Numerical Methods

ProjectAssignment A: Accuracy of computation

March 31, 2023

Krzysztof Watras

Tutor's name: Jakub Wagner

Computer Science

Warsaw University of Technology,
Faculty of Electronics and Technology

I declare that this piece of work, which is the basis for recognition of achieving learning outcomes in the Numerical Methods course, was completed on my own.
Krzysztof Watras

Contents

Introduction

In this report I submit results of first assignment from ENUME course. This assignment focuses on the errors introduced during the numerical calculation due to inaccurate floating point representation standards. All tasks that are to be performed involve the same equation:

$$y = \frac{\cos(x)}{x^3} - x^2$$

Task 1 is focused on calculating the error introduced by the inaccurate storage of floating point numbers itself. This error will be noted as ϵ. This error will be different for different values of x, and so we will call it $T(x)$, to make clear that this property depends on x. Task 2 and task 3 focus on the error introduced by inaccurate representation of intermediate results. This error will be noted as η. Additionally, the tasks try to analyze, how different methods result in total error introduced to the final result. Here we will use $K_{a1}(x)$ and $K_{a2}(x)$ to differentiate it from error obtained in task 1 and from each other.

Throughout the document I will use following convention for two methods of calculating the results:

- **Analytical** when I am using the epsilon calculus to obtain results

- **Numerical** when I am using the method of simulation implemented as instructed in the assignment

This schema may be seen as slightly inaccurate, but it is consistent throughout the document, and it therefore considered best to avoid confusion and repetition

Mathematical Symbols and formulas used

$$\sigma[\tilde{v}^k] = T^k \cdot \sigma[\tilde{v}^{k-1}] + \eta^k \tag{1}$$

$$For\ \alpha \to 0, \sin(\alpha) \approx \alpha - \frac{\alpha^3}{6}, \cos(\alpha) \approx 1 - \frac{\alpha^2}{2} \tag{2}$$

$$\begin{gathered}(1+\epsilon_1)(1+\epsilon_2) \approx 1 + \epsilon_1 + \epsilon_2 \\ (1+\epsilon_1)^a \approx 1 + a\epsilon_1\end{gathered} \tag{3}$$

$$\begin{aligned} For\ \sigma[\tilde{y}] &\approx T_1\epsilon_1 + T_2\epsilon_2 + T_3\epsilon_3 + \dots \\ &+ K_1\eta_1 + K_2\eta_2 + K_3\eta_3 + \dots \\ T_x &= |T_1| + |T_2| + |T_3| + \dots \\ K_x &= |K_1| + |K_2| + |K_3| + \dots \end{aligned} \tag{4}$$

Properties (1,3,4) is introduced to us in the ENUME Lecture Notes[1]. Property (2) stems from the Tailor series of those functions[2].

Description of the numerical methods and algorithms

Inline Functions

In my calculations I use inline functions as they allow me to more easily express the intent. This is because I do not have to create separate files and perform linking of some sort in order to do computation. For general syntax of Inline functions you may refer to official MATLAB documentation[3].

Inline functions I use:

```
y = @(x) cos(x)./x.^3 - x.^2;
```

is just a MATLAB representation of the function we analyze in the report.

```
rand_sign_vec = @() (randi(2,1,5)-1.5)*2;
```

Is a custom function that returns a matrix of size 1x5 (so a vector length 5) of random numbers, either -1 or 1. Spread of both should be uniform as this uses builtin function ‘randi‘ that according to documentation[4] does exactly what one would expect it to. To clarify what is happening: first we choose number: 1 or 2, then we shift it to -0.5 or 0.5, then we scale it to -1 or 1.

Method of calculating the maximal error from random trials

```
tx_numerical = zeros(1, 100);
rand_sign_vec = @() (randi(2,1,5)-1.5)*2;
for i = 1:300
    en = eta * rand_sign_vec();
    v1 = T1 * (1+en(1));
    v2 = T1 * (1+en(2));
    v3 = T1 * (1+en(3));
    v4 = T2 * (1+en(4));
    v5 = (1+en(5));
```

```
        ytilde = v1+v2+v3+v4+v5;
        tx_numerical = max(tx_numerical, ytilde);
    end
```

This is the code that is responsible for calculations in tasks 2 (very similar in task 3). First, we initialize the array of 100 zeros, then we enter a loop and repeat 300 times:

1. Create an array of random numbers
2. Multiply the vector by eta
3. Multiply each component of the answer by the ± eta
4. Sum all components
5. Update max error table

Task 1

Given a function

$$y = \frac{\cos(x)}{x^3} - x^2$$

we want to calculate total error introduced by inaccurate representation of x.

