World Academy of Science, Engineering and Technology
International Journal of Mathematical and Computational Sciences
Vol:1, No:7, 2007

Computable Function Representations Using Effective Chebyshev Polynomial

Mohammed A. Abutheraa, and David Lester

***Abstract*—**We show that Chebyshev Polynomials are a practical representation of computable functions on the computable reals. The paper presents error estimates for common operations and demonstrates that Chebyshev Polynomial methods would be more efficient than Taylor Series methods for evaluation of transcendental functions.

***Keywords*—**Approximation Theory, Chebyshev Polynomial, Computable Functions, Computable Real Arithmetic, Integration, Numerical Analysis.

I. Introduction

IN this paper we explore the use of Chebyshev Polynomials to represent some computable functions of the computable reals. Our goal is to show that the class of representable functions is sufficiently large to be of interest, and furthermore that such representations are efficient enough to be practical.

In Section 2 we discuss the two approaches to defining computable functions taken by Pour-El and Richards[12]. Whilst more than satisfactory for theoretic purposes, the doubly exponential algorithm implied by the effective Weierstrass Theorem is an almost insuperable problem for practical exploration. In Section 3 and 4 we discuss polynomial approximation and in particular Chebyshev Polynomials. In Section 5, we show that a set of standard transcendental functions can be conveniently represented using Chebyshev Polynomial Approximation. In Section 6 we show that these Chebyshev approximations are more efficient. In Section 7 we show how to manipulate the Chebyshev Polynomials to perform other basic operations including Integration.

II. Approximation

"In recursive analysis, when we consider polynomial approximations to computable real functions, a recursive version of the Weierstrass theorem holds"[5], [6]. The Weierstrass approximation theorem is as follows [5], [6], [12].

Theorem 1: ***(Effective Weierstrass Theorem)*** Let f be a continuous function on [0,1]. Then, for each $k \geq 0$, there is a polynomial φ_k such that $|f(x) - \varphi_k(x)| \leq 2^{-k}$ for all $x \in [0,1]$.

In recursion-theoretic practice, a real number x is viewed as a function $a : N \rightarrow N$. Then a function of a real variable $f(x)$ is viewed as functional. Moreover, a function of computable real variable f(x) maps ϕ from functions a, $a : N \rightarrow N$ into similar functions b. The function f is called computable if the corresponding functional ϕ is recursive. Therefore, there another version of the Weierstrass Theorem in recursive analysis which describe the above. Before stating this definition, two definitions must be stated that are Definition A, and Definition B.

Mohammed A. Abutheraa and David Lester are with the School of Computer Science, University of Manchester, Manchester, email: {mabutheraa,dlester}@cs.manchester.ac.uk

Before going in details in these, it should been considered the function f is defined on a closed bounded rectangle $I^q \subseteq R^q$, where $I^q = \{a_i \leq x_i \leq b_i,\ 1 \leq i \leq q\}$. The end points a_i, b_i are computable real.

Definition 2: ***(Definition A)*** let $I^q \subseteq R^q$ be a computable rectangle. A function f:$I^q \rightarrow R^q$ is computable if:

1) f is sequentially computable,i.e f maps every computable sequence of points $x_k \in I^q$ into a computable sequence $\{f(x_k)\}$ of real numbers.
2) f is effectively uniformly continuous, i.e. there is a recursive function d:N→N such that for x, y$\in I^q$ and all N:

$$|x - y| \leq 1/d(N) \ \ implies \ \ |f(x) - f(y)| \leq 2^{-N}$$

where | | denotes the euclidean norm.

Definition 3: ***(Definition B)*** let $I^q \subseteq R^q$ as in Definition A. A function $f : I^q \rightarrow R$ is computable if there is a computable sequence of rational polynomial which converges effectively to f in uniform norm which means that there is a recursive function $e : N \rightarrow N$ such that for all x $\in I^q$ and all N:

$$m \geq e(N) \ \ implies \ \ |f(x) - p_m(x)| \leq 2^{-N}$$

Theorem 4: ***(Weierstrass Theorem)*** Let [a,b] be an interval with computable end points, and let f be a functions on [a,b] which is computable in the sense of Definition A. Then there exists a computable sequence of polynomials $\{p_m\}$ which converges effectively and uniformly to f on [a,b]-i.e f is computable in the sense of Definition B.

III. Polynomial Interpolation

Kincaid and Cheney stated in their book about how to interpolate a function based on a data set. In other words, if the points are known, then the aim is to find the polynomial which exactly goes through these points.

