Mathematical Theory and Modeling www.iiste.org
ISSN 2224-5804 (Paper) ISSN 2225-0522 (Online)
Vol.5, No.6, 2015

Micro models For Numerical Analysis Topics Via Simi-Neural Net between Matlab & Mupad

Dr. Faisal Abdulateef Shaghati
Open Educational College Mathematics Department Iraq - Baghdad - Al adhamiya - near the Judicial Institute

e-mail: faisalshagati@yahoo.com

Abstract

The aim of this paper is to create micro models for specific topics in numerical analysis, when by certain types of combined non-linear equations including trigonometric and/or exponential functions look likes impassive against elaboration while to portrait or determine their roots, also it will offer somewhat modified approach for treatment of these as well as other cases via semi-neural net between MAT LAB & MUPAD, Moreover it includes certain comparison between the proposed approach and the classical ones, those separately depends on Newton-Raphson, False position, Secant, Muller or others methods, While these methods already embedded & act individually, dually, or triply through by built-in Mat lab &Mupad Functions designed to seeking for polynomials or other roots, also there is mutual verification between FTA theorem & the micro models results, except that the author tried to produce elegant & unique of its kind image cal models through by Mat lab integrating within Mupad to enhance his thesis.

Keywords: infinite roots, dancing roots, complex roots, real roots, polynomial's roots, algebraic roots, combined equationroots,Newton-Raphson,Falsi-Regula,Secant,Bisection,Muller classical approximation, Mat lab & Mupad approximation, FTA Theorem.

1. Introduction

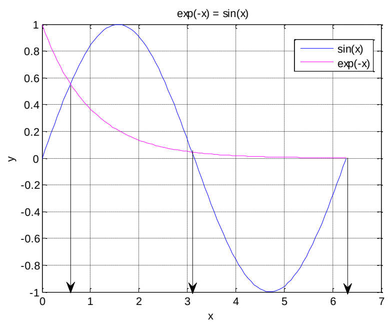
Since ancient Babylonian civilization, numerical analysis had been expressed itself through by constructing numerical approximation of $\sqrt{2}$ - tablet YBC 7289 - , that was to illustrate the length of the diagonal for a rectangular wall within 2.40 times 3.75 meter, a diagonal had to be 4.45 [1,3,18], recently due to the technological progress, one can use simple scientific calculator to get such approximated result up to thirteen decimal places i.e. 4.4522466238967, but it is to say that modern numerical analysis relies on the same concept that doesn't seek the exact answer for the similar problems and those related within them, at these cases it is going on exactly beside that to obtain a reasonable & accurate approximation while by it is impossible to get the exact one. Here it seems worthwhile to quote a great scientist at this field who is Herman H. Gold Stine in his prominent thesis " Remembrance Things Past" saying (Ptolemy developed a neat scheme for interpolation based on inequality of Archimedes that says if $A > B$ then $A / B > \sin A / \sin B$. He applied this to obtain the result : $(2/3)\sin(3/4) < \sin(1/2) < (4/3)\sin(3/4)$. This gave him sin (1/2) with a relative error of 2×10^{-6}. In addition to giving Ptolemy his table - i.e. a table of Sins with a half degree spacing-, this study gave us a whole way of viewing mathematics)[8].

