University of Colorado Boulder

APPM 4350: Fourier Series and Boundary Value Problems

Project Report

Infinite Products and Mittag-Leffler Expansions

Authors
First LAST - UGRD APPM/ASEN[1]
first LAST - UGRD APPM/CSCI[2]
first LAST - UGRD APPM/ECON[3]

[1]SID: 1234
[2]SID: 1234
[3]SID: 1234

Keywords

Entire functions, Meromorphic, Poles, Infinite products, Uniform Convergence, Laurent-expansion, Mittag-Leffler Theorem (simple-poles and general case), Mitagg-Leffler Expansion

Abstract

In previous sections of the course we considered two infinite series representations of analytic functions in a suitable domain: Taylor Series and Laurent Series. These past representations are infinite sums, however, in addition to infinite sums, there is a vast world of potential with exploring representing functions as infinite products. In this report, we will develop the mathematical theory and methods to discuss the convergence of infinite products. Moreover, we will supplement our discussion of infinite products with examples found in the text. We also will touch on Mittag-Leffler expansions. This theorem concerns the existence of meromorphic functions with prescribed poles. Specifically, we can use it to express any meromorphic function as a sum of partial fractions. Using these infinite series representations, we will compare them to the performance of Taylor series, and conduct judgments on practicality of Mittag-Leffler and infinite product series in graphing calculators. We found that these series vastly outperformed Taylor series, and would be very useful in graphing calculators.

1 Introduction

1.1 Entire Functions

In order to understand what a Mittag-Leffler expansions are, one must understand what an entire function is. An **entire** function is one that is analytic at all finite points of the complex plane. Typical examples of functions that are entire are polynomials and the exponential function, and any finite sums, products and compositions of these, such as the trigonometric functions and their hyperbolic counterparts. Entire functions serve as the basis for meromorphic functions, which will be discussed in the next section.

1.2 Meromorphic Functions

A **meromorphic** function is a ratio of entire functions, for example, a rational function which is a ratio of two polynomials. In other words, it is a function that only has a finite number of poles in the finite z-plane. So, a meromorphic function may only have finite-order, isolated poles and zeros and no essential singularties in its domain. We begin with a discussion of entire and meromorphic functions since meromorphic functions are a necessary condition for the the Mitagg-Leffler theorems and expansions which we cover in this report.

1.3 Infinite Products

Our main resource for these next few sections was the textbook.

To understand the Weierstrass factorization of sine, we first need to understand what an infinite product is; for a sequence $\{a_n\}$ of complex numbers, the infinite product of that complex sequence would be:

$$P = \prod_{n=1}^{\infty}(1 + a_n) \tag{1}$$

An infinite product converges if the partial products sequence ($P_n = \prod_{n=1}^{k}(1 + a_n)$) converge, and for a N_0 large enough $\lim_{N\to\infty} \prod_{n=N_0}^{N}(1 + a_n)$ does not equal zero. Otherwise, the infinite product diverges.

1.4 Mittag-Leffler Theorem

We start with the Laurent expansion of a meromorphic function near a pole (of order N_j at $z = z_j$):

$$f(z) = \sum_{n=1}^{N_j} \frac{a_{n,j}}{(z - z_j)^n} + \sum_{n=0}^{\infty} b_{n,j}(z - z_j)^n \tag{2}$$

where the first sum p_j is the principal part at $z = z_j$ and contains the pole contribution:

$$p_j(z) = \sum_{n=1}^{N_j} \frac{a_{n,j}}{(z - z_j)^n} \tag{3}$$

and when the number of poles is finite then $f(z) = \sum_{j=1}^{m} p_j(z)$. However, with Mittag-Leffler expansions we want to represent the principal part of a function with an infinite number of poles, so we want to approximate $f(z)$ with principal parts: $\{p_j\}_{j=0}^{\infty}$ We want:

$$f(z) = p_0(z) + \sum_{j=1}^{\infty}(p_j(z) - g_j(z)) + h(z) = \tilde{f}(z) + h(z) \tag{4}$$

where $h(z)$ is an entire function, $\tilde{f}(z)$ and $f(z)$ have the same principle parts, and the sequence of polynomials $\{g_j(z)\}_{j=0}^{\infty}$ have $g_0(z) = 0$. We want to find the sequence of polynomials. For simple poles:

$$p_j(z) = \frac{a_j}{z - z_j} \tag{5}$$

and:

$$g_j(z) = -\frac{a_j}{z_j}(1 + (\frac{z}{z_j}) + ... + (\frac{z}{z_j})^{m-1}) \tag{6}$$

where m is an integer greater than or equal to 1 and $|\frac{z}{z_j}|$ is less than 1. We can then say that:

$$f(z) = p_0(z) + \sum_{j=1}^{\infty}(\frac{a_j}{z_j})L(\frac{z}{z_j}, m) + h(z) \tag{7}$$

Where $L(w,m)$ is:

$$L(w,m) = \frac{1}{w-1} + 1 + w + w^2 + w^3 + ... + w^{m-1} \tag{8}$$

The Mittag-Leffler Theorem for Simple Poles is as follows:

Let $\{z_n\}$ (where z_n is distinct and $|z_n| \to \infty$ as $n \to \infty$) and $\{a_n\}$ be sequences and m is an integer where $m \geq 1$, and

$$\sum_{j=1}^{\infty} \frac{|a_j|}{|z_j|^{m+1}} < \infty \tag{9}$$

Now,

$$f(z) = p_0(z) + \sum_{j=1}^{\infty} (\frac{a_j}{z_j}) L(\frac{z}{z_j}, m) + h(z) \tag{10}$$

is a meromorphic function where the singularities of the function are the simple poles at z_n (which have residue a_n).

