Numerical Methods for Approximating Functions Using MATLAB Program

D.A.Gismalla

***Abstract*— Methods of approximations are used to approximate well-defined differentiable functions on a given interval [a,b]. Further , some approximations can be used to interpolate the values of functions whose statical values are known at their corresponding points. This means that approximation can be used as an interpolation or vice versa. That is, some method can be used for an interpolation or approximation such as Taylor's Series provided that the function is differentiable at the given interval [a,b]. One disadvantage of Taylor's Series is that it concentrates all the information at a single point x_0 and the evaluations of the n^{th} derivative is sometimes difficult .Other methods for approximation that We shall consider are the Trigonometric functions, Legendre's Polynomials and Cheybeshev function of the first kinds. All these methods are the primary fundamental methods that can generate more sophisticated methods. Furthermore, Matlab Software Programs are written for each one with the explanations in the Computational Remarks to overcome some difficulties arise to approximate any particular function such as exp(x) and $\frac{\ln(1+x)}{\ln(1-x)}$ .Chebyshev for $\frac{\ln(1+x)}{\ln(1-x)}$ *see*[1]&[4].**

***Index Terms*— Chebyshev & Clenshaw–Curtis**

I. TAYLOR SERIES EXPANSION

Taylor series expansion is expressed with its error estimate as

$$y(x+h) = \sum_{k=0}^{n} \frac{h^k y^k(x)}{k!} + R_{n+1} \qquad (1)$$

where the error is given by

$$R_{n+1} = \frac{y^{n+1}(\varepsilon(x))}{(n+1)!} h^{n+1} \qquad (2)$$

Example 1 Find a polynomial of degree three to approximate the exponential e^x at x=0.5 . Compare the Actual Error with Approximated Error.

Solution

First , We expand the exponential e^x at x=0 using the formula (1).

Now , let $f(x)= e^x \Rightarrow f'(x)= e^x$, $f''(x) = e^x$, $f'''(x)= e^x$, $f''''= e^x$

and $f'(0)= f''(0)= f'''(0)= f''''(0)=1$

Therefore, Eqn.(1) gives $f(x) = e^x \approx \sum_{k=0}^{3} \frac{x^k}{k!}$

which is a polynomial of degree three.

$$\therefore e^x \approx 1+x+\frac{x^2}{2}+\frac{x^3}{6} \Rightarrow e^{1/2} \approx$$

$$1+0.5+\frac{(0.5)^2}{2}+\frac{(0.5)^3}{6} = 1.6458333333$$

$\therefore$ The Exact Error E_a=

abs($e^{1/2}$ -1.6458333333)

= 2.8879373-03

The Approximated Error by Eqn.(2) is

$$= \max_{\varepsilon\in[0,1]} \frac{\frac{d^4}{dx^4} e^{(\varepsilon(x))}}{4!} 1^4 \quad E_a$$

$$= \frac{e}{24} = 0.11326174$$

Observe that the actual Error is less than the theoretical Approximated Error that doesn't contradicts the theory of approximations and the Power Series for the exponential e^x about x=0 can expressed as

$$e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!} \qquad (3)$$

and similarly the Power Series around the point x=0 for sin(x) , cos(x) and tan(x) can be found by Eqn.(4) , Eqn.(5) , and Eqn.(6), respectively .

$$\sin(x) = \sum_{k=0}^{\infty} \frac{(-1)^k x^{2k+1}}{(2k+1)!} \qquad (4)$$

$$\cos(x) = \sum_{k=0}^{\infty} \frac{(-1)^k x^{2k}}{(2k)!} \qquad (5)$$

$$\tan(x) = \sum_{p=1}^{\infty} \frac{2^{2p}(2^{2p}-1)}{(2p)!} \mathbf{B}_p \, x^{2p-1} \qquad (6)$$

where B_p are the Bernoulli numbers and the first Bernoulli numbers are

$B_1 = 1/6$ $B_2 = 1/30$ $B_3 = 1/42$ $B_4 = 1/30$ $B_5 = 5/66$

The three equations Eqn.(4) , Eqn.(5) , and Eqn.(6) lead to what is known as **approximation of trigonometric functions** as in the following Example 2.

D.A.Gismalla, Present Address: Taif University , Riana College, Dept. of Mathematics, Riana ,Sudai Arabia
Permanent Address : Gezira University , Faculty of Mathematics , P.O.Box 20, Wad Medani , SUDAN

Example 2.
Approximate cos(0.5) using the first six terms from Eqn.(5)
Solution
Here ,We substitute , the angle $\theta = 0.5$ in radians in Eqn.(5)
cos(0.5) = 0.877582465 while the exact value is 0.8775825

II. MATLAB PROGRAM FOR TAYLOR'S METHOD

Example 3 (a)
This Example 3(a) is meant as demonstration for **Taylor Series in MATLAB .** The MATLAB command for a Taylor polynomial is taylor(f,n+1,a), where f is the function, a is the point around which the expansion is made, and n is the order of thee polynomial. We can use the following code in Fig.(1)to express the function sin(x)

Example 3(b) Find the Taylor polynomials of orders 1, 2, 3, and 4
near x = 1 for f(x) = ln x. as in Fig.(2)

Example 3(c). Use a Taylor polynomial around x = 0 to approximate the $f(x) = e^x$,with an accuracy of .0001 ,Fig(3).First, we observe that for $f(x) = e^x$.

```
>>   syms x;
>> f=inline('sin(x)') ;
>>taylor(f(x),2,0)
 ans = x
>>taylor(f(x),4,0)
 ans = x-1/6*x^3
>>taylor(f(x),6,0)
ans =  x-1/6*x^3+1/120*x^5
```

Fig.(1) Command window to express the function sin(x) using Taylor series

```
>>  syms x
>> f=inline('log(x)')
   f = Inline function: f(x) = log(x)
>>  taylor(f(x),2,1)
    ans = x-1
 >> taylor(f(x),3,1)
   ans = x-1-1/2*(x-1)^2
 >> taylor(f(x),4,1)
  ans =  x-1-1/2*(x-1)^2+1/3*(x-1)^3
 >>  taylor(f(x),5,1)
  ans =x-1-1/2*(x-1)^2+1/3*(x-1)^3-1/4*(x-1)^4
```

Fig.(2) Taylor polynomials of orders 1, 2, 3, and 4 near x = 1 for f(x) = ln x

```
>> syms x
>> f=inline('exp(x)');
 >> taylor(f(x),7,1)
```