There are many problems in mathematics which are so difficult, tedious or impossible to obtain their exact values as those of irregular areas, others naturally need to be approximated such as periodic numbers or so-called damp roots, the constant ratio as pi, e. Also it happens in frequent theoretical as well as practical cases of seeking roots for polynomials of higher degree or some transcendental functions which haven't anti-derivative, Against such problems, there are methods of approximations that belong to what Mathematicians called numerical analysis. From renaissance times and going on Newton & Leibnitz offered a huge developments for Calculus, When by their laws & methods strongly contribute to develop numerical formulas for physical problems, also (Napier 1614), Joseph- Louis Lagrange (1736- 1813), Leon Hard Euler(1707-1783) & Friedrich Gauss (1777- 1855) have their decidedly influence on the whole Mathematics & in between on the numerical analysis, While in logarithms & natural ones , numerical solutions of equations systems, Taylor series, created a new spectrum of mathematical theories, applications & inventions. Thereafter David Hilbert views were significance and strongly related to nature of mathematics while he proposed his theoretical thesis at the International Mathematics Conference at the preface of nineteen century in Paris, calling to transform mathematics to algorithms & to concentrate on applicable mathematics in the real world, but Gödel-Church-Post-Turing proved impassivity of such approaches in spite of algorithms beauties. The most modern development was achieved by Von Neumann coming along with appearance of first generation of IBM computers while by he saw that digital computers can be powerful mean to treat a large sets of specific mathematical problems as a large systems of equations & others, also it's possible to test David Hilbert idea whereas digital machines entirely depend on algorithms to implement their programmes & instructions. Then after the international second war, Von Neumann & Goldstein had offered prominent thesis that made solution of large systems of high order equations possible through by finding the inverse of high order matrices. Therefore at the closest times for our generation, (Ostrowski 1950) & (Motizkin 1954) offered their thesis about a number of algorithms required to solve polynomials, also (Stassen 1969) construct his algorithm for matrices multiplication, while (Wino grad 1969) found the required operation number for special functions solution. During the past half-Century, the growth in power & availability of digital computers has led to an increasing use of realistic mathematical models in science & engineering [4,10,2]. Therefore at this paper the author will make use of a super technical language through by integrating MAT LAB functions within MUPAD numeric functions for finding roots & technical graphics, also he will construct his -aimed to be elegant- micro models for specific impassive problems as well as other ones related to roots approximation for multiple topics of combined equations or polynomial ones.

2. Micro models for impassive numerical problems

In the computer age, high-quality languages & programmes have been appeared, the most significance & closer for Mathematicians interesting in computer applications was and still Mat Lab, it's well-known by its rich library effective and can be creation ally interacted in wide range of mathematical as well as technical tasks, so high and flexible accuracy of MAT LAB calculations as well as its graphical capacity allow one to create fantastic models in multidimensional space of mathematics & science, specifically if one like to integrate Mupad functions within MATLAB ones, it will create incomparable mathematical environment for problems solving & modeling, also one can find that so helpful in applied mathematical tasks that require quick mathematical problems solving as one will see at the coming lines, hereafter one would like to make use of these features to construct semi-neural models to treat what can be called impassive problems in numerical analysis, herein one will discuss an equation that has infinite numbers of roots, interestingly only three of its roots are distinct ones, the rest are multiple integers of pi, it's sin(x) - e^{-x} = 0, FRANCIS SCHEID [6] proposed graphical representation as an alternative for the numerical solving, then let's see if Mat lab programme can produce appropriate model that satisfies the proposition above, here below an illustration:

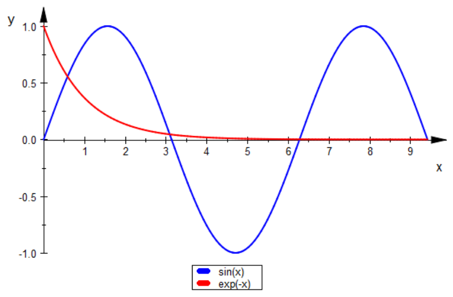
```
>>% roots illustration for combined equation
>> x = linspace (0,7); : Determines x-coordinate
>> g = exp(-x);          : Defines a function
>> h = sin(x);           : Defines 2nd function
>> plot(x, h)            : Draws a function
>> hold on               : Conserves the plane
>>plot(x ,g, 'm')        : Draws 2nd function
```



Moreover it is so easy to portrait the combined function above via micro model in Mupad as below:

% Micro model to portrait combined non-linear equations

```
plotfunc2d(sin(x),exp(-x),x = 0..3*PI):
```



Therefore it may be a kind of the scientist courtesy to lead his readers or audience in relax aiming at solving their problem quietly without stress, for that numerical analyst factually seek for the numeric roots value, but it's also serious & wonderful while it really offers a unique opportunity to see that, there are infinite roots, all of them are positive, the convergent decay of the exponential function explains that except the first root, all of them are integer multiple of pi, also one can gauss at least the initial value of these roots or their brackets, or determine the roots position, then it exhibits a full scope of the problem solution, fortunately one can obtain the wanted roots via strict micro model through by Mupad Numeric Function[16] as below:

