Phase Portraits of Bi-dimensional Zeta Values

Olivier Bouillot(✉)

Gustave Eiffel University, Marne-la-Vallée, France
olivier.bouillot@u-pem.fr
http://www-igm.univ-mlv.fr/∼bouillot/

Abstract. In this extended abstract, we present how to compute and visualize phase portraits of bi-dimensional Zeta Values. Such technology is useful to explore bi-dimensional Zeta Values and in long-term quest to discover a 2D-Riemann hypothesis.

To reach this goal, we need two preliminary steps:

- the notion of phase portraits and a general tool to visualize phase portrait based on interactive Jupyter widgets.
- the ability to compute numerical approximations of bi-dimensional Zeta values, using `mpmath`, a Python library for arbitrary-precision floating-point arithmetic. To this end, we develop a theory to numerically compute double sums and produce the first algorithm to compute bi-dimensional Zeta Values with complex parameters.

Keywords: Phase portrait · Lindelöf formula · double Zeta Values

1 Introduction

The Riemann Zeta function ζ and the bi-dimensional Zeta Values, also called 2D-Zeta Value, bizetas or double Zeta Values, are respectively defined by:

$$\zeta(s) = \sum_{k>0} \frac{1}{k^s}, \text{ for all } s \in \mathbb{C}\ ,\ \Re e\ s > 1. \tag{1}$$

$$\mathcal{Z}e^{s_1,s_2} = \sum_{k>l>0} \frac{1}{k^{s_1} l^{s_2}}, \text{ for all } s_1, s_2 \in \mathbb{C}\ ,\ \begin{cases} \Re e\ s_1 > 1. \\ \Re e\ (s_1 + s_2) > 2. \end{cases} \tag{2}$$

It is well-known that these functions can be respectively meromorphically extended to $\mathbb{C}$ and $\mathbb{C}^2$ (see [9]). It is also conjectured that the zeros of the Riemann Zeta function be complex numbers with real part $\frac{1}{2}$: this is the Riemann hypothesis, stated by Riemann himself in 1859 in [10]. A nice long term quest is to discover the location of zeros of bi-dimensional Zeta Values.

This quest needs first a tool to visualize a representation of a function f defined over (a part of) $\mathbb{C}$ and valued in $\mathbb{C}$, such that looking for zeros of f becomes easy.

A. M. Bigatti et al. (Eds.): ICMS 2020, LNCS 12097, pp. 393–405, 2020.
https://doi.org/10.1007/978-3-030-52200-1_39

Constructing a graphical representation of a function $f : \mathbb{R} \longrightarrow \mathbb{R}$ is quite easy, we just need a 2-dimensional space. However, constructing the graph of a function $f : \mathbb{C} \longrightarrow \mathbb{C}$ is not so simple because we need a 4-dimensional space. In particular, this implies that most of us have difficulties visualizing such a graph. The first objective of this note is to review how to read and construct phase portraits, which are heat maps where colors encode simultaneously phase and modulus (see Sect. 2).

Let us notice that there already exists a tool, developed by Elias Wegert in Matlab (see [12]), to represent phase portraits of complex functions. In particular, it has been used to explore many functions and illustrate a first course on complex analysis (see [13]). Unfortunately, this tool computes a lot of values to produce the desired representation and does not save them. So, research the localization of the zeros of a function with it is necessarily time-consuming. In other words, it is not an efficient tool for our purpose.

Nevertheless, Jupyter widgets (see [14]) are wonderful tools that easily enable us to construct a general phase portrait visualization tool, where the input is just a complex function. The author has implemented such a widget (see [1]) where, with interactivity, the user can change the visualization window, the number of pixels used in a unit square, reuse already computed values or store new ones in a database, save pictures or see information on how the computations progress.

This widget can be extended to visualize phase portraits of functions with two complex variables. Visualizing such a function is nothing else than drawing a representation of the partial functions and move inside them. According to this, the author has implemented a second interactive Jupyter widget to realize this walk.

To benefit from the power of the second widget, we need to have an efficient algorithm to numerically compute bi-dimensional Zeta Value. This is the second preliminary step to look for a 2D-Riemann hypothesis.

Nowadays, researchers can compute multiple zeta values (and in particular bizetas), with integer parameters, with as many digits as we wanted (see [3]). But, to the best of the author's knowledge, nothing is known to compute these numbers with complex parameters in any length.

Therefore, the last objective of this note is to present how a double sum can be numerically computed, and apply the developed method to bizetas. This method is based on a generalization of a poorly known Lindelöf formula explaining how to compute the sum of the values at integers of a class of holomorphic functions. Section 3 contains a presentation of Lindelöf formula, as well as a comparison between three methods to compute evaluations of the Riemann Zeta function.