Fig.(3) Taylor polynomials of order 6 near x = 1 for f(x) = exp(x)

```
% function maclorin with m_file :-
% command windows:-    f(x)=exp(x) ,n=8 ?
% syms x;
% f=inline('exp(x)');
% n=8;
% pn=maclor(f,n)

function pn=maclor(f,n)
syms x;
y=f;
for i=1:n
   g=inline(diff(y(x),i));
   pn(i)=feval(g,0)/factorial(i);
end
disp(' The Macularin  series =');
```

```
>>  syms x;
>>  f=inline('exp(x)');
>>  n=8;
>>  pn=maclor(f,n)
The Macularin  series =
 pn =
   1.0000   0.5000   0.1667   0.0417
   0.0083   0.0014   0.0002   0.0000
```

Fig.(4) shows Maclurain's series for e^x at $x_0=0$

```
% The taylor function with m_file :-taylor11.m
% command windows:- f(x)=exp(x)  , n=5 ,xo=2?
% >>syms x;
% >>f=exp(x);
% >>n=5;
% >>xo=2;
% >> format short
% >> pn=taylor11(f,5,2)
function pn=taylor11(f,n,xo)
syms x;
y=f;
p(1)=feval(y,xo);
for j=1:n
   t(j)=(x-xo)^(j);
end
tt=[1 t]
for i=1:n
    g=inline(diff(y(x),i));
    p(i+1)=feval(g,xo)/factorial(i);
end
p
pptt=tt.*p
pn=sum(pptt);
end
disp('the taylor function approximately=');
```

Fig.(5) The file *taylor11.m*
Different from Taylor(f ,n+1,x_0)

we have $f^{(k)}(x) = e^x \text{ for all } k = 0,1,2,3,\ldots.$
Consequently, the Taylor expansion for e^x is

$$e^x = 1 + x + \frac{1}{2}x^2 + \frac{1}{6}x^3 + \cdots$$

If we take an n^{th} order approximation, the error is $\frac{f^{(n+1)}}{(n+1)!}(c)x^{(n+1)}$

where $c \in (a, b)$. Taking a = 0 and b = 1 this is less than

$$\sup_{c\in(0,1)} \frac{e^x}{(n+1)!} = \frac{e}{(n+1)!} \le \frac{3}{(n+1)!}$$

Notice that we are using a crude bound on e, because if we are trying to estimate it, we should not assume we know its value with much accuracy. In order to insure that our error is less than .0001, we need to find n such

$$\frac{3}{(n+1)!} \prec 0.0001$$

Trying different values for n in MATLAB, we eventually find

>>3/factorial(8)

ans = 7.4405e-05

which implies that the maximum error for a 7th order polynomial with a = 0 and b = 1 is .000074405. That is, we can approximate e with

$$e^1 = 1 + 1 + \tfrac{1}{2}+\tfrac{1}{3}+\tfrac{1}{4}+\tfrac{1}{5}+\tfrac{1}{6}+\tfrac{1}{7} = 2.71825,$$

which we can compare with the correct value to five decimal places

e = 2.71828.

The error is 2.71828 − 2.71825 = .00003. ∆

Further, We know when x_0=0 then Taylor's series expansion is called Maclurain's series. Fig.(4) shows Maclurain's series for e^x at x_0=0

Furthermore, instead of the Built-in function for ***Taylor(f ,n+1,x$_0$),***We wrote another function in the file ***taylor11.m*** as can be seen in the Fig.(5)

```
>> syms x;
>> f=inline('exp(x)');
>> n=5;
>> xo=2;
>> format short
>> pn=taylor11(f,n,2)
tt = [      1,    x-2, (x-2)^2, (x-2)^3,
          (x-2)^4,       (x-2)^5]
p =   7.3891    7.3891    3.6945
      1.2315    0.3079    0.0616
pptt =[  4159668786720471/562949953421312,
4159668786720471/562949953421312*x-
4159668786720471/281474976710656,
4159668786720471/1125899906842624*(x-2)^2,
……..]
the taylor function approximately=

pn =-4159668786720471/562949953421312+
4159668786720471/562949953421312*x+
4159668786720471/1125899906842624*(x-2)^2+…….
.]
```

Fig.(5) The Command Window for The file *taylor11.m Different from* Taylor(f ,n+1,x$_0$)

III. Orthogonal Polynomials Approximation

The set of linearly independent polynomials

$$\left\{\varphi_j(x)\right\} \quad j = 0(1)n$$

are called **Orthogonal Polynomials** with respect to the weight function w(x) on the i

nterval [a , b] whenever the following condition holds.

That is whenever $\int_a^b w(x)\ \varphi_j(x)\ \varphi_k(x)\,dx$

$$= \delta_{jk} = \begin{cases} 0 & if\ j \neq k \\ 1 & if\ j = k \end{cases} \qquad (7)$$

where δ_{jk} is called the kronecker delta.

Now any function f(x) that

is integrable with respect to the weight function w(x) on the interval [a , b] can be approximated with these **Orthogonal Polynomials** . Expand f(x) as

$$f(x) \approx \sum_{j=0}^{n} a_j \varphi_j(x) \qquad (8)$$

Hence , from Eqn.(7) the coefficients a_j j=0(1) n can be determined using the orthogonality process by multiplying Eqn.(8) with the weight function w(x) and the polynomials $\varphi_k(x)$ k=0(1) n to get

$$a_k = \int_a^b w(x)\ f(x)\ \varphi_k(x)\,dx$$

k=0(1)n (9)

There are many orthogonal polynomials that can be found depending on the weight

function w(x) by the **Gram-Schmidt Process** in [1] ,page(380) . Some of these polynomials

function, Legendre's and Chebyshev

polynomials

IV. Trigonometric Polynomials

Approximation

The set of orthogonal trigonometric polynomials with respect to the weight

function w(x)=1 on the interval [-π , π] can found to be

$$\varphi_0(x) = \frac{1}{\sqrt{2\pi}}$$

$$\varphi_k(x) = \frac{1}{\sqrt{\pi}}\cos(kx)\ \ for\ each\ \ k = 1,2,\ldots,n$$

and

$$\varphi_{n+k}(x) = \frac{1}{\sqrt{\pi}}\sin(kx)\ \ for\ each\ \ k = 1(1)n-1$$ Now if

We combine these orthogonal trigonometric polynomials in the

Set $\left\{\varphi_k(x)\right\}_{k=0}^{k=2n-1}$, then many function can be approximated if they are represented by an equation Eqn.(10) similar to

Eqn.(8) as $f(x) \approx \sum_{j=0}^{2n-1} a_j \varphi_j(x)$ (10)

which implies that

$$a_k = \int_{-\pi}^{\pi} f(x)\ \varphi_k(x)\,dx \quad k=0,1,2,\ldots,2n-1 \quad (11)$$