% Micro model for solving combined non-linear algebraic equation

```
numeric::solve(exp(-x)-sin(x)=0,x=0..7, AllRealRoots)
```

```
{0.588532744, 3.096363932, 6.285049273}
```

These results can be compared in several ways, one of them that the easiest of all, it is to take them in comparison with the function portrait at the first model (not every portrait) while by it seems accurate to a large extent, secondly one can see if the results above are comparable to that obtained classically using quadratic or Muller method, usually it needs three starting points and gives several sequence approximations, at this case they are consequently 0.5885290, 0.5885327 for the first root, then 2.93, 3.06, 3.095, 3.096364 for the second root, therefore 6.285025, 6.285049 for the third root. Hereafter it is obvious that is serially clear similarity to the seven, five, six decimal places within the result of the micro model above.

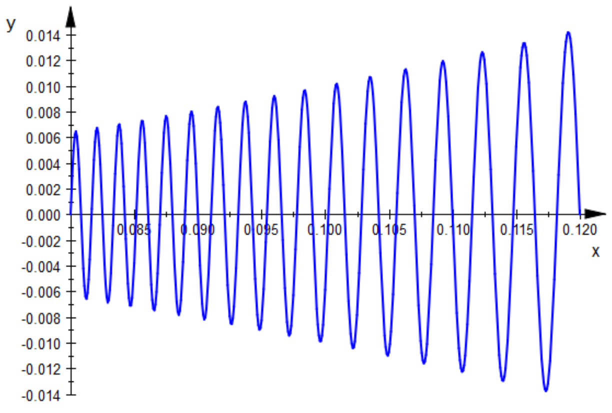
Taking in account that the micro model results can be obtained ad hoc and the classical quadratic ones through by many tedious steps of calculations, the first seems the best specially for practical purposes, in addition to that usually human calculations frequently meet with several mistakes while it is possible to construct micro models via Mat lab Functions integrating within Mupad ones which are distinguished by its so high accuracy.

Therefore let one detail within a specific problem that is considered by W.S. DORN [21] as one of the impassive problem in numerical analysis, it springs from the function below:

$$f(x)=\begin{cases} x^2 * \sin\left(\frac{792*pi}{100*x}\right) & x \neq 0 \\ 0 & x = 0 \end{cases}$$

It seems suitable to portrait this function through by micro model in Mupad Function to study its general behavior as well as the specific one through its intervals:
% Micro model to portrait an impassive function

```
plot(plot::Function2d(x^2*sin(792*PI/(100*x)),x= 0.08..0.12)):
```



Hence it seems better to search for the target roots in shorter intervals as it seen at the portrait above, one by one, then to construct successive micro models while by the macro one may don't offer a complete picture for the required roots, the most significance factor is the function target has various oscillation spaces across its intervals, then there are not the same numbers of roots at each interval, hence one can make use of the function model to suppress the intervals number as well as the roots number included at each interval.

It is reasonable to observe that initial point of the function portrait is 0.08 and not 0 as it seen, that is due to plane resolution and undistinguishable difference between these two roots, mathematically in accordance to the function formula, the root value can't be zero for the whole function will disappear going along the x-coordinate while its value will equal zero too.

Moreover the following micro models stand up as mathematical interpretation for the analytical approach above as below:

% Micro model via Mupad to suppress roots intervals of the function/equation target

```
numeric::realroots(x^2*sin((792*6.28)/(100*x))=0,x=0.080..0.12,   Merge = FALSE)
```

```
[[0.08, 0.085], [0.085, 0.09], [0.09, 0.095], [0.095, 0.1], [0.1, 0.105], [0.105, 0.11], [0.11, 0.115], [0.115, 0.12]]
```

Naturally they are coming in accordance with the function model , but one may go on to wider analysis for the assigned function behavior through by multiple micro models as it might be done, the following Mupad Function is so sensitive with respect to the float point, then it offers various output , the best one here can be obtained via the micro model below:

% Micro model for searching roots of the equation target

```
    numeric::realroots(x^2*sin((792*PI)/(100*x))=0,x= 0.08..0.12,10^(-4) )
```

```
[[0.08, 0.080078125], [0.08078125, 0.080859375], [0.081640625, 0.08171875], [0.082421875, 0.082578125], [0.083359375, 0.0834375],
  [0.08421875, 0.084296875], [0.08515625, 0.085234375], [0.086015625, 0.08609375], [0.08703125, 0.087109375], [0.08796875, 0.088046875],
  [0.088984375, 0.0890625], [0.089921875, 0.090078125], [0.091015625, 0.09109375], [0.09203125, 0.092109375], [0.093125, 0.093203125],
  [0.09421875, 0.094296875], [0.095390625, 0.09546875], [0.0965625, 0.096640625], [0.097734375, 0.0978125], [0.098984375, 0.0990625],
  [0.100234375, 0.1003125], [0.101484375, 0.1015625], [0.1028125, 0.102890625], [0.104140625, 0.10421875], [0.105546875, 0.105625],
  [0.106953125, 0.10703125], [0.1084375, 0.108515625], [0.109921875, 0.110078125], [0.111484375, 0.1115625], [0.113125, 0.113203125],
  [0.114765625, 0.11484375], [0.11640625, 0.116484375], [0.118203125, 0.11828125], [0.119921875, 0.12]]
```

There are thirty four intervals at the output above, where is one can find the required roots, where are they sitting?, factually it is impossible to gauss the roots position on base of knowing only their numbers as well as the intervals that include them, then what can one do?! one may think that the best way is to investigate - here the function model is useful- each of those intervals, then to elaborate them one by one aiming at knowing the number as well as the position of the required roots at every interval, therefore one can select the proper interval to seek for any root via specific micro model using a suitable Mupad Numeric Function.

So interesting that roots number equals the intervals numbers resulted through by the first micro model, that means at every interval there is only one root. most interesting that the difference between every two successive roots belong to the closed interval[0.0006,0.0018], although this value is so small one but it is decider factor which root is the most accurate & best choice as zero value for the function under investigation, then hence the intervals refer to existence of thirty four roots, one can conclude that these thirty four roots can be considered as appropriate approximations. Also interesting - with respect to the plane resolution & the function portrait, these roots already appear as the ones whereas each of them has his accurate position at the plane coordinates.

Still so significance matter, that is to determine the numerical values of the function roots, the micro model above enable us to explain the general behavior of the function as well as its roots, thus one can describe them in culture language as 'dancing roots', but one would like to construct a set of micro models that allow one to know their behavior as well as values at their certain specific intervals, hereafter it is coming along as follows :

% Micro model via Mupad to obtain the function roots at the interval [0.08 0.09]

```
  numeric::solve(x^2*sin((792*PI)/(100*x))=0,x= 0.08..0.09, AllRealRoots)
```

```
{0.08, 0.08081632653, 0.08164948454, 0.0825, 0.08336842105, 0.08425531915, 0.08516129032, 0.08608695652, 0.08703296703, 0.088,
  0.08898876404, 0.09}
```

% Micro model via Mupad to obtain the function roots at the interval [0.09 0.10]

```
numeric::solve(x^2*sin((792*PI)/(100*x))=0,x= 0.09..0.10, AllRealRoots)
```

```
{0.09, 0.09103448276, 0.09209302326, 0.09317647059, 0.09428571429, 0.09542168675, 0.09658536585, 0.09777777778, 0.099}
```

% Micro model via Mupad to obtain the function roots at [0.10 0.11]

```
   numeric::solve(x^2*sin((792*PI)/(100*x))=0,x= 0.10..0.11, AllRealRoots)
```

```
{0.1002531646, 0.1015384615, 0.1028571429, 0.1042105263, 0.1056, 0.107027027, 0.1084931507, 0.11}
```