We can expand this theory for a general case:

The Mittag-Leffler Theorem for the General Case is as follows:

So $f(z)$ is still a meromorphic function with poles $\{z_j\}$ and principal parts $\{p_j(z)\}$. We can then say that there are polynomials $\{g_j(z)\}_{j=1}^{\infty}$ where $f(z) = p_0(z) + \sum_{j=1}^{\infty}(p_j(z) - g_j(z)) + h(z) = \tilde{f}(z) + h(z)$ and where "$\sum_{j=1}^{\infty}(p_j(z) - g_j(z))$ converges uniformly on every bounded set not containing the points $\{z_j\}_{j=0}^{\infty}$".

1.5 Proof of Mittag-Leffler Theorem

Below is a proof of Mittag Leffler Expansions, pulled from the textbook. We want to take a meromorphic function f(z) with prescribed principal parts $p_j(z) : 0 <= j < \infty$. The principal parts come from the Laurent Expansion of f(z), namely

$$f(z) = \sum_{n=1}^{N_j} \frac{a_{n,j}}{(z-z_j)^n} + \sum_{n=1}^{N_j} b_{n,j}(z-z_j)^n \tag{11}$$

where we call the principal part

$$p_j(z) = \sum_{n=1}^{N_j} \frac{a_{n,j}}{(z-z_j)^n} \tag{12}$$

The aim is to take a given meromorphic function, f(z), with prescribed principal parts $p_j(z) : 0 <= j < \infty$ in terms of suitable functions. We want to find polynomials, $g_j(z) : 0 <= j < \infty$ where $g_0(z) = 0$, such that

$$f(z) = p_0(z) + \sum_{j=1}^{\infty}(p_j(z) - g_j(z)) + h(z) = \tilde{f}(z) + h(z) \tag{13}$$

In the case of simple poles,

$$p_j(z) = \frac{a_j}{z - z_j} = -\frac{a_j}{z_j}(\frac{1}{1 - \frac{z}{z_j}}) \tag{14}$$

Then there is an m such that for $|\frac{z}{z_j}| < 1$, the finite series

$$g_j(z) = -\frac{a_j}{z_j}(1 + \frac{z}{z_j} + ... + \frac{z}{z_j}^{m-1}) \tag{15}$$

for $m \geq 1$ integer. Call

$$L(w, m) = \frac{1}{w - 1} + 1 + w^2 + ... + w^{m-1} \tag{16}$$

then, assuming convergence of the infinite sequence, we can express the meromorphic function as

$$f(z) = p_0(z) + \sum_{j=1}^{\infty}(\frac{a_j}{z_j})L(\frac{z}{z_j}, m) + h(z) \tag{17}$$

From the fact that

$$1 + w + w^2 + ... + w^{m-1} = \frac{1}{1 - w} - \frac{w^m}{1 - w} \tag{18}$$

we have

$$L(w, m) = -\frac{w^m}{1 - w} \tag{19}$$

If $|w| < \frac{1}{2}$ we have $|1 - w| \geq 1 - |w| \geq \frac{1}{2}$, and hence

$$|L(w, m)| \leq 2|w|^m \tag{20}$$

Let |z| $<$ R and for J large enough to take j $>$ J, $|z_j| > 2R$, then $\frac{z}{z_j} < \frac{1}{2}$. Then, the estimate above holds for $w = \frac{z}{z_j}$, and

$$|\frac{a_j}{z_j}L(\frac{z}{z_j}, m)| \leq |\frac{2a_j}{z_j}||\frac{z}{z_j}|^m \tag{21}$$

and thus

$$|\frac{a_j}{z_j}L(\frac{z}{z_j}, m)| \leq \frac{2|a_j||R|^m}{|z_j|^{m+1}} \tag{22}$$

Using z_k and a_k as sequences for these equations above, with z_k distinct, $|z_k| -> \infty$ as $k -> \infty$, and m an integer, with all this such that

$$\sum_{j=1}^{\infty} \frac{|a_j|}{|z_j|^{m+1}} < \infty \tag{23}$$

Then, using equation 22, the Mittag Leffler expansion for f, equation 17, is fully representative of f when f's singularities are simple poles at z_k, with residue a_k for k $=$ 1,2,...

In other words, if equation 23 holds for a meromorphic function, f(z), for its poles and residues, equation 7 holds.

The general case also holds, that is, equation 23 holds for a meromorphic function f(z) with principal parts $p_j(z)$, then there exists polynomials $g_j(z)$, and the series

$$\sum_{j=1}^{\infty}(p_j(z) - g_j(z)) \tag{24}$$

converges uniformly on every bounded set not containing the points z_j.
The details of this proof are incredibly difficult and outside the scope of this project. For this project, we will simply accept this theorem at face value without supporting proof.

2 Development of Examples

Mittag-Leffler Expansion Theorem stated simply:
If a meromorphic function $f(z)$ follows condition (23), then we can express it as

$$f(z) = f(0) + \sum_{n=-\infty}^{\infty} b_n[\frac{1}{z - a_n} + \frac{1}{a_n}] \tag{25}$$

where b_n is the residue(s) of $f(z)$, and a_n are the poles of $f(z)$. We will use this expansion as the basis for our next examples.