Eqn.(10) is called the **trigonometric polynomial** for approximation

Example 4(a) Find the trigonometric polynomial to approximate

$$f(x) = |x| \quad for\ -\pi \prec x \prec \pi$$

Solution

$$a_0 = \frac{1}{\sqrt{2\pi}} \int_{-\pi}^{\pi} |x|\,dx = \frac{2}{\sqrt{2\pi}} \int_{0}^{\pi} x\,dx = \frac{\sqrt{2}\pi^2}{2\sqrt{\pi}}$$

$$= \frac{2}{\sqrt{\pi}k^2}[(-1)^k - 1] \quad \text{for each} \quad k=0,1,2,\ldots,n$$

$$a_{k+n} = \frac{1}{\sqrt{\pi}} \int_{-\pi}^{\pi} |x|\ \sin(kx)dx$$

= 0 for each k=0,1,2,,n-1

$$a_{k+n} = \frac{1}{\sqrt{\pi}} \int_{-\pi}^{\pi} |x|\ \sin(kx)dx$$

= 0 for each k=0,1,2,…,n-1

Example 4(b) Find the trigonometric polynomial to approximate

$\cos(-\pi)$

Solution The solution in Fig.(6)

4.1 Matlab Program for Trigonometric functions

```
% This the Method for approximation by the
 % trigonometric function in the file
 % trigonometric_polynomail_app.m
 % command windows:-
 % >> syms x;
 % >> f=cos(x);
 % >> n=6;
 % >> xo=-pi;
 % >> fnx=trigonometric_polynomail_app(f,n,xo)

function fnx=trigonometric_polynomail_app(f,n,xo)
syms x;
q11=(1/sqrt(pi));
q0=1/sqrt(2*pi);
for i=1:n
   q1(i)=q11*cos(i*x);
end
for j=1:n-1
   q2(j)=q11*sin(j*x);
end
   d=[q0 q1 q2]
   d1=f*d
   k=inline(f*d)
   a=int(k(x),x,-pi,pi)
   fn=a.*d
   g=inline(fn);
   g1=feval(g,xo);
   fnx=sum(g1);
```

```
>> syms x;
>> f=cos(x);
>> n=6;
>> xo=-pi;
>> fnx=trigonometric_polynomail_app(f,n,xo);

fnx =   -1.0000
```

Fig.(6) approximation by the trigonometric functions using file trigonometric _polynomail_app.m

V. LEGENDRE POLYNOMIALS

Legendre functions are solutions to **Legendre's differential equation**:

$$\frac{d}{dx}\left[(1-x^2)\frac{d}{dx}P_n(x)\right]+n(n+1)P_n(x)=0 \quad (12)$$

These solutions for $n = 0, 1, 2, \ldots$ (with the normalization $P_n(1) = 1$) form a polynomial sequence of orthogonal polynomials called the **Legendre polynomials**. Each Legendre polynomial $P_n(x)$ is an nth-degree polynomial. It may be expressed using **Rodrigues' formula**

$$P_n(x)=\frac{1}{2^n n!}\frac{d^n}{dx^n}[(x^2-1)^n] \quad (13)$$

That these polynomials satisfy the Legendre differential equation **(12)** follows by differentiating (n+1) times both sides of the identity

$$(x^2-1)\frac{d}{dx}(x^2-1)^n=2nx(x^2-1)^n \quad (14)$$

and employing the general Leibniz rule for repeated differentiation.
The P_n can also be defined as the coefficients in a Taylor series expansion:

$$\frac{1}{\sqrt{1-2xt+t^2}}=\sum_{n=0}^{\infty}P_n(x)t^n \quad (15)$$

In physics, this generating function is the basis for multiple expansions

A. 5.1 Recursive Definition

Expanding the Taylor series in equation (15) for the first two terms gives

$$P_0(x)=1 \quad , \quad P_1(x)=x$$

for the first two Legendre Polynomials. To obtain further terms without resorting to direct expansion of the Taylor series, equation (15) is differentiated with respect to t on both sides and rearranged to obtain

$$\frac{x-t}{\sqrt{1-2xt+t^2}}=(1-2xt+t^2)\sum_{n=1}^{\infty}nP_n(x)t^{n-1} \quad (15')$$

Replacing the quotient of the square root with its definition in (15), and equating the coefficients of powers of t in the resulting expansion gives ***Bonnet's recursion formula***

$$(n+1)P_{n+1}(x)=(2n+1)xP_n(x)-nP_{n-1}(x) \quad (16)$$

This relation, along with the first two polynomials P_0 and P_1, allows the Legendre Polynomials to be generated recursively and can found using the program in Fig.(7)

5.2 Matlab Program for Legendre polynomials

```
% The Legendre polynomials are a basis for the set of polynomials,
% on the interval [-1,1].and the first few Legendre polynomials are:
% P0(x) = 1
% P1(x) = x
% P2(x) = ( 3*x^2 - 1 ) / 2
% P3(x) = ( 5*x^3 - 3*x ) / 2
% P4(x) = ( 35*x^4 - 30*x^2 + 3 ) / 8
% command windows:-
% >> c=rlegendre(0)
% >> c=rlegendre(2)
% >> c=rlegendre(4)

function c= rlegendre(k)
if k==0
  c = 1;
elseif k==1
  c = [1 0];
else

  c = ((2*k-1)*[rlegendre(k-1),0] -
(k-1)*[0,0,rlegendre(k-2)])/k;
end
```

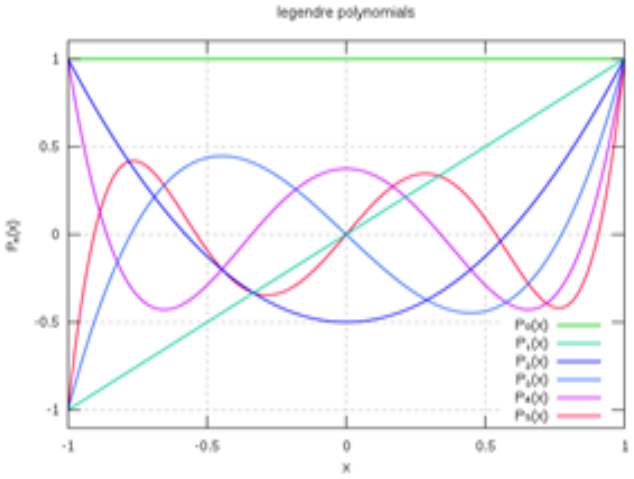
```
>> c=rlegendre(0)
   c =  1
 >> c=rlegendre(2)
    c =   1.5000    0   -0.5000
  >>  c=rlegendre(4)
 c =   4.3750   0   -3.7500   0
0.3750
```