Lindelöf formula generalization to double sum, and therefore to bizetas, is discussed in Sect. 4. One could notice that Lindelöf formula can be written as an integral. Therefore, we generalize the process to compute double sums using double integrals and, then, expansions of these integrals. Of course, this technic is still valid for multiple sums and gives, in theory, an algorithm to compute multiple Zeta Value in any length, with complex parameters!

Finally, once the Jupyter visualization widget of a two complex variables function is available and the computational machinery is developed, we can explore phase portraits of convergent 2D-Zeta Values, or linear combinations of these numbers, to find out some zeros. Therefore, the last necessary point is to be able to numerically compute the analytic extension of bizetas. Only then, using our Jupyter widget, a Riemann hypothesis for bizetas could be conjectured.

2 Drawing Function from $\mathbb{C}$ to $\mathbb{C}$: Phase Portraits

In this section, we review how to construct and read phase portraits. We will see how to find zeros and poles of a function, but also points (called *color saddle*) where the derivative becomes null. References for this section are [11] and [13].

2.1 Analytic Landscapes vs Phase Portraits

Drawing a function $\mathbb{R}^2 \longmapsto \mathbb{R}$ is typically achieved through 2D heat maps (see [15]) or 3D plots. For a function $f : \mathbb{C} \longmapsto \mathbb{C}$, these approaches may be used to visualize the modulus; that is *analytic landscapes*. We could also use such plots to visualize the phase of f.

However, many applications like ours require to visualize the modulus and the phase at the same time to visualize poles and zeroes. In addition, such visualization should highlight simultaneously very small modulus (near zeroes) and huge modulus (near poles).

Here come into play *phase portraits*: heat maps where the color of a pixel z encodes simultaneously the phase of $f(z)$ (by giving color to its values) and the (logarithm of the) modulus (by using different brightness). Eventually, the modulus can then be emphasized by adding contours.

2.2 HSL-representation of Colors

The HSL-representation of colors is an alternative representation of the RGB color model and can be seen as a cylindrical geometry (see Fig. 1). Its angular dimension, the hue, starts at the red color at 0°, passing through the green color at 120° and the blue one at 240° to finally come back to red at 360°. The lightness, *i.e.* the central vertical axis, describes the gray colors, ranging from black at the bottom of the cylinder with lightness or value 0 to white at the top of the cylinder with lightness or value 1.

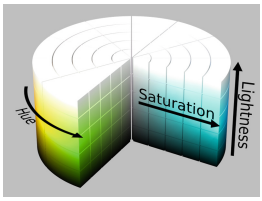


Fig. 1. The HSL cylinder Source: Wikipedia, "HSL and HSV" (Color figure online)

2.3 Principle of Phase Portraits

A pixel of coordinates (x, y) is associated with a complex number z. Then, we find a color c related to the value $f(z) = \rho e^{i\theta}$ using the HLS representation of colors. Therefore, the pixel (x, y) is colored by the color c. (See Fig. 2).

Let us mention that zeros and poles play an important role when visualizing a phase portrait. Using the logarithm of the modulus, instead of the modulus itself, allows us to have a smoother graphic representation and also to have symmetrical behavior between zeros and poles, as well as symmetric visualization of small and big values.

Moreover, to encounter all the possible values of the modulus, we compactify the extended real number line $[-\infty; +\infty]$ to $[-1; 1]$, using $x \longmapsto \dfrac{x}{1+|x|}$.

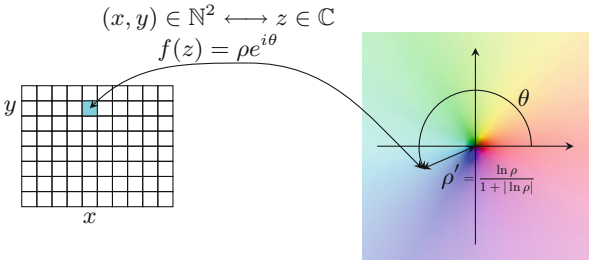


Fig. 2. Color coding of pixels to draw phase portraits of a complex function f

Consequently, a pixel of coordinates (x, y), via its associated complex number z, is associated with the color obtained in the HSL model by:

$$H = \text{phase}\big(f(z)\big) \qquad L = \frac{1}{2}\left(\frac{\ln|f(z)|}{1+\big|\ln|f(z)|\big|} + 1\right) \qquad S = 1. \tag{3}$$

In particular, positive real numbers appear reddish in a phase portrait when negative real numbers appear cyan. Moreover, a zero is a black point while a pole is a white point.

Let us notice that the color map can be, in principle, adapted to colorblind people provided that the inherent periodicity of color be fulfilled.