Theorem 5: If $x_0, x_1, ...x_n$ are distinct real numbers, then for arbitrary values $y_0, y_1, ...y_n$, there is a unique polynomial p_n of degree at most n such that $p_n(x_i) = y_i \quad (0 \leq i \leq n)$[4]. In this paper two forms of Interpolation polynomial will be used in our interpretation which are the Newton form, and the Lagrange form. The Newton form looks as follows:

$$p_k(x) = \sum_{i=0}^{k} c_i \prod_{j=0}^{i-1}(x - x_j) \tag{1}$$

where c_i are the coefficients [4], [7].

The coefficients of the Newton form are calculated using the divided difference. Moreover, the coefficients calculation will be as follows:

- $c_0 = f[x_0] = f(x_0)$
- $c_1 = f[x_0, x_1] = \dfrac{f(x_1) - f(x_0)}{x_1 - x_0}$

So from above we can obtain:

$$\begin{aligned} c_n &= f[x_0, x_1, ..., x_n] \\ &= \frac{f[x_1, x_2, ..x_n] - f[x_0, x_1, ...x_{n-1}]}{x_n - x_0} \end{aligned}$$

On the other hand, the Lagrange is expressed in the following form:

$$p(x) = y_0 l_0(x) + ... + y_n l_n(x) = \sum_{k=0}^{n} y_k l_k(x) \qquad (2)$$

Such that Coordinal functions can be expressed in the following form: $l_i(x) \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}$[4], [7].

There are many ways to approximate a function. Two of the most powerful ways are Taylor Series and Chebyshev Series. In this paper, we are showing that Chebyshev Series provide smaller approximation error than Taylor Series.

IV. Chebyshev Polynomials

Chebyshev polynomials play an important role in numerical analysis. They have been used in approximation of functions, computation of integrals, and solution of differential equations. Also, these polynomials have been used in network synthesis. These polynomials have many properties [2], [8], [9], [10], [13]. Some of them are:

1) Continuous.
2) The least deviation from zero property.
3) The recurrence relation which is an important property for efficient computation.

There are four Chebyshev Polynomials. Their power derives from the close relation with the trigonometric functions 'cos' and 'sin'. The first kind is the most important one and it is the one referred to in most books and articles [9]. In this paper, the first kind Chebyshev polynomial is the one that will be used as well.

The first-kind Chebyshev polynomials are defined recursively by:

$$T_n(x) = cos\ n\theta \ \ when\ x = cos\theta \qquad (3)$$

where x ∈ [-1,1], and $\theta \in [0,\pi]$.

Recurrence relation for this kind of Chebyshev polynomial can also be obtained by:

$$T_n(x) = 2x \cdot T_{n-1}(x) - T_{n-2}(x) \quad n = 2, 3, ... \qquad (4)$$

With initial conditions:

$$T_0(x) = 1 \ \ and \ \ T_1(x) = x \qquad (5)$$

A. Chebyshev Series Expansions

Restricting the range on [-1,1], Chebyshev series expansion of f(x) can be expressed in the form:

$$f(x) \simeq \sum_{i=0}^{\infty} c_i \phi_i(x) \qquad (6)$$

where $\phi_i(x) = T_i(x), U_i(x), V_i(x), \ or \ \ W_n(x)$

So, for the first kind Chebyshev polynomial, the Chebyshev series expansion will be:

$$f(x) \simeq \sum_{i=0}^{\infty}{}' \ c_i T_i(x) = \frac{1}{2} c_o T_o(x) + c_1 T_1(x) + ...$$

where c_i are the coefficients.

B. Chebyshev Interpolation

Theorem 6: ***(Runge phenomenon)*** if x_k are chosen to be the points $x_k = -1 + \dfrac{2k+1}{n+1}$ (k=0,...,n) (means that are equally spaced at a distance $\dfrac{2}{n+1}$ apart), then the interpolating polynomial $p_n(x)$ need not to converge uniformly on [-1,1] as n→ ∞ for the function f(x).

An example function of the above theorem is f(x)=$\dfrac{1}{(1+25x^2)}$. A better choice of interpolating points to ensure uniform convergence, still not necessary for every continuous function, is the set of zeros of the Chebyshev polynomial $T_{n+1}(x)$ where x = $x_k = cos\frac{(k-\frac{1}{2})\pi}{n+1}$ k=1,...,n+1. By expressing polynomial in terms of Chebyshev polynomials, this can make the interpolation more efficient and stable from choosing equally spaced set [9].