Fig.(7) Legendre Polynomials to be generated recursively

The first few Legendre polynomials are:

n	$P_n(x)$
0	1
1	x
2	$\frac{1}{2}(3x^2-1)$
3	$\frac{1}{2}(5x^3-3x)$
4	$\frac{1}{8}(35x^4-30x^2+3)$
5	$\frac{1}{8}(63x^5-70x^3+15x)$
6	$\frac{1}{16}(231x^6-315x^4+105x^2-5)$

The graphs of **these polynomials (up to *n* = 5) are shown below:**



A. 5.3 Orthogonality

An important property of the Legendre polynomials is that they are orthogonal with respect to the L^2 inner product on the interval $-1 \le x \le 1$

$$\int_{-1}^{1} P_m(x) P_n(x) dx = \frac{2}{2n+1}\delta_{mn} \quad (17)$$

(where δ_{mn} denotes the Kronecker delta, equal to 1 if $m = n$ and to 0 otherwise).

Example 5 Prove the Legendre polynomials are orthogonal with respect to the weight function w(x)=1 on the interval $-1 \le x \le 1$. That is

$$\int_{-1}^{1} P_m(x) P_n(x) dx = \frac{2}{2n+1}\delta_{mn} \quad (17)$$

(where δ_{mn} denotes the Kronecker delta, equal to 1 if $m = n$ and to 0 otherwise).

Solution First , We have to prove that

$$\int_{-1}^{1} x^r P_n(x)\, dx = 0 \text{ , for } r = 0(1)n-1 \quad (18)$$

Apart from a constant factor , this can be written

$$\int_{-1}^{1} x^r \frac{d^n}{dx^n}(x^2-1)^n dx = \underbrace{\left[x^r \frac{d^{n-1}}{dx^{n-1}}(x^2-1)^n \right]_{-1}^{+1}}_{0} - \int_{-1}^{1} r x^{r-1} \frac{d^{n-1}}{dx^{n-1}}(x^2-1)^n dx$$

$$= \ldots = (-1)^r \cdot r! \int_{-1}^{1} \frac{d^{n-r}}{dx^{n-r}}(x^2-1)^n\, dx = 0$$

If r=n , We obtain

$$(-1)^n\, n! \int_{-1}^{+1} (x^2-1)^n\, dx = 2n! \int_{0}^{1} (1-x^2)^n\, dx$$

$$= 2n! \int_{0}^{\pi/2} \cos^{2n+1}(\varphi)\, d\varphi$$

$$= 2n! \frac{2n(2n-2).....2}{(2n+1)(2n-1).....3}$$

$$= \frac{2^{2n+1}(n!)^3}{(2n+1)!}$$

Now , if n≠m We have $\int_{-1}^{1} P_m(x)\, P_n(x)\, dx = 0$

,since if m<n , then $P_m(x)$ is a polynomial of degree less than n and the integral vanishes in view of the relation for Eqn.(18) above . Finally, We also compute

$$\int_{-1}^{+1} [P_n(x)]^2\, dx$$

$$= \int_{-1}^{1} \left(\frac{1}{2^n n!} \frac{(2n)!}{n!} x^n + \ldots \right) \frac{1}{2^n n!} \frac{d^n}{dx^n}(x^2-1)^n dx$$

$$= \frac{(2n)!}{2^{2n}(n!)^3} \cdot \frac{2^{2n+1}(n!)^3}{(2n+1)!}$$

$$= \frac{2}{2n+1} \quad (19)$$

Since powers below x^n give no contributions to the integral. Hence We have the following relations

$$\int_{-1}^{1} x^r P_n(x)\, dx = 0, \quad for \quad r = 0(1)n-1 \quad (20)$$

$$\int_{-1}^{1} x^n P_n(x) dx = \frac{2^{n+1}(n!)^2}{(2n+1)!}, \quad for \quad r = n \quad (21)$$

$$\int_{-1}^{+1} [P_n(x)]^2\, dx = \frac{(2n)!}{2^{2n}(n!)^3} \cdot \frac{2^{2n+1}(n!)^3}{(2n+1)!} = \frac{2}{2n+1} \quad (19)$$

In fact, an alternative derivation of the Legendre polynomials is by carrying out the **Gram-Schmidt process** on the polynomials $\{1, x, x^2, ...\}$ with respect to the L^2 inner product on the interval $-1 \le x \le 1$.

Example 6(a)

Expand x^n as a series of Legendre polynomials on the interval $-1 \le x \le 1$.

Solution

Now Eqn.(8) gives

$$x^n = \sum_{j=0}^{n} a_j p_j(x) \qquad (22)$$

and Eqn.(9) when using Eqn.(19) gives

$$a_k = \frac{2k+1}{2} \int_{-1}^{+1} x^n \; p_k(x)\, dx \quad k=0(1)n \qquad (23)$$

to simplify the problem we take n=4 this implies

$$a_0 = \frac{1}{2} \int_{-1}^{+1} x^4 \; p_0(x)\, dx$$

$$= \frac{1}{2} \int_{-1}^{+1} x^4 \, dx = \frac{1}{2}\frac{2}{5} = \frac{1}{5}$$

$$a_1 = \frac{3}{2} \int_{-1}^{+1} x^4 \; p_1(x)\, dx$$

$$= \frac{3}{2} \int_{-1}^{+1} x^4 .x.\, dx = \frac{3}{2}\frac{x^6}{6}\Big|_{-1}^{+1} = 0$$

$$a_2 = \frac{5}{2} \int_{-1}^{+1} x^4 \; p_2(x)\, dx$$

$$= \frac{5}{2} \int_{-1}^{+1} x^4 \; \frac{3x^2-1}{2}\, dx$$

$$= \frac{5}{2} \frac{\frac{3x^7}{7} - \frac{x^5}{5}}{2}\Bigg|_{-1}^{+1} = \frac{5}{2}\frac{8}{35} = \frac{20}{35}$$

$$a_3 = \frac{7}{2} \int_{-1}^{+1} x^4 \; p_3(x)\, dx = \frac{7}{2} \int_{-1}^{+1} x^4 \; \frac{5x^3-3x}{2}\, dx = 0$$

and Eqn.(6.19) gives

$$a_4 = \frac{9}{2}\frac{2^5(4!)^2}{9!} = \frac{9}{2}\frac{16}{315} = \frac{8}{35}$$

$$\therefore x^4 = \frac{8}{35} p_4(x) + \frac{20}{35} p_2(x) + \frac{7}{35} p_0(x) \qquad (24)$$

Put x=1 then x^4 is approximated to 1 ,exactly, because Legendre polynomials are orthonormal , i.e. $p_n(1) = 0$

Alternatively instead of the derivation for a_k's x^4 can be expressed exactly when using ,recursively , the definition of Legendre polynomials as

$$\therefore x^4 = \frac{1}{35}[8p_4(x) + 20p_2(x) + 7p_0(x)] \qquad (24)$$

Eqn.(24) shows that any powers of x can be similarly expressed as a series of Legendre polynomials exactly .