2.1 Cotangent's Series Expansion using Mittag-Leffler

Claim:

$$cot(z) = \frac{1}{z} - 2z\sum_{n=1}^{\infty}\frac{1}{\pi^2 n^2 - z^2} \tag{26}$$

Proof: Firstly, we must find $f(0)$. Taking the $\frac{1}{z}$ term to the left side, (26) $\Leftrightarrow$

$$f(z) = cot(z) - \frac{1}{z} = \frac{cos(z)}{sin(z)} - \frac{1}{z}$$

Now, taking the limit as $z \to 0$ and utilizing L'Hopital

$$\lim_{z\to 0} f(z) = \lim_{z\to 0}\frac{zcos(z) - sin(z)}{zsin(z)} = \lim_{z\to 0}\frac{cos(z) - zsin(z) - cos(z)}{zcos(z) + sin(z)}$$

$$= \lim_{z\to 0}\frac{-sin(z) - zcos(z) - sin(z) + sin(z)}{-zsin(z) + cos(z) + cos(z)} = \frac{0}{2} = 0 \Rightarrow f(0) = 0$$

To find the residues, we note that the poles of $f(z)$ occur at $z = n\pi$, $n \in \mathbb{Z}$, thus

$$Res(f(z); z = n\pi) = \lim_{z \to n\pi} (z - n\pi) \frac{zcos(z) - sin(z)}{zsin(z)}$$

$$= \lim_{z \to n\pi} \frac{(cos(z) - zsin(z) - cos(z))(z - n\pi) + (1)(zcos(z) - sin(z))}{zcos(z) + sin(z)} = \frac{n\pi cos(n\pi)}{n\pi cos(n\pi)} = 1$$

employing L'Hopital again. Finally, we put $f(0)$, the pole, and residue into equation (25). This is where we employ the Mitagg-Leffler Expansion Theorem. We yield

$$f(z) = 0 + \sum_{n=-\infty}^{\infty} (1)[\frac{1}{z - n\pi} + \frac{1}{n\pi}] \Leftrightarrow cot(z) - \frac{1}{z} = \sum_{n=1}^{\infty} [\frac{1}{z - n\pi} + \frac{1}{n\pi} + \frac{1}{z + n\pi} - \frac{1}{n\pi}]$$

Making everything pretty

$$cot(z) - \frac{1}{z} = \sum_{n=1}^{\infty} \frac{2z}{z^2 - n^2\pi^2} \Leftrightarrow cot(z) = \frac{1}{z} - 2z \sum_{n=1}^{\infty} \frac{1}{\pi^2 n^2 - z^2}$$

which proves equation (26).

2.2 Deriving Euler's Infinite Sine Product Using Mittag-Leffler Pole Expansion of Cotangent

As an example, we will be deriving the Euler Infinite Sine Product, using cotangent. We start with the definition of the cotangent Mittag-Leffler Pole Expansion:

$$\cot z = \frac{1}{z} - 2z \sum_{n=1}^{\infty} \frac{1}{\pi^2 n^2 - z^2} \tag{27}$$

Now we want to put $\frac{1}{z}$ on the other side of the equation and integrate both sides from zero to a variable (we'll call it β).

$$\int_0^\beta \cot(z) - \frac{1}{z} dz = \int_0^\beta -2z \sum_{n=1}^{\infty} \frac{1}{\pi^2 n^2 - z^2} dz \tag{28}$$

Trying to simplify this, we have:

$$\ln(\sin(z)) - \ln(z) = \sum_{n=1}^{\infty} \int_0^\beta \frac{-2z}{\pi^2 n^2 - z^2} \tag{29}$$

where z is evaluated from 0 to β on the left-hand side. This can be simplified to:

$$\ln(\frac{\sin(\beta)}{\beta}) = \sum_{n=1}^{\infty} \int_0^\beta \frac{-2z}{\pi^2 n^2 - z^2} \tag{30}$$

where z is evaluated from 0 to β. To evaluate the right-hand side, we start by doing a u-substitution. We let $t = \pi^2 n^2 - z^2$ and $dt = -2zdz$. The updated right-hand side is as follows:

$$\sum_{n=1}^{\infty} \int_0^{\beta} \frac{dt}{t} = \sum_{n=1}^{\infty} \ln(\pi^2 n^2 - z^2). \tag{31}$$

Once again where z is evaluated from 0 to β. Now taking into consideration the bounds on our the right-hand side we have:

$$\sum_{n=1}^{\infty} \ln(\pi^2 n^2 - \beta^2) - \ln(\pi^2 n^2) = \sum_{n=1}^{\infty} \ln(\frac{\pi^2 n^2 - \beta^2}{\pi^2 n^2}) \tag{32}$$

Finally, putting both sides together:

$$\ln(\frac{\sin(\beta)}{\beta}) = \ln(\prod_{n=1}^{\infty}(1 - \frac{\beta^2}{\pi^2 n^2}) \tag{33}$$

so now, when we take e of both sides, we can cancel out the natural logarithms on both sides. Doing that, and then transferring the β in the denominator of the left-hand side to the right-hand side, we finish with this equation:

$$\sin(\beta) = \beta \prod_{n=1}^{\infty}(1 - \frac{\beta^2}{\pi^2 n^2}) \tag{34}$$

Which is Euler's Infinite Sine Product. The textbook provides an alternative form of the infinite product of $sin(z)$:

$$\frac{sin(\pi z)}{\pi z} = z \prod_{n=1}^{\infty}(1 - \frac{z^2}{n^2}) \tag{35}$$

This will be useful in the following example where we will find the product expansion of $e^z - 1$.