C. Chebyshev Nodes

As mentioned in Section IV-B, the set of zeros of the Chebyshev polynomial can make the interpolation more efficient and stable. The Chebyshev nodes of T_{n+1} discussed by Mason and Handscomb[9] can be expressed in one of the following ways:

$$x_k = cos\frac{(k-\frac{1}{2})\pi}{n+1} \qquad (7)$$

$$x_k = cos(\frac{2n+1-2k}{2n+2}\pi) \qquad (8)$$

Where k=0,...,n. Both of the above equations give the same results, but they differ in the order of the results. Furthermore, if the first equation gives the results 1, 0.5,-1, then the results of the second equation will be -1,0.5,1.

D. Chebyshev Approximation Error

The nodes x_i that will be used in our approximation are the roots of the Chebyshev polynomial T_{n+1}. So, as stated in [4], [9] , the error can be defined as follows:

$$|f(x) - P_n(x)| \leq \frac{2(b-a)^{n+1}}{4^{n+1}(n+1)!}\ max_{a\leq x\leq b}|f^{n+1}(x)| \qquad (9)$$

Where a, and b are the end points of the interval [a,b] of approximation, and n is the order of Interpolating polynomial.

Therefore, if we only look at the interval [-1,1], then the following theorem from Kincaid and Cheney[4] holds.

Theorem 7: ***(Theorem on Interpolation error, Chebyshev Nodes)*** If the nodes x_i are the roots of the Chebyshev polynomial T_{n+1}, then error formula will be:

$$|f(x) - p(x) \leq \frac{1}{2^n(n+1)!} max_{|t|\leq 1}|f^{n+1}(t)| \qquad (10)$$

for $|x| \leq 1$.

V. Representing Elementary Functions

In this Section, the Chebyshev series will be used to represent elementary functions such as sin(x), cos(x), exp(x), and log(x). The process of representing these functions using Chebyshev will be done with a known error bound using the equation defined in Section IV-D. Then, complex functions such as an addition , subtraction , multiplication , division, or composite of any two or more elementary functions will be shown how to be represented using Chebyshev.

The functions that are tried to be approximated using Chebyshev in the following Sections are continuous unless stated otherwise.

A. Trigonometric Functions

The process of how to represent cos and sin functions using Chebyshev will be explained. However, only cos(x) will be used in the rest of this Section as the sin and cos behave in similar way.

The Lagrange form of polynomial interpolation as shown in Section III will be used to find the polynomials that best approximate the function cos(x) using Chebyshev. First of all, the Chebyshev nodes need to be calculated as mentioned in Section IV-C. Then, the coefficients need to be calculated as follows:

$$c_0 = \frac{1}{n+1}\sum_{k=0}^{n} f(x_k)\, T_0(x_k) = \frac{1}{n+1}\sum_{k=0}^{n} f(x_k)$$

$$\begin{aligned} c_j &= \frac{1}{n+2}\sum_{k=0}^{n} f(x_k)\, T_j(x_k) \\ &= \frac{1}{n+2}\sum_{k=0}^{n} f(x_k)\, cos(j\frac{2n+1-2k}{2n+2}\pi) \end{aligned}$$

After the coefficients has been calculated, the the interpolating polynomial of f(x) can be defined as follows:

$$f(x) \simeq P_n(x) = \sum_{j=0}^{n} c_j\, T_j(x)$$

This way of approximation can perfectly approximate the functions cos(x) and sin(x). As $n \to \infty$, the error of the polynomial P_n that approximates cos(x) and sin(x) approaches 0. Figure 1 shows how Chebyshev series approximate cos(x).

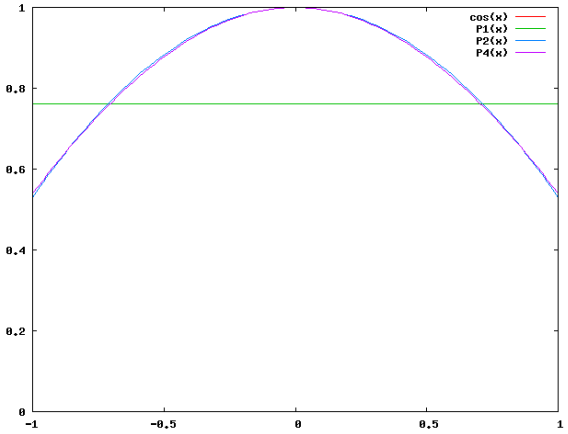

Fig. 1. Approximating cos(x) using Chebyshev

B. Exponential Functions

Exponential functions can be represented using Chebyshev series in the same way as show in the trigonometric functions in Section V-A. Chebyshev series approximate f(x) = exp(x) with small errors that can be calculated using the formula defined in Section IV-D.