% Micro model via Mupad to obtain the function roots at [0.11 0.12]

```
numeric::solve(x^2*sin((792*PI)/(100*x))=0,x= 0.11..0.12, AllRealRoots)
```

```
{0.11, 0.1115492958, 0.1131428571, 0.1147826087, 0.1164705882, 0.1182089552, 0.12}
```

Therefore one has pretty & meaningful picture for the required roots, factually there are only four exact roots i.e. 0.08, 0.09, 0.11,0.12, the rest are approximated ones, the required thirty four roots are divided into four intervals above, it encourages one to obtain all roots of the function via the same Mupad Function, but it will fail to do the task, it offers only eight roots because it can't isolate the search intervals, the following micro model is interpretative one for this case:

% Micro model for clarifying a single case

```
numeric::solve(x^2*sin((792*PI)/(100*x))=0,x= 0.08..0.12, AllRealRoots:

Problem in isolating search intervals. Some roots might be lost[numeric::allRealRoots]

{0.08036532891, 0.08995437384, 0.09368029465, 0.09894981122, 0.104157696, 0.1069727689, 0.1130854985, 0.1155618233}
```

Therefore, it seemed that it is better to treat the problem in accordance to its portrait above, i.e. to search for the required roots at their labeled intervals, one by one via accordingly eight intervals & eight micro models as below, for matter of consistency one may like to determine these intervals via specific micro model once more orderly as the investigation going on, hence the following one:

```
% Micro model for roots intervals specifying

numeric::realroots(x^2*sin((792*PI)/(100*x))=0,x= 0.080..0.12,  Merge = FALSE)
 [[0.08, 0.085], [0.085, 0.09], [0.09, 0.095], [0.095, 0.1], [0.1, 0.105], [0.105, 0.11], [0.11, 0.115], [0.115, 0.12]]
```

Hereafter eight well ordered micro models to explore the required roots:

```
% Micro model 1 to find the roots at the interval [0.08, 0.085]
numeric::solve(x^2*sin((792*PI)/(100*x))=0,x= 0.080..0.085, AllRealRoots)
 {0.08, 0.08081632653, 0.08164948454, 0.0825, 0.08336842105, 0.08425531915}
% Micro model 2 to find the roots at the interval [0.085, 0.09]
numeric::solve(x^2*sin((792*PI)/(100*x))=0,x= 0.085..0.09, AllRealRoots)
 {0.08516129032, 0.08608695652, 0.08703296703, 0.088, 0.08898876404, 0.09}

% Micro model  3 to find the roots at the interval [0.09, 0.095]

numeric::solve(x^2*sin((792*PI)/(100*x))=0,x= 0.09..0.095, AllRealRoots)
 {0.09, 0.09103448276, 0.09209302326, 0.09317647059, 0.09428571429}
% Micro model 4 to find the roots at the interval [0.095,  0.1]

numeric::solve(x^2*sin((792*PI)/(100*x))=0,x= 0.095..0.1, AllRealRoots)
 {0.09542168675, 0.09658536585, 0.09777777778, 0.099}
% Micro model 5 to find the roots at the interval [0.1,  0.105]

numeric::solve(x^2*sin((792*PI)/(100*x))=0,x= 0.1..0.105, AllRealRoots)
 {0.1002531646, 0.1015384615, 0.1028571429, 0.1042105263}
 % Micro model 6 to find the roots at the interval [0.105,  0.11]

numeric::solve(x^2*sin((792*PI)/(100*x))=0,x= 0.105..0.11, AllRealRoots)
  {0.1056, 0.107027027, 0.1084931507, 0.11}
% Micro model 7 to find the roots at the interval [0.11, 0.115]

numeric::solve(x^2*sin((792*PI)/(100*x))=0,x= 0.11..0.115, AllRealRoots)
 {0.11, 0.1115492958, 0.1131428571, 0.1147826087}

% Micro model 8 to find the roots at the interval [0.115,  0.12]

numeric::solve(x^2*sin((792*PI)/(100*x))=0,x= 0.115..0.12, AllRealRoots)
 {0.1164705882, 0.1182089552, 0.12}
```

These roots as it easily visible are distributed asymmetrically & orderly as 6, 6, 5, 4, 4, 4, 4, 3 roots across their consequent intervals. While they are thirty four ones as it can be counted at the function portrait, there are two repeated ones, the first (0.09) is situated at the first place of the third micro model output & the second (0.11) is sitting at the first place of the seventh micro model output.