2.3 Another example: The product expansion of $e^z - 1$

Objective: Find the infinite product expansion of $f(z) = e^z - 1$

Soln: Given that

$$e^z - 1 = e^{\frac{z}{2}+\frac{z}{2}} - e^{\frac{z}{2}-\frac{z}{2}} = e^{\frac{z}{2}} \cdot e^{\frac{z}{2}} - e^{\frac{z}{2}} \cdot e^{-\frac{z}{2}} = e^{\frac{z}{2}}[e^{\frac{z}{2}} - e^{-\frac{z}{2}}]$$

$$= 2ie^{\frac{z}{2}}[\frac{e^{\frac{iz}{2i}} - e^{-\frac{iz}{2i}}}{2i}] = 2ie^{\frac{z}{2}} sin(\frac{z}{2i}) \tag{36}$$

Since we already know that equation (35) holds, from equation (36) we consider that

$$sin\frac{z}{2i} = sin\pi\frac{z}{2\pi i} = \frac{\pi z}{2\pi i} \cdot \frac{\frac{sin\pi z}{2\pi i}}{\frac{\pi z}{2\pi i}}$$

and using (35)

$$= \frac{\pi z}{2\pi i}\prod_{n=1}^{\infty}(1-\frac{(\frac{z}{2\pi i})^2}{n^2}) = \frac{z}{2i}\prod_{n=1}^{\infty}(1+\frac{z^2}{4\pi^2n^2}) \tag{37}$$

Putting the expression from (37) into (36), we get

$$f(z) = e^z - 1 = 2ie^{\frac{z}{2}}\cdot\frac{z}{2i}\prod_{n=1}^{\infty}(1+\frac{z^2}{4\pi^2n^2}) = ze^{\frac{z}{2}}\prod_{n=1}^{\infty}(1+\frac{z^2}{4\pi^2n^2}) \tag{38}$$

as required.

2.4 Numerical Analysis and Extension I

Intro

For this and the next sections, we explored functionality of these series, and applied our knowledge of them to a real engineering solution. Modern-calculators have the ability to calculate trigonometric functions, and it is fairly common knowledge they use series expansions to do this. One series they can use is Taylor series, something commonly known. We will compare performance and accuracy of the series above (Mittag-Leffler and Product), and compare these to Taylor to come to a conclusion if either of these series would perform better for calculators. Note that there is the obvious advantage of using Mittag-Leffler and Product series for complex variables, however we will constrict these series to operate on the real axis. This allows us to fairly compare it to the performance of Taylor series, and apply it to calculator uses. We define cost of calculation to be the amount of time MATLAB takes to complete the calculation, and all errors are absolute, since graphing calculators show 10 digits of accuracy, absolute error matters here. We also explore the convergence, error, and cost of our expansions to analyze their performance as the number of summation iterations increases.

Error Analysis

We used 10 iterations for our analysis on our cotangent expansion, and this is because the Taylor series for cotangent uses Bernoulli numbers which are incredibly expensive to calculate (something worth noting).

Method	Cost	Max Error
Mittag-Leffler	$6.35 * 10^-5$ sec	$9.64 * 10^-3$
Taylor Series	2.37 sec	1.99

Modern calculators definitely shouldn't use Taylor, as the error is huge, as is the cost. Comparing what the 10 iterations from Taylor series to the 10 iterations from Mittag-Leffler, we can conclude that Mittag-Leffler is the obvious choice for use, as our approximation was accurate to at least two digits past the decimal, and it was a lot faster than Taylor series as well. The figures below illustrate the Mittag-Leffler Expansion approximation compared against the exact solution.

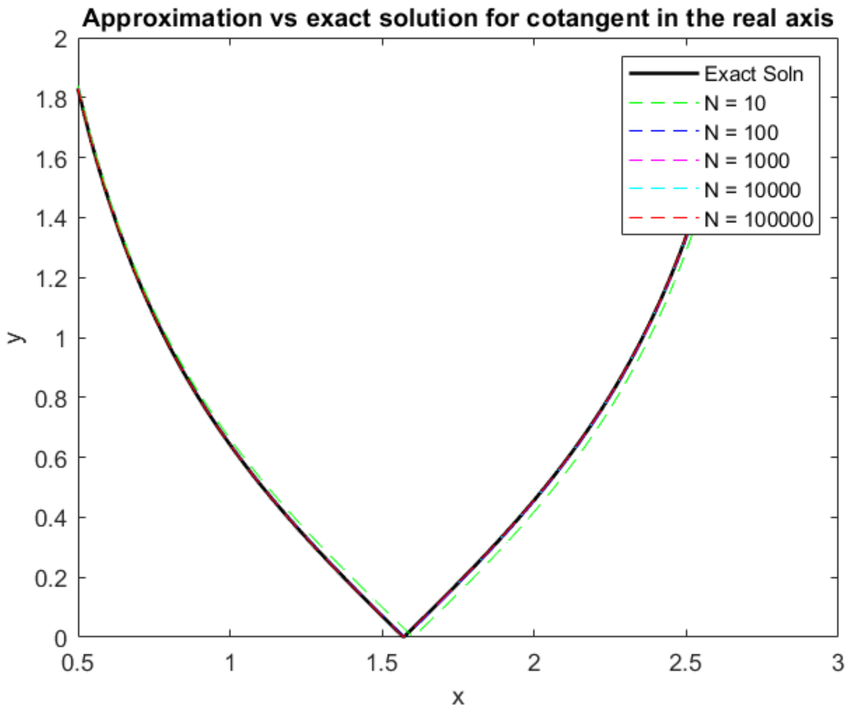

Figure 1: Cotangent Comparison

Figure 1 has different-colored dashed-lines all illustrating a graph of the Mittag-Leffler Expansion for a number of iterations N. We can conclude that for any number of iterations at or above 100, the approximation looks virtually unrecognizable from the exact solution on this graph. Even at ten iterations, our approximation is quite accurate. Therefore, this is yet another piece of evidence to us that the Mittag-Leffler expansion is a great alternative to trying to utilize a Taylor series.