The Chebyshev nodes and the coefficients of the interpolating polynomials for the functions f(x) = exp(x) can be calculated exactly like the function f(x) = cos(x). This will gives the following polynomials when the range is [-1,1]:

Similar to the trigonometric functions, Chebyshev series can perfectly approximate the function exp(x). As $n \to \infty$, the error of the polynomial P_n that approximates exp(x) approaches 0. Similar to the trigonometric functions, Chebyshev series can perfectly approximate the function exp(x). As $n \to \infty$, the error of the polynomial P_n that approximates exp(x) approaches 0. Figure 2 shows how Chebyshev series approximate exp(x).

C. Logarithmic Functions

Logarithmic Functions can be represented using Chebyshev series in Newton form explained previously in Section III. Newton form of Chebyshev better approximates the function f(x) = log(x). The error of this approximation approaches 0 as $n \to \infty$. The Newton form uses the divided difference to calculate the coefficients of the interpolating polynomials. This way of calculating the coefficients is time consuming.

Using Chebyshev nodes in Newton form to approximate f(x) = log(x) on [0,4] will gives the better approximation as shown in figure 3.

VI. Comparison between Taylor and Chebyshev

Taylor series is one of the common way that is being used to approximate functions. It is also called power series as it

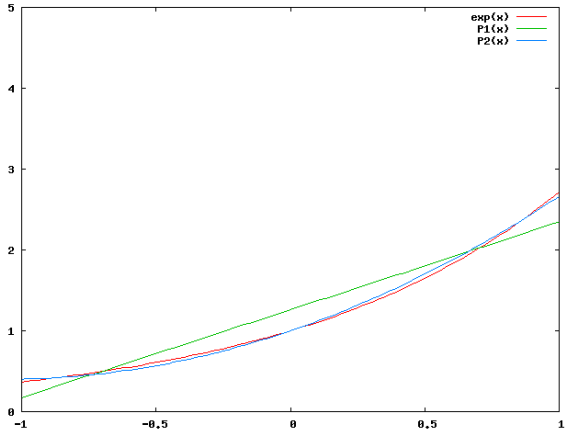

Fig. 2. Approximating exp(x) using Chebyshev

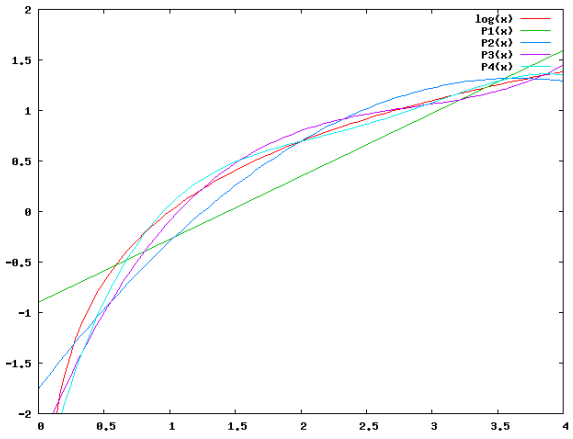

Fig. 3. Approximating log(x) using Chebyshev

is approximate any function as a series of powers. It has a lot of other benefits such as that the Taylor coefficients has been used to derive the characterisation of Laplace transform as shown by Pavlovic and Escardo[11].

The main aim of using Chebyshev nodes for approximation is that it will ensure less level of error than Taylor Series. Taylor series is faster than Chebyshev in calculating the interpolating polynomial as Chebyshev uses sin/cos in its calculation, which consumes time. This all will be shown in Sections VI-B and VI-C.

A. Taylor Series

Taylor series has specific formula for different kind of functions. In this Section, the formula of cos(x),ln(x+1), and exp(x) will be shown.

There is a general formula for Taylor approximation. This formula along with the error for Taylor series can be found in the Taylor's Theorem below defined by Anton[1].