2.4 Easy Properties to Read on Phase Portraits

The phase portraits not only show the location of zeros and poles of a function but also reveal their multiplicity. As an example, Fig. 3a shows a zero with multiplicity three. It is easily recognizable: z^3 travels 3 times around 0 when z moves once around 0 along a small circle. We emphasize that the colors met are in the reverse order for zeros and poles (compare the pole 0 to the three zeros located in cube root of -2 in Fig. 3b).

Moreover, it can be shown that the points where $f'(z) = 0$, with $f(z) \neq 0$, are where the isochromatic lines meet. Such points are called *color saddles*. See Fig. 3b where the color saddles are located at cube roots of -2. See also Fig. 3c.

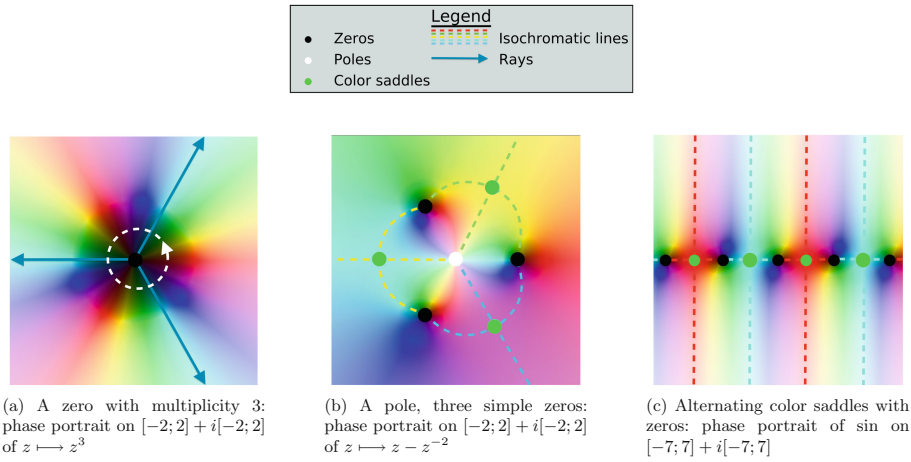
2.5 Necessary Precision to Choose Exact Color

In most cases, only a few significant digits (typically five), need to be known to plot phase portraits of a function $f : \mathbb{C} \longrightarrow \mathbb{C}$. In particular, we do not assume that the underlying system can do arbitrary precision computations.

We review here how to assess the required precision and how to achieve that precision.

If ε_L and ε_H denote the absolute error of L and H, the RGB components of the associated color are fixed as soon as we have: $\dfrac{\varepsilon_H}{60} + 6\varepsilon_L \leq \dfrac{1}{255}$. More precisely, we seek to have: $\varepsilon_L \leq 2040^{-1}$ and $\varepsilon_H \leq 20^{-1}$.

If we assume that Re $f(z)$ and Im $f(z)$ are known up to $\varepsilon > 0$, then, $|f(z)|$ is known up to 2ε. If $|f(z)| \geq 1$, then we can prove that $\varepsilon_L \leq 2\varepsilon$. If $|f(z)| < 1$, then this time $\varepsilon_L \leq \dfrac{k+2}{\ln^2(k+2)}\varepsilon$, where $k \in \mathbb{N}^*$ is such that $|f(z)| \in \left[\dfrac{1}{k+1}; \dfrac{1}{k}\right[$.



(a) A zero with multiplicity 3: phase portrait on $[-2;2]+i[-2;2]$ of $z \longmapsto z^3$

(b) A pole, three simple zeros: phase portrait on $[-2;2]+i[-2;2]$ of $z \longmapsto z - z^{-2}$

(c) Alternating color saddles with zeros: phase portrait of sin on $[-7;7]+i[-7;7]$

Fig. 3. Examples of easy properties seen on phase portraits

Consequently, if $f(z)$ is computed up to $\varepsilon = 10^{-4}$ and satisfied $|f(z)| \geq 0.01$, we have $\varepsilon_L \leq 2040^{-1}$ for all cases and the parameter L is well-approximated.

On the H-side, if we assume that $R = \text{Re } f(z)$ and $I = \text{Im } f(z)$ are known up to $\varepsilon > 0$ and satisfied $|R|, |I| \geq \alpha$ for a positive real number α, then, we can show that $\varepsilon_H \leq \varepsilon\left(\alpha^{-1} + (\alpha - \varepsilon)^{-1}\right)$. In particular, for $\alpha = 5 \cdot 10^{-3}$ and $\varepsilon = 10^{-4}$, we thus have $\varepsilon_H \leq 20^{-1}$ and the parameter H is well-approximated.