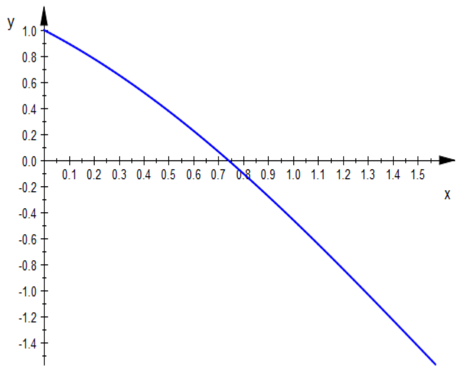
Therefore metaphorically speaking, these roots are dancing in various styles at their intervals, as it seems through by their numerical spacing as follows:

1st	0.0008	0.0006	0.0009	0.0008	0.0009	
2nd	0.0009	0.0010	0. 0010	0.0009	0.0010	0.0011
3rd	0.0010	0. 0010	0. 0011	0.0011	0.0010	
4th	0.0011	0.0012	0.0013			
5th	0.0013	0. 0013	0.0014			

6th	0.0014	0. 0014	0.0016
7th	0.0015	0.0016	0. 0016
8th	0.0018	0. 0018	

Hereafter an interesting case for reasonable comparison between Mupad results & those of Newton-Raphson, Secant, Regula Falsi methods for non-linear equation, the firsts ones are obtained via micro models as follows :

```
plot(plot::Function2d(cos(x)-x, x = 0..PI/2)):
```



```
numeric::realroot (cos(x)-x=0, x= 0..PI/2

0.7390851332

numeric::solve (cos(x)-x=0, x= 0..PI/2 )
{0.7390851332}

numeric::fsolve(cos(x)-x=0,x= 0..PI/2 )
[x = 0.7390851332]
```

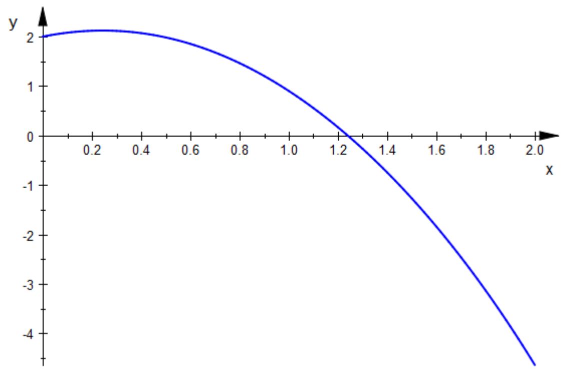
Evidently the results obtained via the classical methods are the same but by different usually tedious calculation steps that orderly are five, six, seven ones[18], this is against one step or at most two steps within portrait model via Mupad micro models.

There are many methods for roots approximation, the most widely applied are Newton- Raphson , False position or Regula falsi & Secant methods, they are regularly studied across the university mathematics curriculums as well as at a wide range of numerical analysis references [18,5,9,11,12,20], these methods are integrated one within other, whereas Secant one is used when by Newton one fails, while Regula Falsi is considered as somewhat modified one of the Secant method, these & other classical approximation methods are of highest significance at all levels, pure mathematical, technical as well as educational level, also they are extremely significance from miniaturization or say information science viewpoint, while by all micro soft packages as well as computer Functions used for roots approximations are based on algorithms of those approximations methods at all, although there are several computer programmes & languages that enhance mathematical approximation, but it isn't paradoxically that MAT LAB/MUPAD Functions for roots approximation are considered as the best ones, that is not only because it is little time & effort consuming but it offers the most accurate results.

At this paragraph one would like to introduce micro models approach for non-linear combined equation that includes trigonometric, exponential as well as algebraic nomials via Mupad as well as Mat Lab micro models, the firsts for approximation while the seconds for both approximation as well as verification, also one will come across certain comparison between their micro models.