Epsilon calculus

To calculate error of representation, everywhere where there is an x, we need to introduce ϵ as the error like so: $\tilde{x} = x(1+\epsilon)$ In our function we have:

$$\begin{aligned} v_1 &= \cos(x) \\ v_2 &= x^3 \\ v_3 &= \frac{v_1}{v_2} \\ v_4 &= x^2 \\ y &= v_3 - v_4 \end{aligned}$$

Therefore we obtain:

$$\begin{aligned} \tilde{v_1} &= \cos(x(1+\epsilon_1)) \\ \tilde{v_2} &= (x(1+\epsilon))^3 = x^3(1+3\epsilon_2) \\ \tilde{v_3} &= \frac{\tilde{v_1}}{\tilde{v_2}} \\ \tilde{v_4} &= (x(1+\epsilon))^2 = x^2(1+2\epsilon_3) \\ \tilde{y} &= \tilde{v_3} - \tilde{v_4} \end{aligned}$$

Let us focus on $\tilde{v_1}$:

$$\cos(x(1+\epsilon)) = \cos(x + x\epsilon) = \cos(x)\cos(x\epsilon) - \sin(x)\sin(x\epsilon)$$

We need to use properties of trigonometric functions in order to solve it. Those properties are listed in the section *Mathematical Symbols and formulas used*, formula (2).

Using those properties:

$$\begin{aligned}
\cos(x(1+\epsilon)) &= \cos(x)\cos(x\epsilon) - \sin(x)\sin(x\epsilon) \\
&= \cos(x)(1 - \frac{(x\epsilon)^2}{2}) - \sin(x)(x - \frac{(x\epsilon)^3}{6})\epsilon \\
&= \cos(x) - \sin(x)x\epsilon \\
&= \cos(x)\cdot(1 - \tan(x)x\epsilon)
\end{aligned}$$

Now solve v_3:

$$\begin{aligned}
\tilde{v_3} &= \frac{\cos(x)\cdot(1 - x\tan(x)\epsilon_1)}{x^3(1+3\epsilon_2)} \\
&= \frac{\cos(x)}{x^3}(1 - x\tan(x)\epsilon_1)\cdot(1+3\epsilon_2)^{-1} \\
&= \frac{\cos(x)}{x^3}(1 - x\tan(x)\epsilon_1)\cdot(1-3\epsilon_2) \\
&= \frac{\cos(x)}{x^3}(1 - x\tan(x)\epsilon_1 - 3\epsilon_2)
\end{aligned}$$

Substituting it to the equation:

$$\begin{aligned}
\tilde{y} &= \frac{\cos(x)}{x^3}(1 - x\tan(x)\epsilon_1 - 3\epsilon_2) - x^2(1+2\epsilon_3) \\
&= \frac{\cos(x)}{x^3} - \frac{\cos(x)}{x^3}(x\tan(x)\epsilon_1 + 3\epsilon_2) - x^2 - 2x^2\epsilon_3 \\
&= y - \frac{\cos(x)}{x^3}(x\tan(x)\epsilon_1 + 3\epsilon_2) - 2x^2\epsilon_3 \\
&= y(1 + (-\frac{\cos(x)}{x^3}(x\tan(x)\epsilon_1 + 3\epsilon_2) - 2x^2\epsilon_3)\frac{1}{y}) \\
&= y\left(1 + \left(-\frac{\sin(x)}{x^2}\epsilon_1 - 3\frac{\cos(x)}{x^3}\epsilon_2 - 2x^2\epsilon_3\right)\frac{1}{y}\right)
\end{aligned}$$