Theorem 8: ***(Taylor's Theorem)*** Suppose that a function f can be differentiated n + 1 times at each point in an interval containing the point a, and let

$$p_n(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \ldots + \frac{f^{(n)}(a)}{n!}(x-a)^n$$

be the nth Taylor polynomial about x = a for f. Then for each x in the interval, there is at least one point c between a and x such that

$$R_n(x) = f(x) - p_n(x) = \frac{f^{(n+1)}(c)}{(n+1)!}(x-a)^{n+1} \qquad (11)$$

B. Time Complexity

Many experiments have been carried out to measure the time for Taylor series, and Chebyshev polynomial to approximate a function. The results of all these tests and experiments have shown that the Taylor series is faster than any polynomial interpolation using Chebyshev nodes.

The time measured in the implementation is in milliseconds. Then, transfered into seconds by multiplying the answer by 10^{-3}. So, the time will be affected by any processes run by the computer during the run time of the program.

For exp(x) and cos(x), the interpolating polynomials are calculated in Lagrange form that is why they are included in one table. On the other hand, ln(x+1) is calculated using Newton form. In Newton form, the coefficients are calculated using divided difference. Divided difference is time consuming and it will take long time to compute.

It can be concluded that Taylor series is faster than polynomial interpolation using Chebyshev nodes. However, if it has been assumed that the cos values of the Chebyshev nodes is been already calculated, then the time complexity will change and will make the Chebyshev approximate functions faster. This will works only in the Lagrange form of Chebyshev approximation as the more time is spent in the evaluating of cos. On the other hand, this will not affect the Newton form of the Chebyshev approximation as it spends large amount of time in the divided difference calculations of the coefficients.

In the next Section it will be proved that the error for Chebyshev is much smaller than the error in Taylor.

C. Error Analysis

In the previous Section, it has been found that the Taylor series is faster than Chebyshev in time. However, in this Section we will prove that Chebyshev will ensure much smaller error than Taylor when the cos/sin values are not recorded. However, if the cos/sin values are recorded then Chebyshev will be faster if it is represented in Lagrange form.

TABLE I
TAYLOR VS. CHEBYSHEV ERROR VALUES FOR COS(X) AND EXP(X)

	exp(x)		cos(x)	
n	Chebyshev	Taylor	Chebyshev	Taylor
0	2.7182	2.7182	1	0.841
5	0.0001	0.0037	0.00004	0.0007
10	6.65×10^{-11}	6.80×10^{-8}	2.44×10^{-11}	2.10×10^{-8}
15	3.96×10^{-18}	1.29×10^{-13}	1.45×10^{-18}	2.58×10^{-14}
20	5.07×10^{-26}	5.32×10^{-20}	-1.88×10^{-26}	1.64×10^{-20}
25	2.008×10^{-34}	6.74×10^{-27}	7.38×10^{-35}	1.34×10^{-27}
30	3.07×10^{-43}	3.30×10^{-34}	1.13×10^{-43}	1.02×10^{-34}

TABLE II
TAYLOR VS. CHEBYSHEV ERROR VALUES FOR LOG(X)

	exp(x)	
n	Chebyshev	Taylor
0	0.5	1
5	0.0001	1
10	1.19×10^{-7}	1
15	1.16×10^{-10}	1
20	1.13×10^{-13}	1
25	1.11×10^{-16}	1
30	1.08×10^{-19}	1

The experiments for error analysis are carried out using Java. The data type used to store the error is Big Decimal. Big Decimal data type in Java allows us to have more accurate results. It provides for immutable arbitrary-precision signed decimal numbers. However, the results that will be shown will be rounded to suitable number of decimals for viewing purpose. The error calculation for both approximation ways can be seen in Tables I and II.

When calculating the error for Taylor series, some assumption has been made. For the Taylor's error equations in VI-A, the following values has been given to the variables: c = 1, a = 0 and x = 1. These values have been chosen to make the comparison between Taylor and Chebyshev as fair as possible.

On the other hand, the range for Chebyshev approximation varies. For f(x) = cos(x) and f(x) = exp(x), the range used in the error calculation is [-1,1]. However, for f(x) = ln(x), the range used is [0,1].

It can be concluded from this Section, that Chebyshev will ensure smaller error that Chebyshev. So, Chebyshev approximates continuous functions much better than Taylor.

VII. OTHER OPERATIONS ON CHEBYSHEV POLYNOMIALS

In this Section, a way of using Chebyshev to represent complex functions is described. Complex functions, is a mixture of the elementary functions discussed in Section V. Furthermore, complex function can be defined by doing an operation between two or more elementary functions. Operations between functions vary; it can be addition, subtraction, multiplication, division and composition. An example of such functions can be $\frac{exp(x)}{sin(x)}$, log(exp(x)), and cos(x) $\times$ log(x).