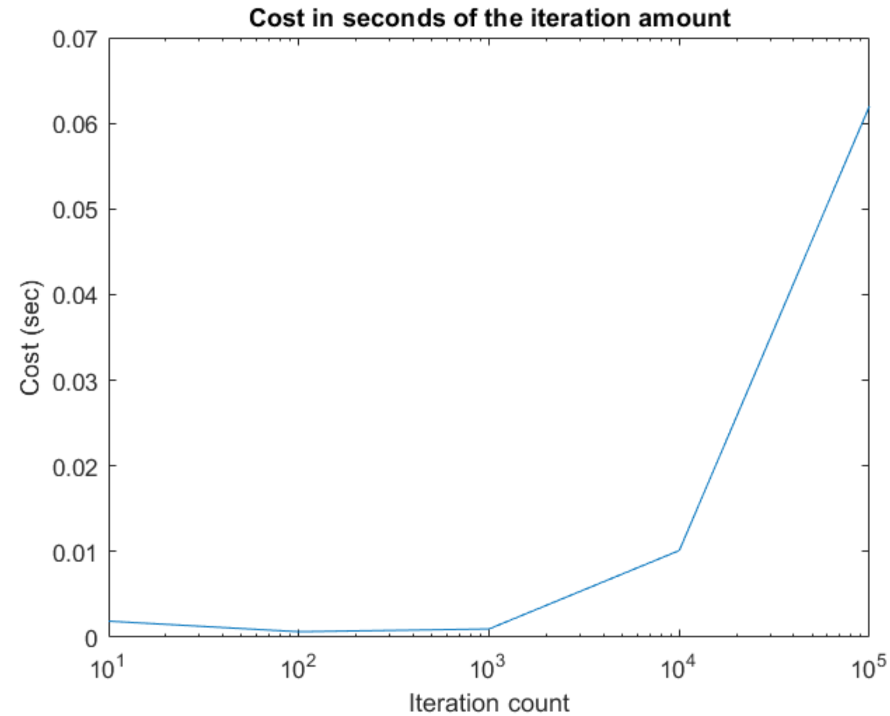

Figure 2: Cotangent Cost

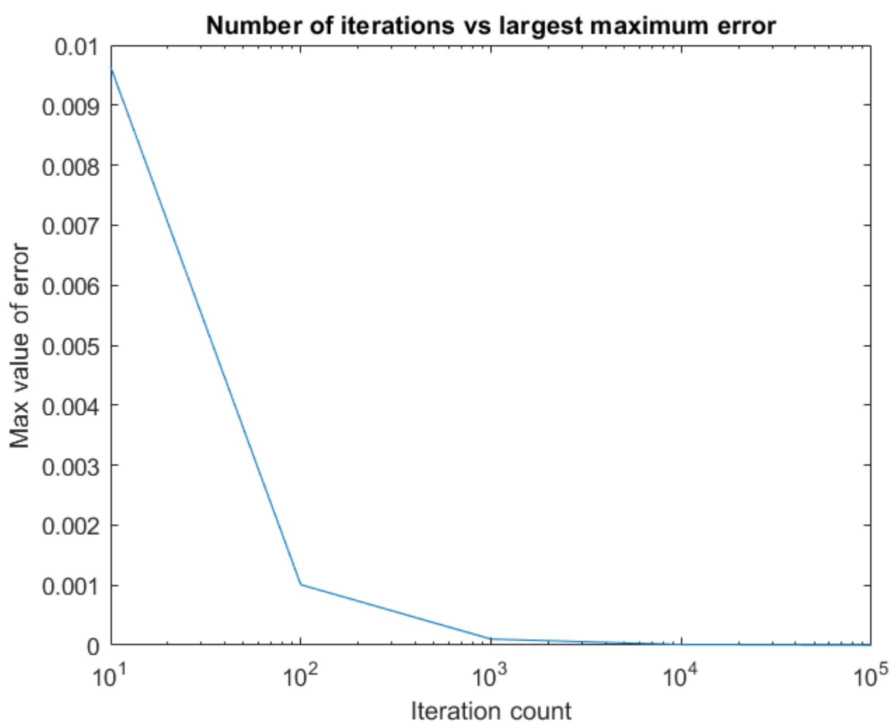

Figure 3: Cotangent Error

Figures 2 and 3 show how the Mittag-Leffler Expansion fares in terms of cost and error. It is interesting on the cost graph to see how for 10^2 iterations and 10^3 iterations the cost is relatively the same, but if we compare the difference between iterations for 10^4 and 10^5 there is a much steeper slope. This is likely due to the extra operations MATLAB has to do to run the methods, so the summation operation is negligible in cost. Similarly, for the number of iterations with our error graph, we can see that for a low amount of iterations, our error is really high, but at around 10^3 iterations, it is almost at zero. These results both make sense, as with a high amount of iterations it will take longer, but our error will be a lot less because we will be approximating much closer to the exact solution. We conclude that for this specific case, the most accurate and efficient result you could get would be around 10^3 iterations, as there is a very small second cost but the error is negligible relative to calculators number of significant figures.