Therefore, using the formula (4):

$$T(x) = \frac{\left|-\frac{\sin(x)}{x^2}\right| + \left|-3\frac{\cos(x)}{x^3}\right| + \left|-2x^2\right|}{\frac{\cos(x)}{x^3} - x^2}$$

Numerical simulation

Function that we need to implement, described in an assignment:

$$T(x) = \frac{1}{\epsilon_{sim}}\left|\frac{y(\tilde{x}) - y(x)}{y(x)}\right|$$

is more general solution for calculating the error. It can be easily implemented in MATLAB code:

```
y = @(x) cos(x)./x.^3 - x.^2;
x_eps = x.*(1+esim);
yeps = y(x_eps);
```

where anonymous function y can be reused for calculating ”true” value of the function.

Comparison of results

Both methods do not differ significantly from one another in terms of results. This can be seen on 1, where the output difference is barely noticable. Therefore, the use of simpler solution should be recommended.

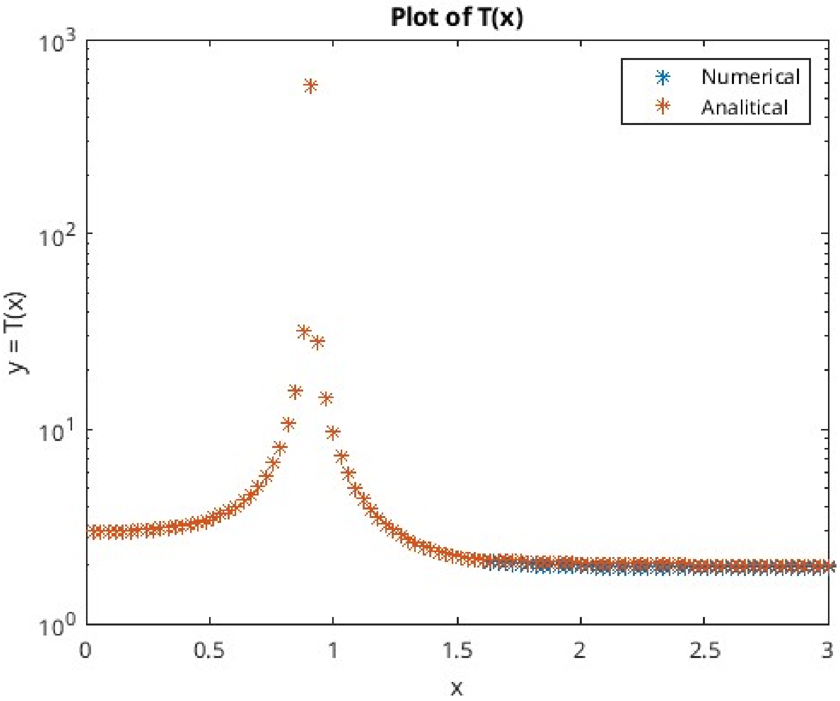

Figure 1: Comparison of calculating error via numerical and analytical way

Task 2

Conveniently, in task 2 we have the same function and same name for each v_n component as we did in Task 1. Therefore, we just need to change the part where we introduce errors:

$$\begin{aligned}
\tilde{v_1} &= \cos(x)(1+\eta_1) \\
\tilde{v_2} &= x^3(1+\eta_2) \\
\tilde{v_3} &= \frac{\tilde{v_1}}{\tilde{v_2}}(1+\eta_3) \\
\tilde{v_4} &= x^2(1+\eta_4) \\
\tilde{y} &= (\tilde{v_3}-\tilde{v_4})(1+\eta_5)
\end{aligned}$$

Epsilon calculus

Using the epsilon calculus rules:

$$\begin{aligned}
\tilde{y} &= \left(\frac{\cos(x)}{x^3}\frac{(1+\eta_1)}{(1+\eta_2)}(1+\eta_3) - x^2(1+\eta_4)\right)(1+\eta_5) \\
&= \left(\frac{\cos(x)}{x^3}(1+\eta_1-\eta_2+\eta_3) - x^2(1+\eta_4)\right)(1+\eta_5) \\
&= y\left(1+\frac{\cos(x)}{x^3}\frac{1}{y}(\eta_1-\eta_2+\eta_3) - x^2\frac{1}{y}\eta_4\right)(1+\eta_5) \\
&= y\left(1+\frac{\cos(x)}{x^3}\frac{1}{y}(\eta_1-\eta_2+\eta_3) - x^2\frac{1}{y}\eta_4+\eta_5\right)
\end{aligned}$$

