approximations*. It describes how strong laser fields interact with atoms and molecules [8, 7].

The functions in question are

$$f_1(z_1, z_2) = (k_s(z_1, z_2) + A(z_1))^2 + 2I_p \tag{28}$$

$$f_2(z_1, z_2) = (p_z + A(z_2))^2 - (k_s(z_1, z_2) + A(z_2))^2 \tag{29}$$

where

$$A(z) = A_0 \cos(\omega_0 z + \phi) + A_1 \cos(\omega_1 z + \phi) + A_2 \cos(\omega_2 z + \phi) \tag{30}$$

$$k_s(z_1, z_2) = -\frac{\alpha(z_2) - \alpha(z_1)}{z_2 - z_1} \tag{31}$$

$$\alpha(z) = \frac{A_0}{\omega_0} \sin(\omega_0 z + \phi) + \frac{A_1}{\omega_1} \sin(\omega_1 z + \phi) + \frac{A_2}{\omega_2} \sin(\omega_2 z + \phi). \tag{32}$$

$$\tag{33}$$

They values used are $A_0 = E_0/\omega_0$, $A_1 = E_1/\omega_1$ and $A_2 = E_2/\omega_2$, $E_0 = F_0/2$, $E_1 = -E_0/2$, $E_2 = E_1$, $F_0 = 0.0534$, $\omega_0 = 0.057$, $\omega_1 = \omega_0 + \omega_p$, $\omega_2 = \omega_0 - \omega_p$, $\omega_p = \omega_0/n_p$, $n_p = 4$, $\phi = 0$, $I_p = 0.5145$. For p_z the value will in practice vary but in this example we will use $p_z = 0$.

The domain to search for zeros in will be $D = [0, 450] + i[0, 450] \times [0, 450] + i[0, 450]$

What can be noted is that the function has a continuum of removable singularities at $z_1 = z_2$.

5.3.1 Calculating the Integral

To find the total number of zeros in the domain we, as before, want to calculate the logarithmic integral around the boundary. However this turns out to be extremely costly and in the end we are unable to find a meaningful enclosure of the integral.

One of the first problems that occur is that the computer is unable to handle the removable singularity at $z_1 = z_2$. To avoid this problem the function would have to be rewritten in such a way that the singularity is removed. This could for example be done by either manually expanding it in a Taylor series around the area. However this was never done, because it turns out, see below, that even in the regions without a singularity we are unable to find a good enclosure.

We focused on calculating the integral on the first cuboid, given by $D_1 = [0]+i[0,450]\times[0,450]+i[0,450]$, with the required tolerance of 0.125. The cuboid were split into 1024 parts and we started by calculating the integral on the last part given by $[0] + i[421.875, 450] \times [393.75, 450] + i[393.75, 450]$, which would require a tolerance of $0.125/1024 = 0.0001220703125$. Note that this domain has no singularity. After running the computations on 12 cores for more than 24 hours the enclosure had a width of 1.18, far from the required tolerance.

Being unable to compute even a tiny part of the integral using 24 hours of computations of course means that calculating the whole integral is out question.

5.3.2 Random Search for Zeros

Even though we are unable to calculate the number of zeros in the domain we can still perform a search for them. We can find and isolate zeros and thus get a lower bound for the number of zeros, however we will never be able to show that we have found all zeros this way.

Since the maximum number of zeros is unknown what we will do is to generate more and more possible zeros until the number of zeros found stabilizes.