Java programming language and C programming language has been used to implement operations such as addition, subtraction, multiplication, division, and composition. Anton has defined in [1] the basic operations on functions. The following is his definition on operations between functions.

Definition 9: Given functions f and g, their sum f + g, difference f - g, product f $\cdot$ g, and quotient f/g are defined by

- (f + g)(x) = f(x) + g(x)
- (f - g)(x) = f(x) - g(x)
- (f $\cdot$ g)(x) = f(x) $\cdot$ g(x)
- (f / g)(x) = f(x) / g(x)

For the functions f + g, f - g, and f $\cdot$ g the domain is defined to be the intersection of the domains of f and g, and for f/g the domain is this intersection with the points where g(x)=0 excluded.

D'Aguanno, Nobile, and Roman has implemented some of these operations in FORTRAN [3].

A. Addition and Subtraction

In this Section, an explanation of how to approximate functions such as exp(x) + cos(x) or cos(x) - log(x) using Chebyshev is explained. As mentioned in Section V, elementary functions can be approximated by Chebyshev series. So, for functions like exp(x) + sin(x), we will have a Chebyshev series for each term. To approximate the whole function f(x) = exp(x) + cos(x), an addition operation must be applied to the two Chebyshev series. This has been implemented using Java, and it is proved to work. An addition operation can be applied to two or more Chebyshev series of any orders.

For example, in adding two Chebyshev polynomials $T_2(x) = -1.0 + 2.0x^2$ and $T_5(x) = 5.0x - 20.0x^3 + 16.0x^5$. The result of this computation is $-1.0 + 5.0x + 2.0x^2 - 20.0x^3 + 16.0x^5$. So, any two functions can be added together.

This is also the case when dealing with subtraction operation. So, any complex function ,which contains a mixture of the elementary functions, with addition or subtraction operations can be approximated or represented using Chebyshev.

Figure 4 shows how Chebyshev is able to approximate the function $f(x) = exp(x) + cos(x)$ and Figure 5 shows how Chebyshev is able to approximate the function $f(x) = exp(x) - cos(x)$.

B. Division and Multiplication

The long division between Chebyshev polynomials has been implemented. The result of this operation will give two polynomials. One of them is the answer of long division and the other one is the reminder of this division. For example, when trying to divide $T_6(x) = 32x^6 - 48x^4 + 18x^2 - 1$ by $T_3(x) = 4x^3 - 3x$, then the result of such operation is $8x^3 - 6x$ and the reminder is -1.

Therefore, three methods have been written. One, to handle the division process. The second one is to return the answer of the division, and the last one will return the reminder of the division.

On the other hand, multiplication between Chebyshev polynomials has been implemented as well. The answer of this will give the result polynomial. The process involves multiplying each term from one polynomial by all the terms in the second polynomial. Then, an addition operation shall be done to add the terms with the same power.