2.5 Numerical Analysis and Extension II

We used 100 iterations for our analysis on our sine expansion. This is because the Taylor series uses a factorial, so anything larger than 100 iterations MATLAB considers to be infinite, since it is outside of the floating point limit.

Method	Cost	Max Error
Weierstrass	$1.79 * 10^-4$ sec	$1.213 * 10^-4$
Taylor Series	0.00726 sec	2.64

It is clear from the table that the computational cost and absolute error for the infinite sine product (value of approximated function vs. sine's actual value) is much less compared to if we chose to use 100 iterations to approximate sine's value with a Taylor series. We can conclude that Weierstrass is the obvious choice for use, due to smaller computational costs and absolute error. Taylor's higher computation cost and absolute error can be attributed to the factorial in the Taylor series. Cost and error simply grows with a growing x value. All in all, it is impractical to use Taylor series in this case if we wish to expect an accurate and timely result for our numerical calculations.

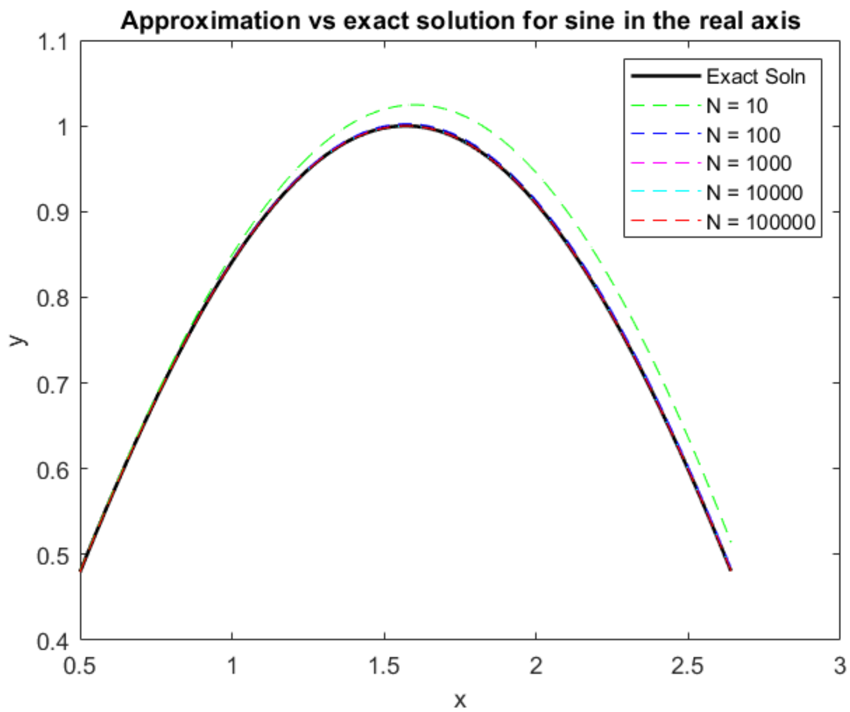

Figure 4: Sine Comparison

Figure 4 shows the Product Expansion approximation for a number of iterations N, this time with the sine function. As you can see here, once again, with 100 or more iterations, the graphs are extremely close to one another. It is interesting that between $x = 0.5$ and $x = 1$ the approximation of 10 iterations is very close to the exact solution; which further points to the rapid convergence capabilities of Product Expansions in some cases. We would recommend using $N = 100$ iterations everywhere the function is not as exact with $N = 10$ iterations.