Consequently, in most cases, we compute the function $f : \mathbb{C} \longrightarrow \mathbb{C}$ up to 10^{-4}. However, in the other cases, *i.e.* when $|f(z)| < 0.01$, or $|\text{Re } f(z)| < 5 \cdot 10^{-3}$ or $|\text{Im } f(z)| < 5 \cdot 10^{-3}$, then we will compute again $f(z)$ up to a smaller precision, *e.g.* $\varepsilon = 10^{-5}$ or 10^{-6}, in order to finaly have $\varepsilon_L \leq 2040^{-1}$ and $\varepsilon_H \leq 20^{-1}$.

2.6 Interactive Visualization of Phase Portraits for Jupyter Notebook

The next step is to create a tool to visualize phase portraits of a given complex function $\mathbb{C} \longrightarrow \mathbb{C}$.

Let us remind such a tool was already available in Matlab (see [12]). However, for our long-term quest of a 2D Riemann hypothesis, we need a tool where the visualization window of our phase portraits can be easily modified.

According to the official website, a "Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text" (see also [7] about Jupyter). Moreover, interactive widgets like Sliders, Buttons or Images are now available in Jupyter (see [14]).

Consequently, the author has implemented an interactive Jupyter widget for general visualization of phase portraits (see [1]). Figure 4 is a graphic representation of its usage. For examples of outputs, let us note that all the phase portraits shown in this paper have been constructed from this widget.

The code of this interactive widget is already freely available in the following GitHub repository: https://github.com/tolliob/PhasePortrait. In particular, the interested reader will find in this repository an online usable Jupyter notebook. The widget can also be installed using the `pip` Unix command.

The input of this widget is a complex function $f : \mathbb{C} \longrightarrow \mathbb{C}$. Interactivity allows the user to fix a visualization window by setting the left below and right upper corners. Once these corners fixed, the user can:
$\rightsquigarrow$ launch the needed evaluations of the function f, or retrieve them;
$\rightsquigarrow$ show the phase portrait of f in the desired window;
$\rightsquigarrow$ save the produced image
Then, the user can change the visualization window by going back to these steps.

Another interactive Jupyter widget has also been implemented by the author. It aims to visualize phase portraits of two variables complex functions from its partial functions phase portraits. The input is a function $f : \mathbb{C}^2 \longrightarrow \mathbb{C}$, outputs are again phase portraits, alternatively of $f(s_0, \cdot)$ and $f(\cdot, s_0)$. Interactivity allows the user to move a partial function to another one by moving the point s_0 to a close complex point, or to switch from a partial function to the other one.

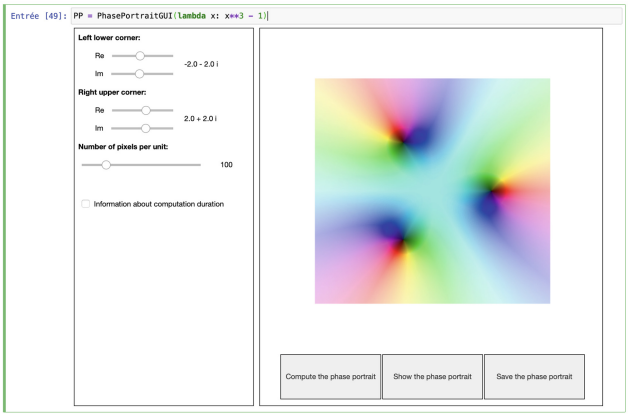


Fig. 4. Interactive Jupyter widget to construct phase portraits

3 Computation of the 1D-Zeta Value

In this section, we present a poorly known formula due to Lindelöf (see [8], chapter 3) which explains how to compute the sum of the values at integers of holomorphic functions. Then, we apply it to the Riemann Zeta Function and compare its performances to the Euler-Mac Laurin summation.

However, we warn the reader that Lindelöf formula application to the Zeta function could not compete computation technics dedicated to the Zeta function, such that the Riemann-Siegel formula or the Odlyzko-Schönhage algorithm (see [5] for example). However, Eqs. (7) and (9) will prove that Lindelöf formula can be extended to the numerical computation of double sums, and so to the numerical computation of 2D-Zeta Values, while it seems nowadays impossible to adapt specific ζ computational technics to 2D-Zeta Values.

3.1 On Lindelöf Formula

Before giving the Lindelöf formula, we first emphasize a technical definition from Chapter 3, p. 54, of [8].

Definition 1. *Let $m_0 > \frac{1}{2}$ be a real number. Let also Ω_s be the set defined for all real number s by $\Omega_s = \{z \in \mathbb{C}\ ,\ Re\ z \geq s\}$.*
A holomorphic function $f : \Omega_{m_0-\frac{1}{2}} \longrightarrow \mathbb{C}$ satisfies the 1D-Lindelöf hypothesis if:

1. *for all $s \geq m_0 - \frac{1}{2}$, $\lim\limits_{t \longrightarrow \pm\infty} e^{-2\pi|t|} f(\tau + it) = 0$ uniformly for $\tau \in \left[m_0 - \frac{1}{2}; s\right]$;*
2. *for all $s \geq m_0 - \frac{1}{2}$, $t \longmapsto e^{-2\pi|t|} f(s + it) \in \mathcal{L}^1(\mathbb{R})$ and*

$$\lim_{s \longrightarrow +\infty} \int_{-\infty}^{+\infty} e^{-2\pi|t|} |f(s + it)\ dt = 0. \tag{4}$$

Now, exploring Chapter 3 of [8], one can reconstruct the following theorem which gives an integral representation of sums of values of holomorphic functions. The proof is based on a clever use of the residuum principle. Nevertheless, the result is quite unknown.