5.4 Matlab Program to Express a function f(x) as a series of Legendre polynomials

Example 6(b)

Write a Matlab program to express x^4 as a series of Legendre polynomials exactly and evaluate it at x_0=1. The program in Fig.(8) is the file Legendre_polynomail_app.m with the command window that shows the value of x^4 at x_0=1 is equal to one.

Example 6(c)

Expand e^x as a series of Legendre polynomials on the interval $-1 \le x \le 1$.

Solution

Now Eqn.(8) gives

$$e^x \approx \sum_{j=0}^{n} a_j p_j(x) \qquad (25)$$

and Eqn.(9) when using Eqn.(19) gives

$$a_k = \frac{2n+1}{2} \int_{-1}^{+1} e^x \; p_k(x)\, dx \quad k=0(1)n \qquad (26)$$

Since , the integral in Eqn.(26) is difficult to be computed analytically and We can evaluated it using a Quadrature Formulae such as Simpson's Rule . To express exp(x) as a Legendre Series by applying the File Legendre_polynomail_app.m in Fig.(8) , unfortunately the program does not work because We really need a quadrature formula instead of INT Built-in Matlab function .

VI. Chebyshev polynomials

Chebyshev polynomials, named after Pafnuty Chebyshev[1] are a sequence of orthogonal

```
% This the Method for approximation by the
% Lgendre polynomial in the file
% Legendre_polynomail_app.m
% command windows:-
% >> syms x;    % >> f=x^4;   % >> n=4;
% >> xo=1;
% >> fnx=Legendre_polynomail_app(f,n,xo)
function fnx=Legendre_polynomail_app(f,n,xo)
syms x;
for i=0:n
    C=rlegendre(i)
    k=(2*i+1)/2;
    p(i+1)=poly2sym(C)
    q(i+1)=k*poly2sym(C)
end
d=f*p
 d1=f*q
 k=inline(d1)
 a=int(k(x),x,-1,1)
 fn=a.*d
g=inline(fn);
g1=feval(g,xo);
 fnx=sum(g1);
% The Legendre polyonomials are a basis for the
set of polynomials,
% on the interval [-1,1].and the first few Legendre
polynomials are:
% P0(x) = 1
% P1(x) = x
% P2(x) = ( 3*x^2 - 1 ) / 2
% P3(x) = ( 5*x^3 - 3*x ) / 2
% P4(x) = ( 35*x^4 - 30*x^2 + 3 ) / 8
 %  command windows:-
 %  >> c=rlegendre(0)
 %  >> c=rlegendre(2)
 function c= rlegendre(k)
 if k==0    c = 1;
 elseif k==1   c = [1 0];
  else
  c = ((2*k-1)*[rlegendre(k-1),0] -
(k-1)*[0,0,rlegendre(k-2)])/k;
  end
```

```
>> syms x;
 >> f=x^4;
 >> n=4;
 >> xo=1;
 >>
fnx=Legendre_
polynomail_
app(f,n,xo)
   fnx = 1.0000
```

Fig.(8) File Legendre_polynomail_app.m with the command windows that shows the value of x^4 at x_0=1 is equal to one.

polynomials which are related to de Moivre's formula and which can be defined recursively. One usually distinguishes between **Chebyshev polynomials of the first kind** which are denoted T_n and **Chebyshev polynomials of the second kind** which are denoted U_n. The letter T is used because of the alternative transliterations of the name *Chebyshev* as *Tchebycheff* (French) or *Tschebyschow* (German).

The Chebyshev polynomials T_n or U_n are polynomials of degree n and the sequence of Chebyshev polynomials of either kind composes a polynomial sequence.

Chebyshev polynomials are important in **approximation theory** because the roots of the Chebyshev polynomials of the first kind, which are also called **Chebyshev nodes**, are used as nodes in **polynomial interpolation**. The resulting interpolation polynomial minimizes the **problem of Runge's phenomenon** and provides an approximation that is close to the polynomial of best approximation to a continuous function under the maximum norm. This approximation leads directly to the method of **Clenshaw–Curtis quadrature.** In the study of **differential equations** they arise as the solution to the **Chebyshev differential equations**

$$(1-x^2)y'' - xy' + n^2 y = 0 \quad (27)$$

and

$$(1-x^2)y'' - 3xy' + n(n+2)y = 0 \quad (28)$$

for the polynomials of the first and second kind, respectively.

A. 6.1 Recurrence relation

The **Chebyshev polynomials of the first kind** are defined by the recurrence relation

$$T_0(x) = 1$$
$$T_1(x) = 1$$
$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x) \quad (29)$$

The conventional **generating function** for T_n is

$$\sum_{n=0}^{\infty} T_n(x)t^n = \frac{1-tx}{1-2tx+t^2} \quad (30)$$

The exponential generating function is

$$\sum_{n=0}^{\infty} T_n(x)\frac{t^n}{n!} = \frac{1}{2}\left(e^{\left(x-\sqrt{x^2-1}\right)t} + e^{\left(x+\sqrt{x^2-1}\right)t}\right) \quad (31)$$

The generating function relevant for 2-dimensional **potential theory** and **multipole expansion** is

$$\sum_{n=0}^{\infty} T_n(x)\frac{t^n}{n!} = -\frac{1}{2}ln(1-2xt+t^2) \quad (32)$$

The **Chebyshev polynomials of the second kind** are defined by the **recurrence relation**

$$U(x) = 1$$
$$U_1(x) = 1$$
$$U_{n+1}(x) = 2xU_n(x) - U_{n-1}(x) \quad (33)$$

One example of a **generating function** for U_n is

$$\sum_{n=0}^{\infty} U_n(x)t^n = \frac{1}{1-2tx+t^2} \quad (34)$$

6.2 Trigonometric definition

The Chebyshev polynomials of the first kind can be defined compactly in an orbit through the trigonometric conjugac

$$T_n(x) = \cos\left(n\,arccos(x)\right)$$
$$= \cosh\left(n\,arccosh(x)\right) \quad (35)$$