The first micro model is coming to illustrate the root finding process for 2x+3cos(x)-e^{x} = 0 at the interval $0 \leq x \leq 2$ as follows:

```
% Mupad micro model to portrait non- linear combined equation
    plot(plot::Function2d(2*x+3*cos(x)-exp(x),x = 0..2)):
```



The second step is to find an approximation value for the assigned root:

```
% Micro model via Mupad Function numeric::solve to find the root value
  numeric::solve(2*x+3*cos(x)-exp(x)=0,x= 0..2 )
     {1.239714698}
```

Also one may like to elect alternative micro model for the same purpose:

```
 %Micro model via Mupad Function numeric::realroot to find the root value
numeric::realroot(2*x+3*cos(x)-exp(x)=0,x= 0..2 )
     1.239714698
```

In addition to that one can create another micro model to get the same value

```
% Micro model via Mupad Function numeric::fsolve to achieve the same task
    numeric::fsolve(2*x+3*cos(x)-exp(x)=0,x= 0..2 )
    [x = 1.239714698]
```

Therefore , factually one of these micro models is completely enough, but it is preferable to have many ways going to a desired aim.

Moreover, to taste & test Mat Lab Function for roots finding[14] , let's construct a specific micro model for the same polynomial as follows :

```
>> % Micro model in Mat Lab to do the task
>> fun = @(x)2*x + 3*cos(x) - exp(x);: Define anonymous function
>> x   = fzero(fun,1)                : Find the function root
    x = 1.23971469797522
```

Thus one is here, the same result of Mupad micro models but up to fourteen decimal places. After all, it seems the time to compare the obtained results through by direct substitution in the original equations via micro model in Mat Lab :

```
>> % Mat Lab micro model for roots verification
>> s = [1.239714698     1.23971469797522]; : Puts  two roots in a vector
>> for n = 1:2                             : Makes a loop
    x = s(n)                               : Determines a root value
   f = 2*x + 3*cos(x) - exp(x)             : Evaluates the function value
   end                                     : Closes the loop

x = 1.239714698,         f =-1.0635936575909e-10
x =1.23971469797522,  f = -1.11022302462516e-14
```

Therefore it seems that both results of MATLAB & Mupad are closed to the zero value of the target equation within assigned difference.

3. Polynomials roots

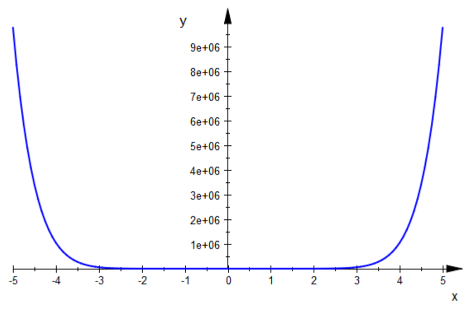
Herein one has to deal within finite non- linear algebraic equations of order or degree equals n, the general form of these polynomials is:

$f(x) = a_1x^n + a_2x^{n-1} + a_3x^{n-2} + \dots + a_{n-2}x^3 + a_{n-1}x^2 + a_n x + a_{n+1}$.

Then one would like to detail within appropriate micro models to treat the problem of roots finding for these polynomials via Mupad Numeric Functions, going also to certain comparison within the classical methods. The first micro model is coming to treat a problem of finding roots for the polynomial equation $x^{10} -1 = 0$:

% Micro model to portrait a polynomial

```
plot(x^10-1),1..-1:
```



The micro model is portrayed this way for otherwise the plot(plot::Function2d… command[17] ignore the complex points to yield an image look likes L'Escarpolette of Fragonard in the lower half of a real plane, then to find the required roots :
% Micro model to find roots of a polynomial

```
numeric::solve((x^10-1=0,-1..1),AllRealRoots)
```

```
{−1.0, 1.0}
```

while finding one real root for this polynomial through by Newton-Raphson method in its classical utilization required forty two steps of tedious calculations, also frequently in engineering practice , one need to find real as well as complex roots, then the following Mupad micro models achieve the task which can't be done through by Newton-Raphson or Bisection method[19].