Theorem 1. *(Lindelöf, 1905)*
Let $m_0 > \frac{1}{2}$ be a real number, m a positive integer such that $m \geq m_0$ and $f : \Omega_{m_0-\frac{1}{2}} \longrightarrow \mathbb{C}$ be an holomorphic function over $\Omega_{m_0-\frac{1}{2}}$ satisfying

- *the 1D-Lindelöf hypothesis ;*
- *$\sum_{\nu \geq m} f(\nu)$ is a convergent series.*

Then, $f \in \mathcal{L}^1([m_0 - \frac{1}{2}; +\infty[)$ and

$$\sum_{\nu \geq m} f(\nu) = \int_{m-\frac{1}{2}}^{+\infty} f(t)\ dt - i \int_0^{+\infty} \frac{f(m - \frac{1}{2} + it) - f(m - \frac{1}{2} - it)}{e^{2\pi t} + 1}\ dt. \tag{5}$$

The Lindelöf formula (5) can be used to numerically compute sums, expanding its second integral using Taylor expansion with integral remainder: this gives us a Lindelöf's Euler-Maclaurin like formula.

Theorem 2. *(Lindelöf, 1905)*
Let m_0, m and f be like in Theorem 1. Then:

$$\forall K \in \mathbb{N}, \sum_{\nu \geq m} f(\nu) \approx \int_{m-\frac{1}{2}}^{+\infty} f(t)\ dt + \sum_{k=1}^{K} \left(1 - \frac{1}{2^{2k-1}}\right) \frac{B_{2k}}{(2k)!} f^{(2k-1)}\left(m - \frac{1}{2}\right). \tag{6}$$

Coefficients in (6) are nothing else than the Taylor coefficients of $z \longmapsto \dfrac{x}{2\sinh\left(\frac{z}{2}\right)}$. So, in a certain sens, Eq. (6) means that:

$$\sum_{\nu \geq m} f(\nu) \approx \int_{m-\frac{1}{2}}^{+\infty} \frac{\frac{d}{dz}}{2\sinh\left(\frac{1}{2}\frac{d}{dz}\right)}(f)(t)\ dt. \tag{7}$$

While in Lindelöf's days, it was not necessary to have a precise estimate of the approximation error, we now need it due to the existence of computers and their immense associated computing power.

Theorem 3. *Let m_0, m, K and f be like in Theorem 1. Let also assume that, for all $u \geq m_0 - \frac{1}{2}$, the quantity $M_u(f) = \sup_{\substack{\zeta \in \mathbb{C} \\ Re\ \zeta \geq u}} |f(\zeta)|$ is well-defined.*

If we denote by $\mathcal{R}_{K,m}(f)$ the Lindelöf remainder in Eq. (6), i.e. the difference between the right hand side and the left hand side of Eq. (6), then we have the following upper bound:

$$|\mathcal{R}_{K,m}(f)| \leq \frac{M_{m_0-\frac{1}{2}}(f)}{(m - m_0)^{2K+1}} \cdot \frac{(2K+1)!}{(2\pi)^{2K+1}}. \tag{8}$$

3.2 Application to the Riemann Zeta Function

To produce a phase portrait of the Riemann Zeta function ζ, we will compute the values $\zeta(s)$ for s in a grid up to 10^{-4}.

First, the Euler-Maclaurin summation process applies (see [2]).

Moreover, the functions $f_s : z \longmapsto e^{i\pi z} z^{-s}$ satisfy the $1D$-Lindelöf hypothesis for all complex numbers s such that $\Re es > 0$. So, Eq. (5) can be used, its second integral being numerically computed with the `mpmath` Python library.

Equation (6) can also be used. We just have to find the best choice of triplet (m_0, m, K). Note that the bigger K is, the longest the computation, due to the computation of the successive derivative which concentrates most of the multiplications. However, the size of m also intervenes.

This allows us to compute $\eta(s) = \sum_{n>0} (-1)^n n^{-s}$ and $\zeta(s)$, for all $s \in \Omega_{\frac{1}{2}}$. Now, classical tools can be used to compute ζ on the whole punctured complex plane $\mathbb{C} - \{1\}$ (reflection formula, analytic continuation process; see [4]). Using the Jupyter widget [1], this generates the phase portraits available in Figs. 5a–5c.