Where

$$\mathbf{T_n(\cos(\theta) = \cos(n\theta)}$$

for n = 0, 1, 2, 3, ... which is a variant (equivalent transpose) of **Schröder's equation**, viz. $T_n(x)$ is functionally conjugate to nx, codified in the nesting property below. The polynomials of the second kind satisfy:

$$\mathbf{U_n(\cos(\theta) = \frac{\sin((n+1)\theta)}{\sin(\theta)}} \qquad \mathbf{(36)}$$

Evaluating the first two Chebyshev polynomials:

$$T_0(x) = cox(0\ x) = 1$$

and

$$T_1(cos(x)) = cos(x)$$

one can straightforwardly determine that:

$$\cos(2\theta) = 2\cos(\theta)\cos(\theta) - \cos(0\theta)$$

$$= 2\cos(\theta)^2 - 1$$

$$\cos(3\theta) = 2\cos(\theta)\cos(2\theta) - \cos(\theta)$$

$$= 4\cos(\theta)^3 - 3\cos(\theta)$$

and so forth.

$$T_n(T_m(x)) = T_{nm}(x) \qquad (37)$$

The first few Chebyshev polynomials

of the first kind are

$$T_0(x) = 1$$

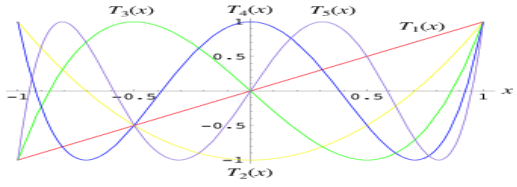
$$T_1(x) = x$$

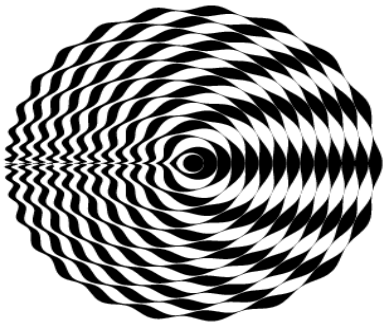
$$T_2(x) = 2x^2 - 1$$

$$T_3(x) = 4x^3 - 3x$$

$$T_4(x) = 8x^4 - 8x^2 + 1$$

The Graphs of these polynomials (up to n=5) are shown below





A beautiful plot can be obtained by plotting $T_n(x)$ radially, increasing the radius for each value of n , and filling in the areas between the curves (Trott 1999, pp. 10 and 84).

The Chebyshev polynomials of the first kind are defined through the identity

$$\mathbf{T_n(\cos(\theta)) = T_n(n\cos^{-1}(\cos(\theta))}$$

$$= \quad \mathbf{\cos(n\theta))} \qquad (38)$$

The Chebyshev polynomials of the first kind can be obtained from the generating functions

$$g_{1(t,x)} = \frac{1-t^2}{1-2xt+t^2}$$

and

$$g_{2(t,x)} = \frac{1-xt}{1-2xt+t^2} = \sum_{n=0}^{\infty} T_n(x)\,t^n$$

for $|x| \le 1$ and $|t| < 1$ (Beeler *et al.* 1972, Item 15). (A closely related generating function is the basis for the definition of Chebyshev polynomial of the second kind.)

A direct representation is given by

$$T_n(z) = \frac{1}{2}z^2\left[\left(\sqrt{1-\frac{1}{z^2}}+1\right)^n + \left(\sqrt{1-\frac{1}{z^2}}\right)^n\right]$$

The polynomials can also be defined in terms of the sums

$$T_n(x) = \frac{n}{2}\sum_{r=0}^{\lfloor n/2\rfloor}\frac{(-1)^r}{n-r}\binom{n-r}{r}(2x)^{n-2r}$$

$$= \mathbf{T_n(n\cos^{-1}x)}$$

$$= \sum_{m=0}^{\lfloor n/2\rfloor}\binom{n}{2m}(2x)^{n-2m}(x^2-1)^m$$

where $\binom{n}{k}$ is a binomial coefficient and $\lfloor x\rfloor$ is the floor function, or the product

$$T_n(x) = 2^{n-1}\prod_{k=1}^{n}\left\{x - cos\left[\frac{(2k-1)\pi}{2n}\right]\right\}$$

Zeros occur when

$$x = \cos\left[\frac{\pi\left(k-\frac{1}{2}\right)}{n}\right]$$

for $k = 1, 2, ..., n$. Extrema occur for

$$x = \cos\left(\frac{\pi k}{n}\right),$$

where k=0,…,n. At maximum, $T_n(x) = 1$, and at minimum,

$$T_n(x) = -1.$$

The Chebyshev polynomials are orthogonal polynomials with respect to the weighting function $(1-x^2)^{-1/2}$

$$\int_{-1}^{1} \frac{T_m(x)\,T_n(x)\,dx}{\sqrt{1-x^2}} = \begin{cases} \frac{1}{2}\pi\delta_{nm} & \text{for } m \neq 0, n \neq 0 \\ \pi & \text{for } m = n = 0, \end{cases}$$

where δ_{mn} is the Kronecker delta. Chebyshev polynomials of the first kind satisfy the additional discrete identity

$$\sum_{k=1}^{m} T_i(x_k)\,T_j(x_k) = \begin{cases} \frac{1}{2}\,m\,\delta_{ij} & \text{for } i \neq 0,\ j \neq 0 \\ m & \text{for } i = j = 0, \end{cases}$$

where x_k for k=1,…., m are the m zeros of $T_m(x)$ **and a**

Rodrigues representation

$$T_n(x) = \frac{(-1)^n \sqrt{\pi}\,\left(1-x^2\right)^{1/2}}{2^n\left(n-\frac{1}{2}\right)!}\,\frac{d^n}{dx^n}\left[\left(1-x^2\right)^{n-1/2}\right].$$

Example 8

Prove that Chebyshev polynomials are orthogonal polynomials with respect to the weighting function $(1-x^2)^{-\frac{1}{2}}$ and satisfies Eqn.(39)

$$\int_{-1}^{+1}\frac{T_m(x)T_n(x)}{\sqrt{1-x^2}}dx = \int_{0}^{\pi}\cos(m\theta)\cos(n\theta)d\theta$$
$$= \begin{cases} \frac{1}{2}\pi\ \delta_{nm} & for\ \ m \neq 0, n \neq 0 \\ \pi & for\ \ m = n = 0 \end{cases} \qquad (39)$$

where δ_{nm} is the Kronecker delta .

Solution Substitute x= cos(θ) into Eqn.(39) to get

$$\int_{-1}^{+1}\frac{T_m(x)T_n(x)}{\sqrt{1-x^2}}dx = \int_{0}^{\pi}\cos(m\theta)\cos(n\theta)d\theta$$
$$= \begin{cases} \frac{1}{2}\pi\ \delta_{nm} & for\ \ m \neq 0, n \neq 0 \\ \pi & for\ \ m = n = 0 \end{cases} \qquad (39)$$

where δ_{nm} is the Kronecker delta .