Finally, we get:

$$K_{A1} = \frac{\left|\frac{\cos(x)}{x^3}\right| + \left|-\frac{\cos(x)}{x^3}\right| + \left|\frac{\cos(x)}{x^3}\right| + \left|-x^2\right|}{\left|\frac{\cos(x)}{x^3}-x^2\right|} + 1$$

Numerical simulation

In this exercise we simulate the error that we get due to rounding errors. Here, it is important to note that some errors may interfere such that they add up, or cancel out. Ideally we would try to calculate each permutation of $-\eta$ and $+\eta$, as that would yield true max error. However, in practice, it may take too long to calculate full coverage. Often, we may want to know just ”good enough” approximation of this error. Therefore, we try a number of randomly selected sample and calculate the error by updating the biggest error. Exact code solution can be seen in *Description of the numerical methods and algorithms* section.

Comparison of results

Similarly to results of task 1, both methods do not differ significantly from one another in terms of results. This can be seen on 2, where the output difference cannot even be seen (due to one point drawn on the other).

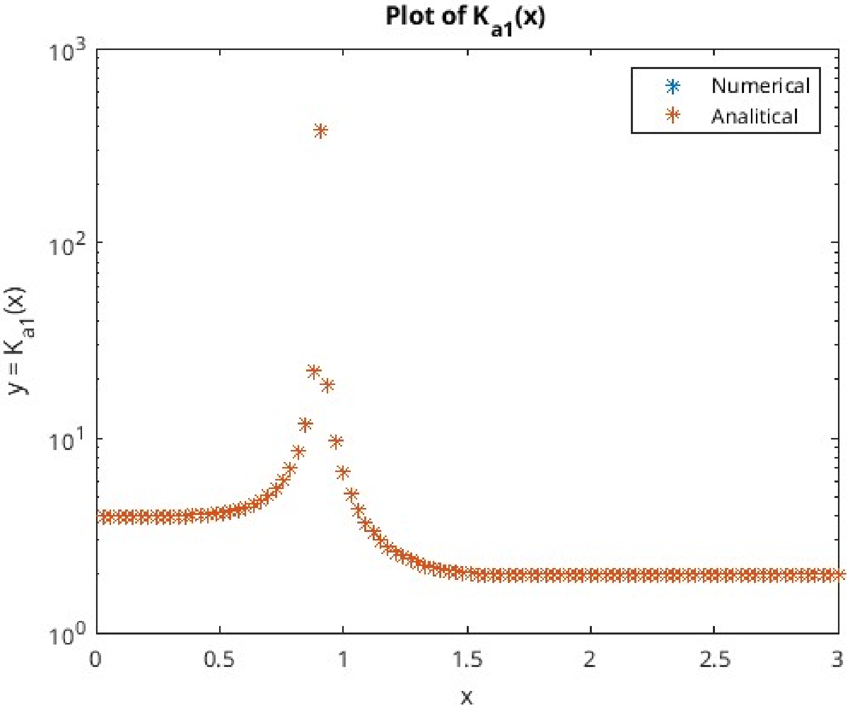

Figure 2: Comparison of calculating error via numerical and analytical way

Task 3

$$\begin{aligned}
v_1 &= \cos(x)\\
v_2 &= x^5\\
v_3 &= v_1 - v_2\\
v_4 &= x^3\\
y &= \frac{v_3}{v_4}
\end{aligned}$$

Therefore:

$$\begin{aligned}
\tilde{v_1} &= \cos(x)(1+\eta_1)\\
\tilde{v_2} &= x^5(1+\eta_2)\\
\tilde{v_3} &= (\tilde{v_3} - \tilde{v_4})(1+\eta_3)\\
\tilde{v_4} &= x^3(1+\eta_4)\\
\tilde{y} &= \frac{\tilde{v_3}}{\tilde{v_4}}(1+\eta_5)
\end{aligned}$$