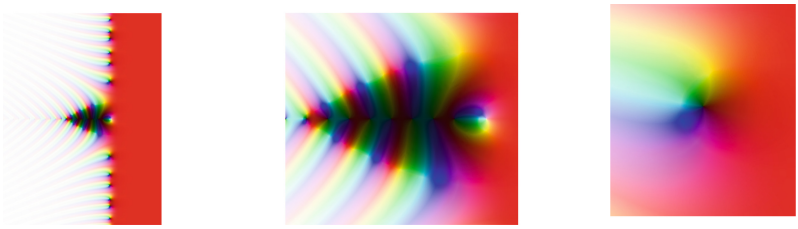
3.3 Comparaison of the Three Methods

Figures 6a to 6d show some comparisons between the three previous methods (*i.e.* the Euler-Maclaurin summation process, the Lindelöf formula (5) and the Lindelöf Euler-Maclaurin like formula given by Eq. (6)).

First, we can see that the integral Lindelöf method is the most stable method, even if there are some discontinuities (see Fig. 6c and 6d). This is explained by the different numerical integration methods used by the `mpmath` Python library.

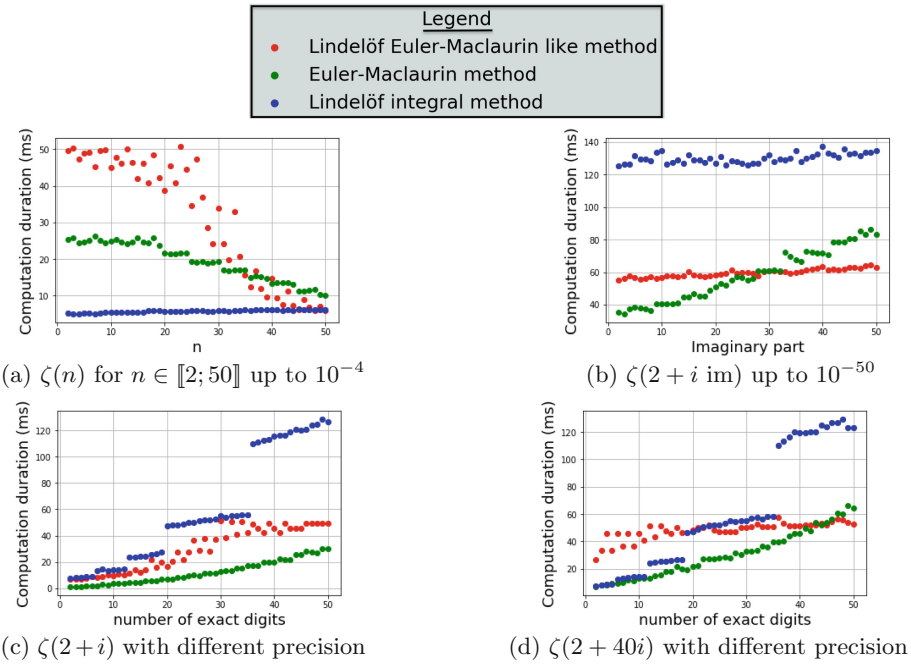
According to these Figures, we see that to have numerous exact digits, the bigger $|s|$ is, the more efficient the Lindelöf Euler-Maclaurin-like method is.

Finally, for the three methods, we observe that the computation gets longer as we want more exact digits. This increase is correlated to the modulus of $|s|$. It turns out that Lindelöf Euler-Maclaurin-like formula is the method with the slower computation time increase relatively to the required number of digits in the computation of $\zeta(s)$.



(a) On $[-40; 20] + i[-40; 40]$ (b) On $[-18; 14] + i[-10; 10]$ (c) Near the first non trivial zero $z_1 \approx 0.5 + 14.135$

Fig. 5. Three phase portraits of the Rieman Zeta function ζ



(a) $\zeta(n)$ for $n \in [\![2; 50]\!]$ up to 10^{-4}

(b) $\zeta(2 + i \text{ im})$ up to 10^{-50}

(c) $\zeta(2+i)$ with different precision

(d) $\zeta(2 + 40i)$ with different precision

Fig. 6. Comparaison between three methods to compute Zeta Values

4 Computation of Convergent Two Dimentional Multiple Zeta Values by a 2D-Lindelöf Formula

To the best of the author's knowledge, no one was able to numerically compute a convergent Multiple Zeta Values (MZV) with complex exponents so far. The only known algorithm was dedicated to MZV with integers parameters: it uses binomial expansions where exponents are related to the MZV parameters (see [3]).