Example 9 Express the successive powers of x in terms of Chebyshev polynomials.

Solution We find

$1=T_0$

$x=T_1$

$x^2=\frac{1}{2}(T_0+T_2)$

$x^3=\frac{1}{4}(3T_1+T_3)$

$$x^4=\frac{1}{8}(3T_0+4T_2+T_4)$$

$$x^5=\frac{1}{16}(10T_1+5T_3+T_5)$$

$$x^6=\frac{1}{32}(10T_0+15T_2+6T_4+T_6)$$

$$x^7=\frac{1}{64}(35T_1+21T_3+7T_5+T_7)$$

and the general coefficient in the expansion $x^k = \sum c_n T_n$ can be proved to be

$$c_n = \frac{2}{\pi}\int_{-1}^{+1}\frac{x^k T_n(x)}{\sqrt{1-x^2}}dx = \frac{2}{\pi}\int_{0}^{\pi}\cos^k(\theta)\cos(n\theta)d\theta$$
$$= 2^{-k+1}\binom{k}{(n+k)/2} \qquad (40)$$

where n=1,3,5,….,k if k is odd and n=0,2,4,….,k if k is even

1) Example 10

Consider the Chebyshev expansion of $\log(1+x)$. One can express

$$\log(1+x) = \sum_{n=0}^{\infty} a_n T_n(x).$$

One can find the coefficients a_n either through the application of an *inner product* or by the discrete orthogonality condition. For the inner product,

$$\int_{-1}^{+1}\frac{T_m(x)\log(1+x)}{\sqrt{1-x^2}}dx = \sum_{n=0}^{\infty} a_n \int_{-1}^{+1}\frac{T_m(x)T_n(x)}{\sqrt{1-x^2}}dx,$$

which gives

$$a_n = \begin{cases} -\log(2) & : n = 0 \\ \frac{-2(-1)^n}{n} & : n > 0. \end{cases}$$

Alternatively, when you cannot evaluate the inner product of the function you are trying to approximate, the discrete orthogonality condition gives

$$a_n = \frac{2-\delta_{0n}}{N}\sum_{k=0}^{N-1} T_n(x_k)\log(1+x_k),$$

where δ_{ij} is the *Kronecker delta* function and the x_k are the

N Gauss–Lobatto zeros of $T_N(x)$

$$x_k = \cos\left(\frac{\pi\left(k+\frac{1}{2}\right)}{N}\right).$$

This allows us to compute the coefficients $a_n$very efficiently through the *discrete cosine transform*

$$a_n = \frac{2-\delta_{0n}}{N}\sum_{k=0}^{N-1}\cos\left(\frac{n\pi\left(k+\frac{1}{2}\right)}{N}\right)\log(1+x_k).$$

2) Example 11

Write a Matlab program to generate the sequence of Chebyshev polynomial. Express each ChebT as power of x with its coefficients.

Solution is in **Fig.(8(b))**.

Fig.(9) Matlab file ChebySumSeries1.m with its Command Window to evaluate

ln[(1+0.5)/(1-0.5)].

Fig.(10) Matlab file ChebySumSeries2.m with its Command Window to evaluate

ln[(1+0.75)/(1-0.75)].

Fig.(11) Matlab file echebser3with its Command Window to evaluate exp(x) and its three derivatives at x=0

```
function [ y0, y1, y2, y3 ] = echebser3 ( x, coef, nc )

%*****************************************************80
%% ECHEBSER3 evaluates a Chebyshev series and three derivative
%  Parameters:
%    Input, real X, the evaluation point.  -1 <= X <= +1.
%    Input, real COEF(NC), the Chebyshev series.
%    Input, integer NC, the number of terms in the series.  0 < NC.
%    Output, real Y0, the value of the Chebyshev series at X.
%    Output, real Y1, the value of the 1st derivative of the
%    Chebyshev series at X.
%    Output, real Y2, the value of the 2nd derivative of the
%    Chebyshev series at X.
%    Output, real Y3, the value of the 3rd derivative of the
%    Chebyshev series at X.
 b0 = coef(nc);
 b1 = 0.0;
 b2 = 0.0;
 c0 = coef(nc);
 c1 = 0.0;
 c2 = 0.0;
 d0 = coef(nc);
 d1 = 0.0;
 d2 = 0.0;
 e0 = coef(nc);
 e1 = 0.0;
 e2 = 0.0;

 x2 = 2.0 * x;
 for i = nc - 1 : -1 : 1
  b2 = b1;
  b1 = b0;
  b0 = coef(i) - b2 + x2 * b1;
  if ( 1 < i )
   c2 = c1;
   c1 = c0;
   c0 = b0 - c2 + x2 * c1;
  end
  if ( 2 < i )
   d2 = d1;
   d1 = d0;
   d0 = c0 - d2 + x2 * d1;
  end
  if ( 3 < i )
   e2 = e1;
   e1 = e0;
   e0 = d0 - e2 + x2 * e1; end
 end
 y0 = 0.5 * ( b0 - b2 );
 y1 = c0 - c2;
 y2 = ( d0 - d2 ) * 4.0;
 y3 = ( e0 - e2 ) * 24.0;
 return
```

```
>> format long
>> nc = 18;
>>  table5 = [ ...
   2.53213175550401667120, ...
   1.13031820798497005442, ...
   0.27149533953407656237, ...
   0.04433684984866380495, ...
   0.00547424044209373265, ...
   0.00054292631191394375, ...
   0.00004497732295429515, ...
   0.00000319843646240199, ...
   0.00000019921248066728, ...
   0.00000001103677172552, ...
   0.00000000055058960797, ...
   0.00000000002497956617, ...
   0.00000000000103915223, ...
   0.00000000000003991263, ...
   0.00000000000000142376, ...
   0.00000000000000004741, ...
   0.00000000000000000148, ...
   0.00000000000000000004 ];
>>  [ y0, y1, y2, y3 ] = echebser3 ( x,table5,nc)
      y0 =  1.000000000000000
      y1 = 1
      y2 =  1
       y3 = 1
```

Fig.(11) The Matlab file ECHEBSER3.m with its Command Widow evaluates a Chebyshev series and three derivative

```
x=chebfun('x');
format long
disp('Cheb coeffs of exp(x) :')
a=chebpoly(exp(x))'  ; a=a(end:-1:1)
```

