Name, Surname: Cansu YANIK

Student Number: 150170704

ITU Computer and Informatics Faculty

BLG 202E Numerical Methods in CE 2018-2019 Spring

Homework-1

1. If we take β=e for e and β=π for π, we can have two finite floating point systems for the exact representation for these two numbers. According to these systems t can be chosen. But it is difficult to find a single finite floating point system that works for both numbers. There is not any simple rule about it because it is not known whether there is a dependence between π and e numbers or not. Even if such a rule, we would need a system to keep very extremely large numbers. If we had a large system to keep these numbers, we could have an exact representation for e and π numbers.

2. The number $\frac{8}{7}$ = 1.14285714285714... obviously has no exact representation in any decimal floating point system (β = 10) with finite precision t.

- To be able to represent this number in base 10, c*hopping* or *rounding* can be used. Therefore, t has to be an integer.

  Assume that t = 4, chopping gives:

  $$\frac{8}{7} \simeq \left(\frac{1}{10^0} + \frac{1}{10^1} + \frac{4}{10^2} + \frac{2}{10^3}\right) \text{x } 10^0 = 1{,}142 \text{ x } 10^0$$

  On the other hand, with rounding note that, $d_t = 8 > 5$, so $\beta^{1-t} = 10^{-3}$ is added to the number before chopping, which gives 1.143 x 10^0.

- Or, if we select β as 7, we can also have a finite floating point system in which this number does have an exact representation (the digit series has finite length).

  $\beta = 7 \quad \Rightarrow \quad (1 + \frac{1}{7})_7 = (1{,}1)_7 \quad \Rightarrow \quad t = 2$

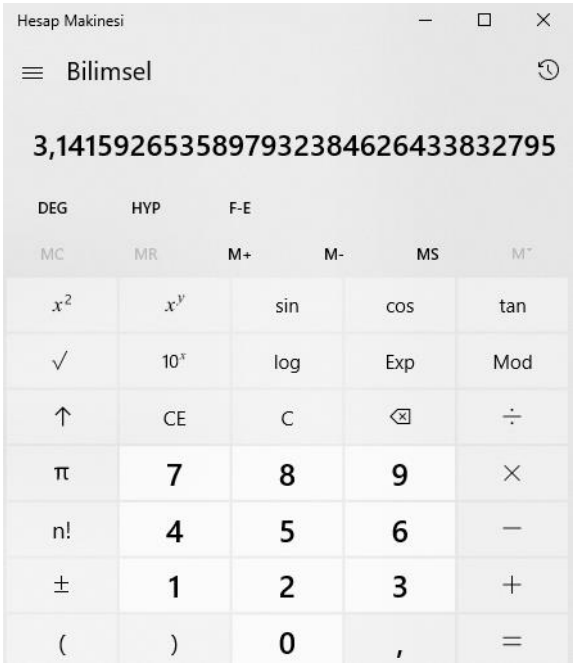
3. For a general floating point system (β, t, L, U) the rounding unit is calculated in this way:

$$\eta = \frac{1}{2}\beta^{1-t}$$

I decided to show the rounding unit of my computer calculator. To be able to determine approximately the rounding unit of my calculator, I need to know the values of β and t.

1. For my computer calculator, I know what is the number of base, which is 10. Therefore, $\beta = 10$.

2. I also need to know what is the number of t. The maximum number of digits that the calculator can show can be taken as t. In order to find the maximum digit number, I can give a number to the calculator to calculate a floating point with infinite number of digits. Therefore, I will choose π for this number.



According to this result, the value of t is 32 for my calculator. Therefore, I get:

$$\eta = \frac{1}{2}10^{-31} = 0.5 \times 10^{-31}$$

This is the rounding unit of my computer calculator.

4. a. There is the MATLAB script which evaluates the polynomial function given below at 161 equidistant points in the interval [1.92,2.08] using two methods. Firstly, I defined the equidistant points and bounds as variables. Using these variables, I calculated the required number of pieces, and I divided the interval to these equal pieces.
The remaining step was to calculate and plot the graphs.

$$f(x) = (x-2)^9$$
$$= x^9 - 18x^8 + 144x^7 - 672x^6 + 2016x^5 - 4032x^4 + 5376x^3 - 4608x^2 + 2304x - 512.$$

```
n=161;                                  %161 equidistant points
bound1=1.92; bound2=2.08;               %Interval bounds
step = (bound2 - bound1)/(n - 1);       %Calculating the number of equal pieces
x = (bound1 : step : bound2);           %Dividing the interval to these equal pieces


%Expanded form of the polynomial function
f = x.^9 - 18*x.^8 + 144*x.^7 - 672*x.^6 + 2016*x.^5 - 4032*x.^4 + 5376*x.^3 - 4608*x.^2 + 2304*x.^1 - 512;

%subplot(2,1,1)
figure(1)
plot(x,f);          %Plotting the expanded form of the function
xlabel('x')         %Giving names to the x and y labels
ylabel('f(x)')
grid

%Calculating the polynomial function directly
g = (x-2).^9;

%subplot(2,1,2)
figure(2)
plot(x,g);          %Plotting the closed form of the function
xlabel('x')         %Giving names to the x and y labels
ylabel('g(x)')
grid
```

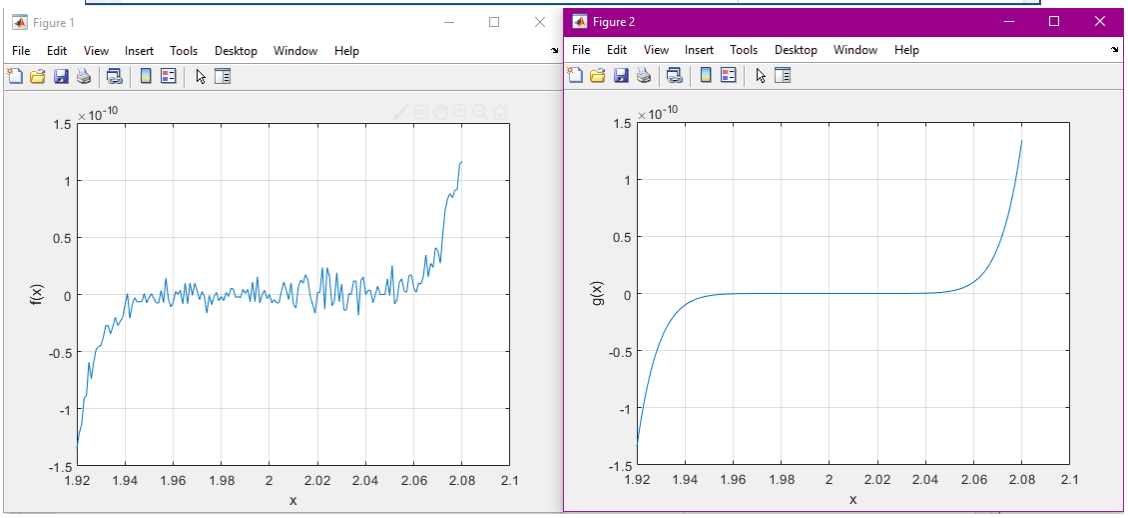


Figure1 represents the result of the nested evaluation for evaluating the polynomial in the expanded form, and Figure2 shows the result of directly calculation of the function.

b. The reason of the difference between these two graphs is due to the roundoff error. Since the expanded form of the function has more number representations and elementary operations than the second one, there is more roundoff error and this affects the result and the smoothness of the graph. In the figure2, this effect is reduced by doing direct calculation.