Epsilon calculus

Using the epsilon calculus rules:

$$\begin{aligned}
\tilde{y} &= \left(\frac{\cos(x)(1+\eta_1) - x^5(1+\eta_2)(1+\eta_3)}{x^3(1+\eta_4)}\right)(1+\eta_5)\\
&= \frac{\cos(x)(1+\eta_1+\eta_3-\eta_4+\eta_5) - x^5(1+\eta_2+\eta_3-\eta_4+\eta_5)}{x^3}\\
&= \frac{\cos(x) + cos(x)(\eta_1+\eta_3-\eta_4+\eta_5) - x^5 - x^5(\eta_2+\eta_3-\eta_4+\eta_5)}{x^3}\\
&= y + \frac{cos(x)\eta_1 + cos(x)(\eta_3-\eta_4+\eta_5) - x^5\eta_2 - x^5(\eta_3-\eta_4+\eta_5)}{x^3}\\
&= y + \frac{cos(x)\eta_1 - x^5\eta_2 + (cos(x)-x^5)(\eta_3-\eta_4+\eta_5)}{x^3}\\
&= y\cdot\left(1 + \frac{cos(x)}{x^3}\frac{1}{y}\eta_1 - x^2\frac{1}{y}\eta_2 + \frac{cos(x)-x^5}{x^3}\frac{1}{y}(\eta_3-\eta_4+\eta_5)\right)
\end{aligned}$$

Finally, we get:

$$K_{A2} = \frac{\left|\frac{\cos(x)}{x^3}\right| + \left|-x^2\right| + \left|\frac{\cos(x)-x^5}{x^3}\right| + \left|-\frac{\cos(x)-x^5}{x^3}\right| + \left|\frac{\cos(x)-x^5}{x^3}\right|}{\left|\frac{\cos(x)}{x^3} - x^2\right|}$$

Numerical simulation

This part is identical to the corresponding one in Task 2. Of course the components of the equation change but overall there is nothing to explain further.

Comparison of results

Similarly to results of task 2, both methods do not differ significantly from one another in terms of results. This can be seen on 3, where the output difference cannot even be seen (due to one point drawn on the other).

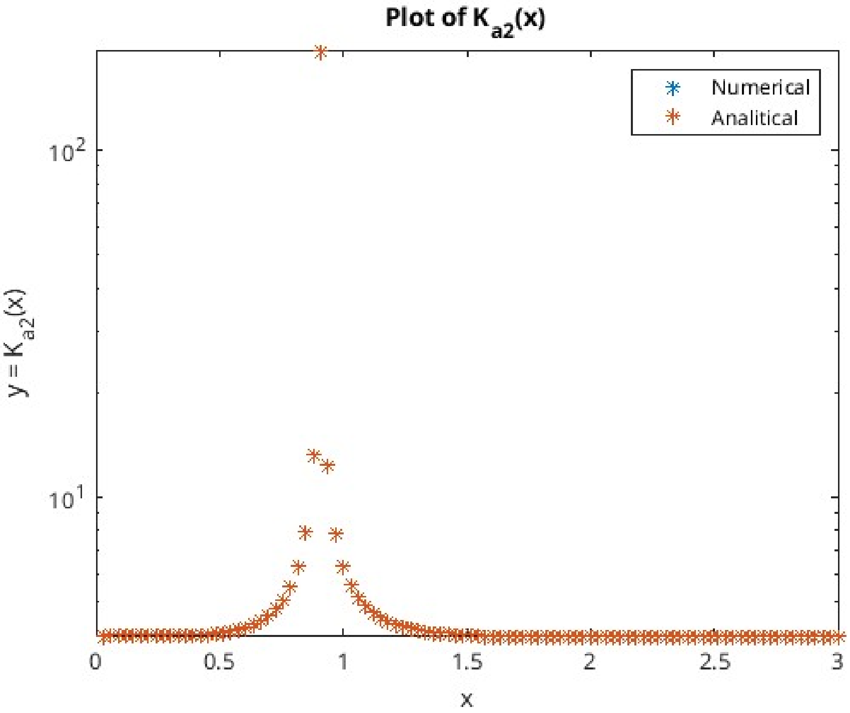

Figure 3: Comparison of calculating error via numerical and analytical way

Task 4 - Summary and Conclusions

It's important to know that each floating point calculation if corrupted by error so before analyzing the results it may be necessary to analyze if the result is within satisfiable range of our precision. There is more to error of the computation than just simple inaccuracy of value stored. Instead, each calculation introduces more error.