To produce and explore phase portraits of 2D-Zeta Values, we nevertheless need, first, to *be able to compute* these numbers, then to compute them *efficently* because producing phase portraits of 2D-Zeta Values could easily require millions of different evaluations. This difficulty is, of course, the cornerstone of our quest for a 2D-Riemann hypothesis.

Hopefully, one of the most important advantages of Lindelöf formulas (5) and (6) over Euler-Maclaurin formula is that it could be extended to higher dimensions. Consequently, in this section, we will present the first general method to compute double sums and then apply it to produce the first bi-dimensional Zeta Values with complex exponents computing algorithm.

Efficientness will not be discussed here for technical reasons, as well as the upper bound of the error term in the computation of a double sum.

4.1 On the 2*D*-Lindelöf Formula

First, we extend the 1D-Lindelöf hypothesis to the bi-dimensional case:

Definition 2. *We say that a function* $f : \Omega_{m_{01}} \times \Omega_{m_{02}} \longrightarrow \mathbb{C}$ *satisfies the 2D-Lindelöf hypothesis when:*

1. $\forall z_1 \in \Omega_{m_{01}}, f(z_1, \cdot)$ *satisfies the 1D-Lindelöf hypothesis over* $\Omega_{m_{02}}$*;*
2. $S = \sum_{l \geq m_2} f(\cdot, l)$ *is a well-defined function over* $\Omega_{m_{01}}$ *satisfying the 1D-Lindelöf hypothesis;*

Then, using an iteration of (7), we can prove the following:

Theorem 4. *Let* m_{01} *and* m_{02} *be two real number greater than* $\frac{1}{2}$, m_1 *and* m_2 *be two positive integers such that* $m_i \geq m_{0i}$, $i \in \{1; 2\}$ *and* $f : \Omega_{m_{01}} \times \Omega_{m_{02}} \longrightarrow \mathbb{C}$ *a fonction satisfying the 2D-Lindelöf hypothesis.*

Let us also suppose that $\sum_{\substack{k \geq m_1 \\ l \geq m_2}} f(k,l)$ *is a convergent double series. Then:*

$$\bullet \sum_{\substack{k \geq m_1 \\ l \geq m_2}} f(k,l) \approx \int_{m_1 - \frac{1}{2}}^{+\infty} \int_{m_2 - \frac{1}{2}}^{+\infty} \frac{\frac{d}{dz_1}}{2\sinh\left(\frac{1}{2}\frac{d}{dz_1}\right)} \frac{\frac{d}{dz_2}}{2\sinh\left(\frac{1}{2}\frac{d}{dz_2}\right)} (f)(u,v)\, dudv \tag{9}$$

$$\bullet \sum_{\substack{k \geq m_1 \\ l \geq m_2}} f(k,l) \approx \sum_{\substack{p,q \in \mathbb{N} \\ p+q \leq K}} \left(1 - \frac{1}{2^{2p-1}}\right)\left(1 - \frac{1}{2^{2q-1}}\right) \frac{B_{2p}}{(2p)!} \frac{B_{2q}}{(2q)!} \times \left(\int_{m_1 - \frac{1}{2}}^{+\infty} \int_{m_2 - \frac{1}{2}}^{+\infty} \frac{\partial^{2p+2q} f}{\partial_{z_1}^{2p} \partial_{z_2}^{2q}}(u,v)\, du\, dv \right) \tag{10}$$

for all integers K.

Some of the integrals in Eq. (10) could be explicitely computed; some could not. So, to obtain numerical approximation, we use the `mpmath` Python library for arbitrary-precision floating-point arithmetic (see [6]).

4.2 Computation of a Double Sum over $\mathbb{N}^2$

Now, we are able to compute a double sum over $\mathbb{N}^2$ by cutting $\mathbb{N}^2$ onto six parts (see Fig. 7):

$$\sum_{k \geq 0} \sum_{l \geq 0} f(k,l) = \sum_{k=0}^{m_1 - 1} \sum_{l=0}^{m_2 - 1} f(k,l) + \sum_{k=0}^{m_1 - 1} \sum_{l=m_2}^{q-1} f(k,l) + \sum_{k=0}^{m_1 - 1} \sum_{l \geq q} f(k,l) + \sum_{k=m_1}^{p-1} \sum_{l=0}^{m_2 - 1} f(k,l) + \sum_{k \geq p} \sum_{l=0}^{m_2 - 1} f(k,l) + \sum_{k \geq m_1} \sum_{l \geq m_2} f(k,l). \tag{11}$$

For "good" functions f, we can compute $\sum_{k=0}^{m_1-1}\sum_{l\geq q} f(k,l)$ and $\sum_{k\geq p}\sum_{l=0}^{m_2-1} f(k,l)$ using the 1D-Lindelöf Euler-Maclaurin-like formula, while $\sum_{k\geq m_1}\sum_{l\geq m_2} f(k,l)$ is computed using the 2D-Lindelöf Euler-Maclaurin-like formula.