Fig.(12) Matlab fileChebySumSeries2.m with its Command Window

```
function [ ChebyT,T ]= ChebT(n)
% Coefficients T of the nth Chebyshev
%  polynomial of the first kind.
% They are stored in the descending
% order of powers.
   t0 = 1;
   t1 = [1 0];
   if n == 0
      T = t0;
  elseif n == 1;
      T = t1;
 else
 for k=2:n
    T = [2*t1 0] - [0 0 t0];
    t0 = t1;
    t1 = T;
 end
     ChebyT=poly2sym(T);
 end
   1)
```

```
>> [ ChebyT,T ]= ChebT(3)

 ChebyT =    4*x^3-3*x
  T =   4     0    -3     0

>> [ ChebyT,T ]= ChebT(4)

 ChebyT =   8*x^4-8*x^2+1
    T =    8     0    -8     0     1
```

Fig.(8(b)) Matlab file ChebT.m with its command window shows the Coefficients of Chebshev polynomial and its powers of x.

```
function [sum]=ChebySumSeries1(n,x)
%  Direct Chebyshev Seris Summation for
 %ln[(1+x)/(1-x)] &  nc is number of terms
%executed .The exact ln[(1+0.5)/(1-0.5)]
 % valule is 1.098612288668110
for j=1:n
    coef(j)=4/(2*j-1);
 end
 sum=coef(1)*x;
for j=2:n
     [ChebyT,T ]= ChebT(2*j-1);
     sum=sum+polyval(T,x)*coef(j);
end
```

Fig.(9) Matlab file ChebySumSeries1.m with its Command Window

```
>> format long
>> syms x
>> x=0.5;
>> n=30;
>> [sum]=ChebySumSeries1(n,x)
  sum =1.098242634663975
```

```
function  [nc ,sum]=ChebySumSeris2(n,x)
% Direct Chebyshev Seris Summation for ln[(1+x)/(1-x)]
% nc is number of terms executed
% The exact value ln[(1+0.75)/(1-0.75)]= 1.945910149055313
 sum=0;
 sum1=0;
for j=1:n
    sum=sum+4*cos((2*j-1)*acos(x))/(2*j-1);
    disp( [j   sum])
    if abs(sum1-sum)<0.00005 , break , end
    sum1=sum;
    nc=j;
end
```

```
>> format long
>> n=200;
>> x=0.75;
>>  [nc ,sum]=ChebySumSeris2(n,x)

  nc =200
  sum = 1.946396172603678
```

Fig.(10) Matlab file ChebySumSeries2.m with its Command Window

I. Computational Remarks

Chebyshev Expansion for $\frac{\ln(1+x)}{\ln(1-x)}$ can been seen in [ ] & [ ] as

$$\ln[(1+x)/(1-x)]= \sum_{k=1}^{\infty} \frac{4}{2k-1} T_{2k-1}(x) \qquad (41)$$

while Chebyshev Expansion for exp(x) is given by

$$e^x = \sum_{n=0}^{\infty} a_n \; T_n(x) \qquad (42)$$

If one compares between the two coefficients in Eqn.(41) and Eqn.(42) , he can observe that each coefficient in Eqn.(41) followed by Chebyshev polynomial of odd degree and equals

$$a_k = \frac{4}{2k-1} \qquad k = 1,23,\ldots \qquad (43)$$

While the coefficients in Eqn.(42) can't be evaluated unless one uses a quadraure formula ((which is usually the general case for all three methods discussed)). The coefficients in Eqn.(42) are evaluated up n=18 or so as in Table5 by a matlab program uses CHEBFUN as in [5].

7.1 Clenshaw's algorithm. Let a polynomial $p \in P_n$ be given by a finite Chebyshev series (42) and let $x \in [-1, 1]$ be given. Show that p(x) can be evaluated by the following process. Set $u_{n+1} = 0$ and $u_n = a_n$ and

$$u_k = 2x\,u_{k+1} - u_{k+2} + a_k, \quad k = n-1, n-2, \ldots, 0. \quad (3.18)$$

Then $p(x) = \frac{1}{2}(a_0 + u_0 - u_2)$.

The program in Fig.(11) evaluates exp(x) efficiently for $x \in [-1,1]$ with its three derivatives . However , when We runs the same program to valuates $\frac{\ln(1+x)}{\ln(1-x)}$ for x=0.5 and x=0.75 We can't evaluates correctly for any few decimals places of accuracy . Alternatively , We run two different programs ChebySumSeries1.m and ChebySumSeries2.m as in Fig.(9) and Fig.(10) ,respectively. The accuracy evaluation is correct to four decimals and we are surprise WHY they don't compute to a very high decimal places as the Matlap Program in Fig.(11). *We conclude that Clenshaw's algorithm need to be adjusted to evaluate series of odd monomials and perhaps for even ones. Second ,our two programs needs only to investigate the evaluations of Chevbyshev Polynomial of degree n for large n and whether it oscillates or the sum does.*

ACKNOWLEDGEMENT

I would like to thank My Son , the Chemical Engineering ,Ahmed Dafalla for sending me the Matlab Software MATLAB R2013a and My Daughter , the Medical doctor , Shemia Dafalla to setup it together . Second , Lambert Academic Publishing , Germany for publishing , My book Numerical Methods and Algorithms using Matlab Programming Language[1], 2015.

REFERENCES

[1] Dafalla Awadalla Gismalla , Numerical Methods and Algorithms using Matlab Programming Language , LAMBERT Academic Publishing , https://www.lap-publishing.com/site/how-to-publish/14

[2] D. A. Gismalla , Chebyshev Approximation for COS (½ πX$^{4)}$ &SIN (½ πX$^{4)}$ Proceedings of the International Conference on Computing, pp.37, ICC 2010, INDIA

[3] D. A. Gismalla, MATLAB programs for some Numerical Methods and Algorithms, International Journal of Algorithms, Computing and Mathematics, Vol. 5 Number 1, pp.58, Feb. 2012

[4] D.A.Gismalla , Summation Method for Some Special Series Exactly , International Journal of Mathematics,Science,Technology and Management, (ISSN:2319-8125) Vol.1 Issue2

[5] The Internet
http://www.maths.ox.ac.uk/chebfun.
http://www.math.technion.ac.il/hat/
http://www.maths.ox.ac.uk/chebfun/ATAP

D.A.Gismalla, B.Sc.H. Mathematics ,Khartoum University ,1976,Sudan. M.Sc.in Computing 1982 ,Ph.D.in Numerical Analysis 1984 both Wales University, U.Kindom.I Worked at Gezira Sudan,Philadfia Jordan, Hadrmout Yamen and Taif &Tabouk Saudia Arabia Universities. D.A.Gismalla in I.J.C.M., & I.J.E.T.R &Member in N.Y.A.S