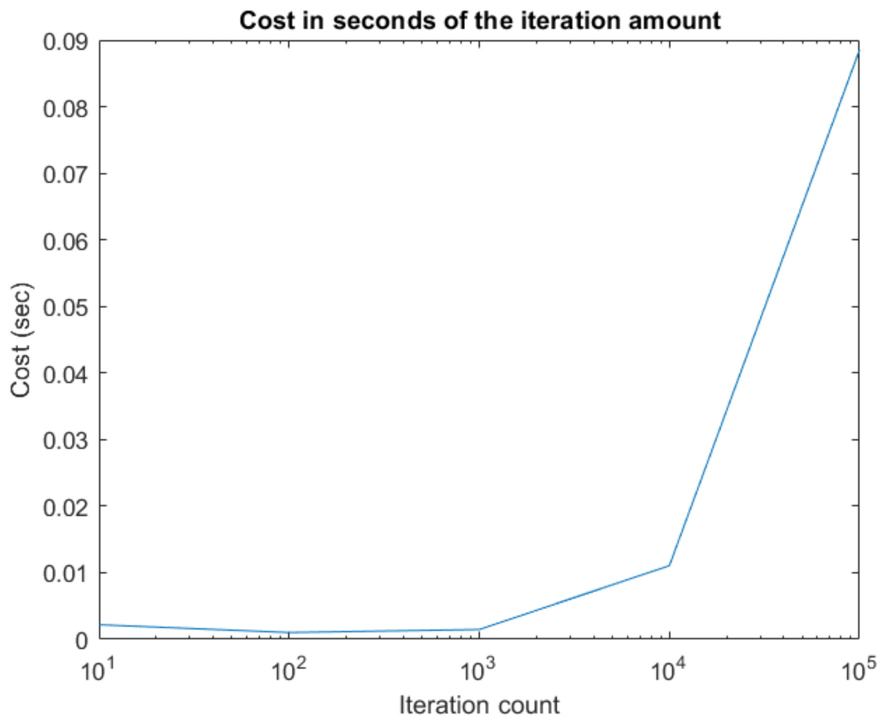

Figure 5: Sine Cost

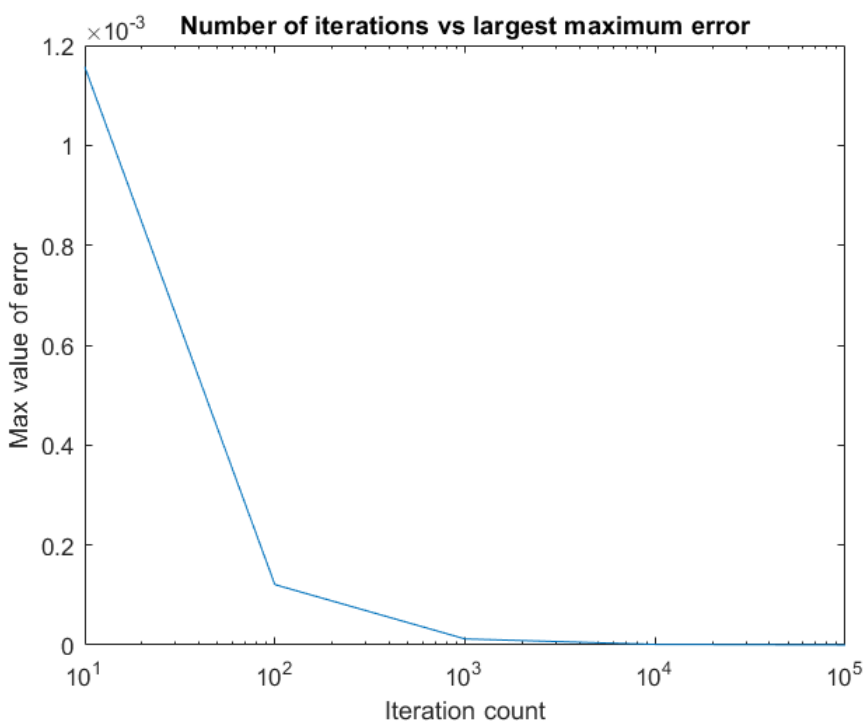

Figure 6: Sine Error

Comparing with figures 2 and 3 of the cotangent expansion, the cost and error functions of the Weierstrass Expansion with the sine function (figures 5 and 6) further illustrate the pattern that with a large amount of iterations the cost is very high but the error is very low, while with a small amount of iterations the cost is very low but the error is very high. It is surprising that the error for this function starts already at 10^-3, while the cotangent expansion had an error starting at 10^-2. This could once again be a byproduct simply because of the function that we chose to approximate, but the most important thing to note here is that the pattern did stay the same. Regardless, we would still recommend with the sine function to use 10^-3 iterations, as there is the lowest cost and the lowest error at that specific number of iterations, leading to high accuracy for calculators significant figure count, but very fast computation speed.

3 Concluding Remarks

3.1 Theoretical discussion

This section analysis poses the question: Can a meromorphic function can be constructed so that it has poles of specified orders at assigned points with no other poles? Or, can an entire function be constructed so that it has zeroes of specified orders at assigned points with no other zeroes? Asking this question leads to certain infinite products and infinite series. Up until this point, we have extended our arsenal in representing functions of a specified type. Namely, we can construct a Mitagg-Leffler expansion for a meromorphic functions and Weierstrass products for entire functions. Earlier on in the course, we only had Taylor and Laurent series representations to work with. In our examples, we began with finding cotangent's Mitagg-Leffler expansion. Then, using this result, found Euler's infinite sine product expansion. Sine's infinite product representation can be used to find several other infinite product expansions of other complex functions, like $f(z) = e^z - 1$, as covered in our final theoretical example.

3.2 Numerical Results

Following our theoretical examples, we chose to analyze the performance of our Mittag-Leffler and Weierstrass Product expansions for sine and cotangent. In particular, analyze the computational cost and absolute error compared to their exact solutions. Using these errors and cost, we can compare the performance to Taylor Series to make engineering decisions as to how scientific and graphing calculators can better optimize their trigonometic function calculations. We determined that using Mittag-Leffler and Weierstrass product expansions are not only far less expensive, but also lead to much less error with proper iteration counts than Taylor series. This leads us to encourage the use of Mittag-Leffler and Weierstrass product expansions in modern calculators.

4 Sources

Ablowitz, Mark J., and Athanassios S. Fokas. Introduction to Complex Variables and Applications. Cambridge University Press, 2021.

"Entire Functions." Wikipedia, Wikimedia Foundation, 23 Dec. 2021, https://en.wikipedia.org/wiki/Entire_function.