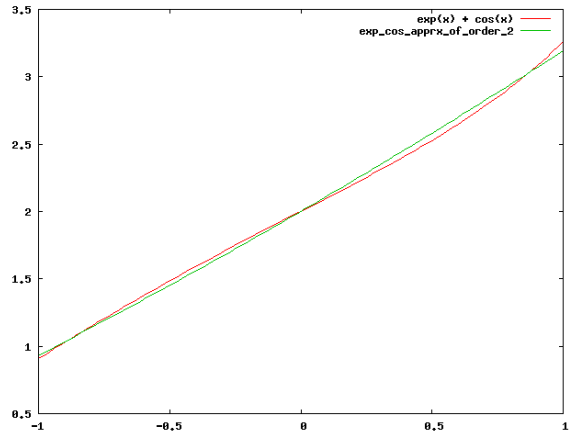

Fig. 4. Approximating exp(x) + cos(x) using Chebyshev

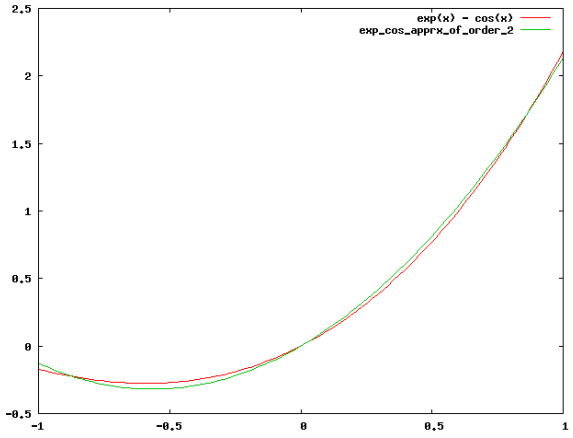

Fig. 5. Approximating exp(x) - cos(x) using Chebyshev

If we have the two polynomials: $3x^2 + x + 10$ *and* $2x^2 + 3x + 2$, then the result of multiplying these polynomials is : $20.0 + 36.0x + 35.0x^2 + 15.0x^3 + 6.0x^4$. Figure 6 shows how Chebyshev is able to approximate the function $f(x) = exp(x) * cos(x)$.

As a result of that, functions that consist of the multiplication or division between elementary functions can be approximated using Chebyshev. Moreover, this will allow us to approximate functions such as $\frac{exp(x)}{cos(x)}$ or exp(x) $\times$ log(x).

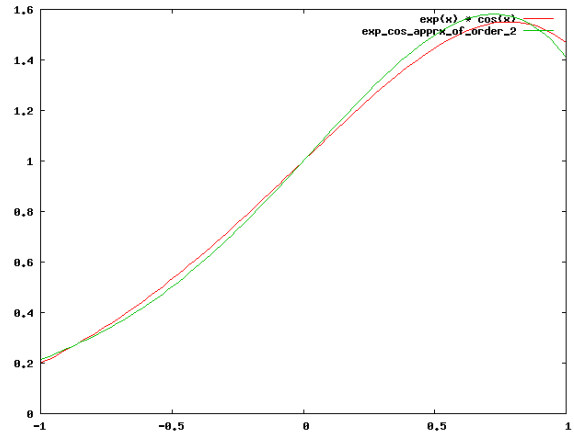

Fig. 6. Approximating exp(x) * cos(x) using Chebyshev

C. Composition

One of the most difficult functions to approximate is the one that has a composite operation within its terms. A composite is an operation on functions. It can be written as f(g(x)) or (f $\circ$ g)(x). The following is a definition of composition by Anton[1].

Definition 10: Given functions f and g, the composition of f with g, denoted $f \circ g$, is the functions defined by

$$(f \circ g)(x) = f(g(x))$$

The domain of $f \circ g$ is defined to consist of all x in the domain of g for which $g(x)$ is in the domain of f.

At the end, it has been managed to implement composition of functions using Java. For example, if f(g(x)) need to be calculated where f(x) = $T_4(x) = 1.0 - 8.0x^2 + 8.0x^4$ and g(x) = $T_3(x) = -3.0x + 4.0x^3$. Then the result of the composite operation is f(g(x)) = $1.0 - 72.0x^2 + 840.0x^4 - 3584.0x^6 + 6912.0x^8 - 6144.0x^{10} + 2048.0x^{12}$. Figure 7 shows how Chebyshev is able to approximate the function $f(x) = exp(cos(x))$.

This result will enable us to use the composition operation between any two functions. Therefore, a composite function which consists of elementary functions can be approximate using Chebyshev. Simply, by approximating each function alone using Chebyshev series. Then the composite of theses two approximation can be calculated.

D. Integration

These polynomials have important properties such as the least deviation from zero property, the recurrence relation which is an important property for efficient computation, and the fact that there are continuous. These properties will make the integration over these polynomials and there series more efficient and easy.

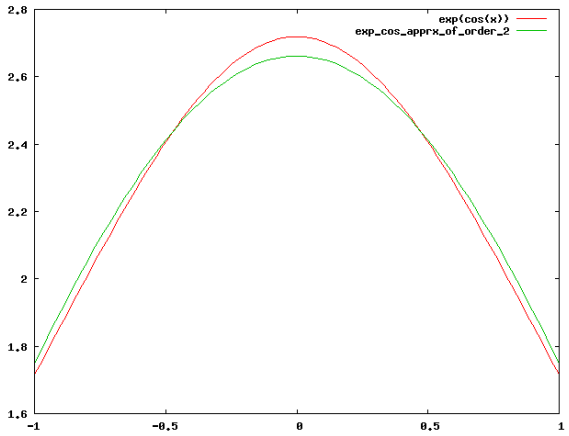

Fig. 7. Approximating exp(cos(x)) using Chebyshev

The error of evaluating the integration of these functions over a closed interval [a,b] can be easily estimated. This is due to that we can approximate the functions by Chebyshev with a defined error bound.