Before going to details, mathematically there is certain verifications one through by one between micro models results & two significance corollaries of Fundamental Theorem of Algebra as follows [13],

1st: Every polynomial of nth degree has only n roots.
2nd: In an polynomial within real coefficient, non-real complex roots occur in conjugate pairs.

% Micro model to find real & complex roots of a polynomial

```
numeric::polyroots(poly(x^10-1,[x]))
```

```
[1.0, 0.8090169944 + 0.5877852523 i, 0.8090169944 − 0.5877852523 i, 0.3090169944 + 0.9510565163 i, 0.3090169944 − 0.9510565163 i, −0.3090169944 + 0.9510565163 i,
  −0.3090169944 − 0.9510565163 i, −0.8090169944 + 0.5877852523 i, −0.8090169944 − 0.5877852523 i, −1.0]
```

Then, there are several alternative micro models to do the task:

% Micro model (2) to find real & complex roots of a polynomial

```
numeric::polyroots(x^10 - 1)
```

```
[1.0, 0.8090169944 + 0.5877852523 i, 0.8090169944 − 0.5877852523 i, 0.3090169944 + 0.9510565163 i, 0.3090169944 − 0.9510565163 i, −0.3090169944 + 0.9510565163 i,
  −0.3090169944 − 0.9510565163 i, −0.8090169944 + 0.5877852523 i, −0.8090169944 − 0.5877852523 i, −1.0]
```

% Micro model (3) to find real & complex roots of a polynomial

```
numeric::solve((x^10-1=0,x=-1..1))
```

```
{−1.0, 1.0, 0.8090169944 − 0.5877852523 i, −0.3090169944 + 0.9510565163 i, −0.3090169944 − 0.9510565163 i, −0.8090169944 − 0.5877852523 i,
  0.8090169944 + 0.5877852523 i, 0.3090169944 + 0.9510565163 i, 0.3090169944 − 0.9510565163 i, −0.8090169944 + 0.5877852523 i}
```

Therefore all these micro models via integrating net between Mat Lab & Mupad can find its multiple applications in seeking for real & complex roots, in mathematics as well as in appropriate fields of science & engineering , while they can contribute in creation a fantastic environment for roots finding & offer several alternatives for roots approximation .

REFRENCES

1. Amjad Abbas Abu Jazar & Others, Numerical Analysis Methods Using Mat Lab, 2013
2. Berlent S.M & A. Alomar, Numerical Analysis, 2010
3. David Flower & Eleanor Robson, Square Root Approximation in Old Babylonian Mathematics,1998
4. Donald W. Peaceman, A personal Retrospection of Reservoir Simulation, 1990
5. Doron levy, Numerical Analysis,2005
6. FRANCIS SHIED, Numerical Analysis, 1990
7. Gerald & Wheatley, Numerical Analysis Methods, 1997
8. Herman H. Gold Stine, Remembrance of Things Past, 1990
9. Jeffery R. Chasnow, Introduction to Numerical Methods, 2012
10. Kendall. E . Atkinson, Numerical Analysis, 2001
11. .L. Ridgway Scott, Numerical Analysis, 2011
12. Mihail Konstantinov, Foundation of Numerical Analysis, 2007
13. N. P. Bali & Manish Goyal, Engineering Mathematics, 2011
14. PROFESSORS TEAM, fzero Function, MAT LAB 2014a. 2014
15. PROFESSORS TEAM, MAT LAB, The Technical Language of Computation 2014a, 2014
16. PROFESSORS TEAM, Mupad numeric Functions, MAT LAB 2014a,2014
17. PROFESSORS TEAM, Symbolic Math Toolbox, MuPAD, Graphics, MAT LAB 2014a, 2014
18. Richard L. Burden & J. Douglas Faires, Numerical Analysis, 2011
19. Steven C. Chapra, Applied Numerical Methods with MATLAB for engineers & scientists, 2012
20. Tood Young & Martin J. Mohlenkamp, Introduction to Numerical Methods & MAT LAB, 2105
21. WILLIAM S.DORN & Daniel D . McCracken, Numerical Methods, 1972