"Meromorphic Functions." From Wolfram MathWorld, https://mathworld.wolfram.com/MeromorphicFunction.html.

https://www.youtube.com/watch?v=mJCY9_iEV6Mt=3s

https://www.youtube.com/watch?v=G5foIyCzbQwt=453s

5 Code

Below is the code used for the cotangent numerical analysis

```
%% EVERYTHING IS IN RADIANS

%% COTANGENT
x = 0.5:0.01:pi-0.5;

exact = cot(x);

%% Approximations
% Get approximations and their associated costs
tic
approx10 = get_cot_approx(x, 10);
cost10 = toc;

tic
approx100 = get_cot_approx(x, 100);
cost100 = toc;

tic
approx1000 = get_cot_approx(x, 1000);
cost1000 = toc;

tic
approx10000 = get_cot_approx(x, 10000);
cost10000 = toc;

tic
```

```
approx100000 = get_cot_approx(x, 100000);
cost100000 = toc;

% Comparison to exact
figure(1)
plot(abs(x), abs(exact), k, Linewidth, 1.5);
hold on
plot(abs(x), abs(approx10), g-)
plot(abs(x), abs(approx100), b-)
plot(abs(x), abs(approx1000), m-)
plot(abs(x), abs(approx10000), c-)
plot(abs(x), abs(approx100000), r-)
title(Approximation vs exact solution for cotangent in the real axis)
xlabel(x)
ylabel(y)
legend(Exact Soln,N = 10, N = 100, N = 1000, N = 10000, N = 100000)

%% Error
% Calculation of error
error10 = abs(max((exact - approx10)));
error100 = abs(max((exact - approx100)));
error1000 = abs(max((exact - approx1000)));
error10000 = abs(max((exact - approx10000)));
error100000 = abs(max((exact - approx100000)));
error = [error10, error100, error1000, error10000, error100000];
xaxis_ = [10, 100, 1000, 10000, 100000];

% Plot error
figure(2)
semilogx(xaxis_, error);
title(Number of iterations vs largest maximum error)
xlabel(Iteration count)
ylabel(Max value of error)

%% Cost
cost = [cost10, cost100, cost1000, cost10000, cost100000];
xaxis_ = [10, 100, 1000, 10000, 100000];

% Plot cost
figure(3)
semilogx(xaxis_, cost);
title(Cost in seconds of the iteration amount)
xlabel(Iteration count)
ylabel(Cost (sec))

%% Calculator Analysis
```

```
tic
approx10 = get_cot_approx(x, 10);
cost10 = toc
error10 = abs(max((exact - approx10)))

tic
approx10t = get_cot_taylor(x, 10);
cost10t = toc
error10t = abs(max((exact - approx10t)))

function approx = get_cot_approx(z, n)
    if z == 0
        error(Z must be non-zero)
    end

    if n == 0
        error(N must be non-zero)
    end

    approx = 1./z;
    for index = 1:n
        approx = approx - 2*z./(pi^2*index^2 - z.^2);
    end
end

function approx = get_cot_taylor(z, n)
    if z == 0
        error(Z must be non-zero)
    end

    if n == 0
        error(N must be non-zero)
    end

    approx = 0;
    for index = 1:n
        approx = approx + (-1^index * 2.^(2*index) * bernoulli(2*index) * z.^(2*
            index-1)) / (factorial(2*index));
    end
end
```

Below is the code used for the sine numerical analysis

```
%% EVERYTHING IS IN RADIANS

%% SIN
x = 0.5:0.01:pi-0.5;
```

```
exact = sin(x);

%% Approximations
% Get approximations and their associated costs
tic
approx10 = get_cot_approx(x, 10);
cost10 = toc;

tic
approx100 = get_cot_approx(x, 100);
cost100 = toc;

tic
approx1000 = get_cot_approx(x, 1000);
cost1000 = toc;

tic
approx10000 = get_cot_approx(x, 10000);
cost10000 = toc;

tic
approx100000 = get_cot_approx(x, 100000);
cost100000 = toc;

% Comparison to exact
figure(1)
plot(abs(x), abs(exact), k, Linewidth, 1.5);
hold on
plot(abs(x), abs(approx10), g-)
plot(abs(x), abs(approx100), b-)
plot(abs(x), abs(approx1000), m-)
plot(abs(x), abs(approx10000), c-)
plot(abs(x), abs(approx100000), r-)
title(Approximation vs exact solution for cotangent in the real axis)
xlabel(x)
ylabel(y)
legend(Exact Soln,N = 10, N = 100, N = 1000, N = 10000, N = 100000)

%% Error
% Calculation of error
error10 = abs(max((exact - approx10)));
error100 = abs(max((exact - approx100)));
error1000 = abs(max((exact - approx1000)));
error10000 = abs(max((exact - approx10000)));
error100000 = abs(max((exact - approx100000)));
```

```
error = [error10, error100, error1000, error10000, error100000];
xaxis_ = [10, 100, 1000, 10000, 100000];

% Plot error
figure(2)
semilogx(xaxis_, error);
title(Number of iterations vs largest maximum error)
xlabel(Iteration count)
ylabel(Max value of error)

%% Cost
cost = [cost10, cost100, cost1000, cost10000, cost100000];
xaxis_ = [10, 100, 1000, 10000, 100000];

% Plot cost
figure(3)
semilogx(xaxis_, cost);
title(Cost in seconds of the iteration amount)
xlabel(Iteration count)
ylabel(Cost (sec))

%% Calculator Analysis
tic
approx100 = get_cot_approx(x, 100);
cost100 = toc
error100 = abs(max((exact - approx100)))

tic
approx100t = get_sin_taylor(x, 100);
cost100t = toc
error100t = abs(max((exact - approx100t)))

function approx = get_cot_approx(z, n)
    if z == 0
        error(Z must be non-zero)
    end

    if n == 0
        error(N must be non-zero)
    end

    approx = z;
    for index = 1:n
        approx = approx .* (1 - z.^2./(pi^2*index^2));
    end
end
```

```
function approx = get_sin_taylor(z, n)
    if z == 0
        error(Z must be non-zero)
    end

    if n == 0
        error(N must be non-zero)
    end

    approx = 0;
    for index = 1:n
        approx = approx + z.^(2*index+1)*(-1)^index/factorial(2*index+1);
    end
end
```