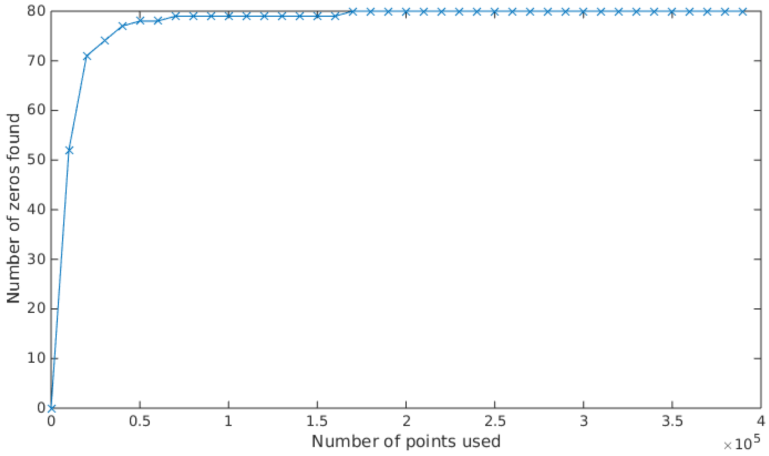
We will start by generating points in D and perform 15 Newton iterations on these. This continues until we have found 10000 points that stay inside D. These boxes are then check for zeros using the validated Newtons method. This takes 15 seconds and gives us a list of zeros to the function. We then do the same thing again, generating a number of possible zeros and validating them. The new zeros found are then added to the list of zeros. When the number of zeros found no longer increases the process will stop and we get a final list of zeros. In figure 2 the number of zeros found is plotted towards the number of points used. From this we can conclude that the function has at least 80 zeros in the domain. It is possible that we have found all zeros to the function, or at least the remaining zeros are hard to reach using Newton's method. One possibility is also that the function has non-simple zeros since the validated Newton's method cannot find those.

6 Conclusions

We have presented a method that is able to find and validate simple zeros to systems of two analytic functions on boxes. It can also find the number of zeros in the box which enables it to show that all zeros have been found. In practical examples however calculating the number of zeros is very time consuming and not always applicable, though we can still find and validate simple zeros.

The method requires only the function to be inputted and calculates the required derivatives automatically. It however requires some tuning for prob-

Figure 2: The number of zeros found,



lem. The parallelization of the integral is partially done manually and must be adapted to the hardware and difficulty of the integral. When searching for zeros the number of points to generate depends greatly on the function, and the number of iterates used it Newton's method can also be adapted. Finally an adequate size for the boxes used in the validated Newton's method must be chosen.

The time consumption when performing the random search for zeros depends greatly on the function, most notably on the number of zeros it has in the domain. It ranges from some seconds to several minutes. Since most part of the algorithm is trivially parallelizable the time used can be reduced by using more hardware.

Future research would aim at finding higher order methods for calculating the integral and implementing methods for localization of multiple zeros.

References

[1] L.A. Aizenberg and A.P. Yuzhakov. *Integral Representations and Residues in Multidimensional Complex Analysis*. Vol. 58. American Mathematical Society, 1983.

[2] Frithjof Blomquist, Werner Hofschuster, and Walter Krmer. "Real and Complex Taylor Arithmetic in C-XSC". In: (2005).

[3] *CXSC - C++ eXtension for Scientific Computation, version 2.5.4. Available from http://www2.math.uni-wuppertal.de/x̃sc/.*

[4] J. Hubbard, D. Schleicher, and S. Sutherland. "How to find all roots of complex polynomials by Newtons method". In: *Invent. math* 146.1 (2001), pp. 1–33.

[5] Thomas Johnson and Warwick Tucker. "Enclosing all zeros of an analytic function A rigorous approach". In: *Journal of Computational and Applied Mathematics* 228.1 (2009), pp. 418–423.

[6] Peter Kravanja. “On Computing Zeros of Analytic Functions and Related Problems in Structured Numerical Linear Algebra”. PhD thesis. Katholieke Universitiet Leuven, 1999.

[7] M. Yu. Ivanov Anne L’Huillier M. Lewenstein Ph. Balcou and P. B. Corkum. “Theory of high-harmonic generation by low-frequency laser fields”. In: *Phys. Rev. A* 49.3 (1994), pp. 2117–2132.

[8] D B Milosevic et al. “Above-threshold ionization by few-cycle pulses”. In: *J. Phys. B* 39 (2006), R203–R262.

[9] V. Pan. “Solving a polynomial equation: Some history and recent progress”. In: *SIAM Rev* 39.2 (1997), pp. 187–220.

[10] Warwick Tucker. *Validated Numerics*. Princeton University Press, 2011.