E. A Counter-Example

In the previous Sections, the functions that we looked at are only continuous functions. So, in this Section we will be looking at discontinuous functions. The discontinuous function that is looked at here is the step functions. Step functions can be defined as follows on [-1,2):

$$f(x) = \begin{cases} -1 & -1 \leq x < 0 \\ 0 & 0 \leq x < 1 \\ 1 & 1 \leq x < 2 \end{cases}$$

The function need to be differentiable n+1 times in order for Chebyshev to approximate it. However, step functions can not be differentiated n+1 time. Thus, Chebyshev cannot approximate step functions. As a result, Chebyshev cannot be used directly to represent discontinuous functions. Note that we could use piecewise approximation in the case of functions that are discontinuous at only finitely many points.

VIII. Conclusion and Future Work

In this paper, we have described the polynomial approximation to functions. Partically, it has been shown that Chebyshev can approximate functions with much less error than Taylor. Also, if we record sin/cos values needed, we will get the results in much less time than Taylor series.

Chebyshev play a great role in the Approximation Theory. Moreover, Chebyshev can only approximate continuous functions on the interval [a,b]. However, it has been shown in Section VII-E that it can not approximate discontinuous functions such as step functions. This is due to that the functions are not differentiable n+1 times when we are trying to approximate it with n order Chebyshev for some integer n.

Chebyshev polynomials and series have been shown to be superior in terms of accuracy and computational efficiency over Taylor series. Also, it has been shown that they can perfectly approximate elementary funcitons, and complex functions(addition, subtraction, multiplication, division, or composition of two or more elementary functions). In addition we can easily integrate these functions.

One interesting line of further work is to see whether we can incorporate some of the ideas of Brisebarre, Muller, and Tisserand[2] on machine-efficient Chebyshev Polynomials Approximations. There are also applications to the solution of integral equations that might prove faithful.

Acknowledgement

The authors would like to thank Dr. Len Freeman for his review feedback.

References

[1] Howard Anton, *Calculus With Analytic Geometry*, Fourth edition. Anton Textbooks, Inc., 1992.
[2] Nicolas Brisebarre, Jean-Michel Muller, and Arnaud Tisserand, *Computing Machine-Efficient Polynomial Approximation*. ACM Transactions on Mathematical Software, Number 2, Volume 32,pp. 236-256 ,2006.
[3] B. D'Aguanno, A. Nobile, and E. Roman, *CHPACK: A Package For The Manipulation of Chebyshev Approximations*. Computer Physics Communications, Number 29, pp. 361-374 ,1983.
[4] David Kincaid and Ward Cheney, *Numerical Analysis: Mathematics of Scientific Computing*. Brooks/Cole, 2002.
[5] Ker-I Ko, *On the Computational Complexity of Best Chebyshev Approximations*. Complexity, Number 2,pp. 65-120 ,1986.
[6] Ker-I Ko, *Complexity Theory of Real Functions*. Boston: Birkhauser, 1991.
[7] Roland E. Larson, Robert P. Hostetler, Bruch H. Edwards and David E. Heyd, *Calculus with Analytic Geometry*. D. C. Heath and Company, 1994.
[8] J. Mason, *Chebyshev Polynomials: Theory and Applications*. Kluwer Academic, 1996.
[9] John C. Mason, and David C. Handscomb, *Chebyshev Polynomials*. CRC Press Compan, 2003.
[10] Jean-Michel Muller, *Elementary Functions: Algorithms and Implementation*. Birkhauser, 1997.
[11] D. Pavlovic and M.H. Escardo, *Calculus in Conductive Form*. Thirteenth Annual IEEE Symposium on Logic in Computer Science, pp. 408-417 ,1998.
[12] Marian Pour-El and J. Ian Richards, *Computability in Analysis and Physics*. Berlin: Springer-Verlag, 1989.
[13] Theodore J. Rivlin, *Chebyshev Polynomials: from approximation theory to algebra and number theory*. New York, Chichester : Wiley, 1990.

Mohammed A. Abutheraa Abutheraa is currently a PhD student in the School of Computer Science at the University of Manchester. He has got his Bachelor of Science degree on Software Engineering from the same University. His main research topic is computer arithmetic. He is specifically working in Exact Arithmetic.

David Lester Lester is a Lecturer in the Department of Computer Science at the University of Manchester. He is member of Advanced Processor Techology and Foundation of Mathematics research groups.