Let us consider the function $f_{s_1,s_2} : (z_1, z_2) \longmapsto (z_1+1)^{-s_2}(z_1+z_2+2)^{-s_1}$. f_{s_1,s_2} satisfies the $2D$ Lindelöf hypothesis, while its partial functions satisfy the $1D$-Lindelöf hypothesis. Therefore, we can use the decomposition (11) to compute the double Zeta Value (see Fig. 8a and 8b).

According to Fig. 8a, we could conjecture that there is a pole near $(s_1, s_2) = (1, 0)$.

According to [9], we know exactly the localization of the poles of double Zeta values and $(1, 0)$ is actualy a pole. Up to long computations, we now are able to find out zeros of convergent double Zeta Values.

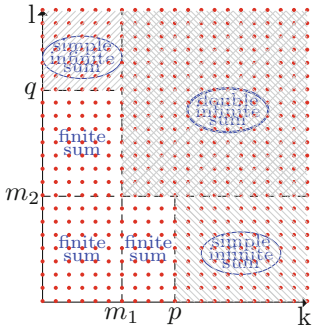
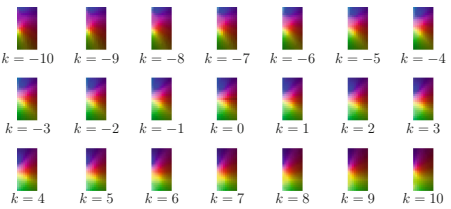
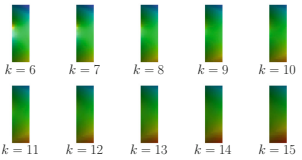


Fig. 7. Decomposition of $\mathbb{N}^2$ onto 6 parts to compute a double sum



(a) $s \longmapsto \mathcal{Z}e^{s,\frac{6+ik}{5}}$, on $\left[\frac{6}{5};3\right] + i[-2;2]$



(b) $s \longmapsto \mathcal{Z}e^{\frac{k}{5}-2i,s}$, on $\left[\frac{6}{5};3\right] + i[-2;5]$

Fig. 8. Expansion of phase portrait of partial functions of double Zeta values

Acknowledgment. The author would like to warmly thank the reviewers for their time and effort devoted to improving the quality of this work.

References

1. Bouillot, O.: Phaseportrait, a github repository (2020). https://github.com/tolliob/PhasePortrait
2. Cohen, H., Olivier, M.: Calcul des valeurs de la fonction zêta de riemann en multiprécision. C.R. Acad. Sci. Paris Sér I Math. **314**, 427–430 (1992)
3. Crandall, R.E.: Fast evaluation of multiple zeta sums. Math. Comp. **67**(223), 1163–1172 (1998)

4. Edwards, H.E.: Riemann's Zeta Function. Academic Press, New York (1974)
5. Gourdon, X., Sebah, P.: Numerical evaluation of the riemann ζ-function. http://numbers.computation.free.fr/Constants/Miscellaneous/zetaevaluations.pdf
6. Johansson, F., et al.: mpmath: a Python library for arbitrary-precision floating-point arithmetic (version 0.18), December 2013. http://mpmath.org/
7. Kluyver, T., et al.: Jupyter notebooks - a publishing format for reproducible computational workflows. In: Loizides, F., Schmidt, B. (eds.) Positioning and Power in Academic Publishing: Players, Agents and Agendas, pp. 87–90. IOS Press, Amsterdam (2016)
8. Le Lindelöf, E.L.: calcul des résidus et ses applications à la théorie des fonctions. Gauthier-Villars, Paris (1905)
9. Mehta, J., Saha, B., Viswanadham, G.K.: Analytic properties of multiple zeta functions and certain weighted variants, an elementary approach. J. Number Theor. **168**, 487–508 (2016)
10. Riemann, B.: Über die anzahl der primzahlen unter einer gegebenen grösse. Mon. Not. Berlin Akad. **2**, 671–680 (1859)
11. Semmler, G., Wegert, E.: Phase plots of complex functions: a journey in illustration. Notices Am. Math. Soc. **58**, 768–780 (2011)
12. Wegert, E.: Complex function explorer. https://www.mathworks.com/matlabcentral/fileexchange/45464-complex-function-explorer. Accessed 08 May 2020. MATLAB Central File Exchange
13. Wegert, E.: Visual Complex Functions. An Introduction with Phase Portraits. Springer, Basel (2012). https://doi.org/10.1007/978-3-0348-0180-5
14. Jupyter widgets community: ipywidgets, a github repository (2015). https://github.com/jupyter-widgets/ipywidgets
15. Wilkinson, L., Friendly, M.: The history of the cluster heat map. Am. Stat. **63**, 179–184 (2009)