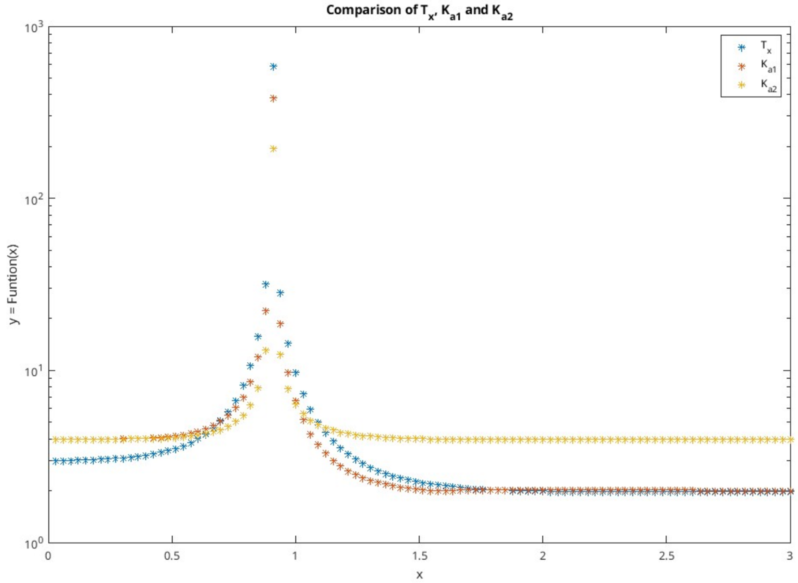

Figure 4: Comparison of error from all 3 methods applied

Different ways of calculating the same function can often have big impact on the quality of the results. As one can clearly see from Figure 4 different methods yield different error results for the same input. In particular, the difference between the K_{a1} and K_{a2}, for $x \geq 1$ is worthy of your attention. Furthermore, error introduced by the inaccurate representation is different from the one that is introduced by the inaccurate intermediate representation. Knowing that allows to design better, more accurate algorithms. While developing and testing those methods it is ok to use simplified formulas for calculating the error as they provide the same results as the ones tailored to the specific problem.

References

[1] R. Z. Morawski: Numerical methods (ENUME) – *2. Accuracy and complexity of computing*, Warsaw University of Technology, Faculty of Electronics and Information Technology

[2] Tailor Series Table and definitions:
https://people.math.sc.edu/girardi/m142/handouts/10sTaylorPolySeries.pdf

[3] Mathworks, official anonymous-function documentation:
https://www.mathworks.com/help/matlab/matlab$_prog/anonymous-functions.html$

$[4] Mathworks, officialanonymous - functiondocumentation:$
$https://www.mathworks.com/help/releases/R2022b/matlab/ref/randi.html?searchHighlight =$
$randis_tid = doc_srchtitle$

Appendix: Listing of the developed programs

Source code for Task 1

```
% clear previous experiment results
clc, clearvars, close all

% define domain and function
x = linspace(0,3,100);
y = @(x) cos(x)./x.^3 - x.^2;

% define erronous domain
esim = 1.0e-8;
x_eps = x.*(1+esim);

% calculate true y and epsilon y
ydot = y(x);
yeps = y(x_eps);

% calculate T numerically
numerator = yeps - ydot;
abs_error = abs(numerator ./ ydot);
tx = 1/esim * abs_error;

% calculate T analiticaly (formula calculated in the report)
T1 = -sin(x)./(x.^2);
T2 = -3*cos(x)./(x.^3);
T3 = 2*x.^2;

Tn = (abs(T1) + abs(T2) + abs(T3));
tx_analitical = Tn./abs(ydot);

% plot results to validate both methods result in simmilar results
semilogy(x,tx, '*');
hold on
semilogy(x,tx_analitical, '*');
title("Plot of T(x)"), xlabel("x"), ylabel("y = T(x)"),
legend("Numerical", "Analitical")
```

Source code for Task 2

```
% clear previous experiment results
clc, clearvars, close all;

% define domain and function
x = linspace(0,3,100);
y = cos(x)./x.^3 - x.^2;
eta = 1e-08;

% calculate T numerically
T1 = abs((cos(x) ./ x.^3)./y);
T2 = abs((-1* x.^2)./y);

% calculate T analiticaly (formula calculated in the report)
tx_analitical = 3*T1 + T2 + 1;

% begin of numerical calculations
tx_numerical = zeros(1, 100);

% create a function that creates a vector of random values: -1 or 1
rand_sign_vec = @() (randi(2,1,5)-1.5)*2;
for i = 1:300
    en = eta * rand_sign_vec();
    v1 = T1 * (1+en(1));
    v2 = T1 * (1+en(2));
    v3 = T1 * (1+en(3));
    v4 = T2 * (1+en(4));
    v5 = (1+en(5));
    ytilde = v1+v2+v3+v4+v5;
    tx_numerical = max(tx_numerical, ytilde);
end

% plot results to validate both methods result in simmilar results
semilogy(x,tx_numerical, '*');
hold on
semilogy(x,tx_analitical, '*');
title("Plot of K_{a1}(x)"), xlabel("x"), ylabel("y = K_{a1}(x)"),
legend("Numerical", "Analitical")
```

Source code for Task 3

```
% clear previous experiment results
clc, clearvars, close all;

% define domain and function
x = linspace(0,3,100);
y = cos(x)./x.^3 - x.^2;
eta = 1.0e-08;

% calculate T numerically
T1 = abs((cos(x) ./ x.^3)./y);
T2 = abs((-1* x.^2)./y);
T3 = abs(((cos(x) - x.^5) ./ x.^3)./y);

% calculate T analiticaly (formula calculated in the report)
tx_analitical = T1 + T2 + 3*T3;

% begin of numerical calculations
tx_numerical = zeros(1, 100);

% create a function that creates a vector of random values: -1 or 1
rand_sign_vec = @() (randi(2,1,5)-1.5)*2;
for i = 1:300
    en = eta * rand_sign_vec();
    v1 = T1 * (1+en(1));
    v2 = T2 * (1+en(2));
    v3 = T3 * (1+en(3));
    v4 = T3 * (1+en(4));
    v5 = T3 * (1+en(5));
    ytilde = v1+v2+v3+v4+v5;
    tx_numerical = max(tx_numerical, ytilde);
end

% plot results to validate both methods result in simmilar results
semilogy(x,tx_numerical, '*');
hold on
semilogy(x,tx_analitical, '*');
title("Plot of K_{a2}(x)"), xlabel("x"), ylabel("y = K_{a2}(x)"),
legend("Numerical", "Analitical")
```

Source code for Task 4

```
x = linspace(0,3,100);
y = @(x) cos(x)./x.^3 - x.^2;
esim = 1.0e-8;
xeps = x.*(1+esim);
ydot = y(x);

yeps = y(xeps);
numerator = yeps - ydot;
abs_error = abs(numerator ./ ydot);
tx_numerical = 1/esim * abs_error;
semilogy(x,tx_numerical, '*');
hold on

T1 = abs((cos(x) ./ x.^3)./ydot);
T2 = abs((-1* x.^2)./ydot);
tx_numerical = 3*T1 + T2 + 1;
semilogy(x,tx_numerical, '*');
hold on

T1 = abs((cos(x) ./ x.^3)./ydot);
T2 = abs((-1* x.^2)./ydot);
T3 = abs(((cos(x) - x.^5) ./ x.^3)./ydot);
tx_numerical = T1 + T2 + 3*T3;
semilogy(x,tx_numerical, '*');

title("Comparison of T_x, K_{a1} and K_{a2}"), xlabel("x"),
ylabel("y = Funtion(x)"), legend("T_x", "K_{a1}", "K_{a2}")
```