CS 323: Homework Solutions 1

Due on 2/11/2014

Apostolos Gerasoulis

Vilelmini Kalampratsidou

Contents

Problem 1 (Question i)

Problem 1 (Question ii)

n	f(z)	Pn(z)	error
1.000000000000000	0.841470984807897	1.000000000000000	0.158529015192103
2.000000000000000	0.841470984807897	1.000000000000000	0.158529015192103
3.000000000000000	0.841470984807897	0.833333333333333	0.008137651474563
4.000000000000000	0.841470984807897	0.833333333333333	0.008137651474563
5.000000000000000	0.841470984807897	0.841666666666667	0.000195681858770
6.000000000000000	0.841470984807897	0.841666666666667	0.000195681858770
7.000000000000000	0.841470984807897	0.841468253968254	0.000002730839643
8.000000000000000	0.841470984807897	0.841468253968254	0.000002730839643
9.000000000000000	0.841470984807897	0.841471009700176	0.000000024892280
10.000000000000000	0.841470984807897	0.841471009700176	0.000000024892280

Problem 1 (Question iii)

By definition we have

$$Rn(x) = \frac{(x-a)^{n+1}}{(n+1)!} f^{(n+1)}(c) \tag{2}$$

$$max(|Rn(x)|) = max(|\frac{(x-a)^{n+1}}{(n+1)!} f^{(n+1)}(c)|) \tag{3}$$

$$max(|Rn(x)|) \leq max(|\frac{(x-a)^{n+1}}{(n+1)!}|) * max(|f^{(n+1)}(c)|) \tag{4}$$

n	error
1.000000000000000	0.500000000000000
2.000000000000000	0.166666666666667
3.000000000000000	0.041666666666667
4.000000000000000	0.008333333333333
5.000000000000000	0.001388888888889
6.000000000000000	0.000198412698413
7.000000000000000	0.000024801587302
8.000000000000000	0.000002755731922
9.000000000000000	0.000000275573192
10.000000000000000	0.000000025052108

Problem 1 (Question iv)

n=9

Problem 1 (Question v)

n	m	f(z)	Pn(z)	error_n_exact	error_n_m	error_m_exact
1	21	0.841470984807897	1.000000000000000	0.158529015192103	0.000000000000000	0.158529015192103
2	22	0.841470984807897	1.000000000000000	0.158529015192103	0.166666666666667	0.008137651474563
3	23	0.841470984807897	0.833333333333333	0.008137651474563	0.008333333333333	0.000195681858770
4	24	0.841470984807897	0.833333333333333	0.008137651474563	0.008134920634921	0.000002730839643
5	25	0.841470984807897	0.841666666666667	0.000195681858770	0.000195656966490	0.000000024892280
6	26	0.841470984807897	0.841666666666667	0.000195681858770	0.000195682018599	0.000000000159828
7	27	0.841470984807897	0.841468253968254	0.000002730839643	0.000002730840404	0.000000000000762
8	28	0.841470984807897	0.841468253968254	0.000002730839643	0.000002730839640	0.000000000000003

Problem 2

In 32-bit arithmetic, the first bit is reserved as a sign bit. The largest positive number can be represented by the remaining 31 bits is $2^{31} - 1$. The smallest positive integers is $(000...001)_2 = (1)_{10}$.

In the case of 32 bit floating numbers, there is 1 bit reserved for sign, 8 bit reserved for exponent and 23 bit reserved for mantissa. Based on IEEE format 2^{128}, is the largest exponent and 2^{-127} is the smallest. The largest mantissa number is $2-2^{-23}$ and the smallest number is 2^{-23}. As a results the largest positive number is $(2-2^{-23})*2^{128}$ and the smallest positive number is $1*2^{-127}$

Problem 3

(i)This is easier to understand if we think about the problem in decimal form with a limited number of significant figures:

$$1-3(\frac{4}{3}-1)=1-3(1.3333...333-1)=1-3*0.3333...333=1-0.999...999=0.000...0001 \tag{10}$$

The error creeps in when we compute 4/3 which has an infinite number of significant figures, and this error carries out until the end. Of course, the computation is done in binary, but the concept is the same.

(ii)This again is the result of 0.1 not having an exact representation in binary, so there is some concatenation error due to the finite number of bits used to represent it. This error can be seen if we print the value $a-1=-1.11*10^{-16}$

(iii)The issue here is that c is exactly 1 because the concatenated representation 10^{-16} of is subtracted from itself getting zero exactly, and then 1 is added. For b however, when 1 is added to 10^{-16} the mantissa of 10^{-16} is shifted to the right by a lot to accommodate it (1.00..001011..) so a small error is introduced on concatenation (it loses precision). Therefore when 10^{-16} is subtracted this error remains and b is not exactly one ($b-1=-1.11*10^{-16}$).

(iv)In this case we have the opposite effect, where the least significant digits of the square root are so small that the mantissa doesnt have enough digits to hold them and they are concatenated out. This way we lose precision and the square root is perceived to be equal to 1.

Problem 4

We know that

$$\text{Relative error} = \frac{\text{actual value} - \text{approximate value}}{\text{actual value}} \tag{11}$$

Observing the results represented on the table below, we can see that the relative error gets bigger when x approaches 0.

x	f	approx. f	Error	Relative Error
0.1000000000	0.0998334166	0.0998334166	0.0000000000000006	0.0000000000000057
0.0100000000	0.0099998333	0.0099998333	0.0000000000000006	0.0000000000000552
0.0010000000	0.0009999998	0.0009999998	0.0000000000000190	0.0000000000189651
0.0001000000	0.0001000000	0.0001000000	0.0000000000001372	0.0000000013720691
0.0000100000	0.0000100000	0.0000100000	0.0000000000004139	0.0000000413868511
0.0000010000	0.0000010000	0.0000010000	0.0000000000444493	0.0000444493034665
0.0000001000	0.0000001000	0.0000001000	0.0000000000399719	0.0003997188062402
0.0000000100	0.0000000100	0.0000000000	0.0000000100000000	1.0000000000000000

Problem 5

For this question we edit function "fun" so as to built the Taylor polynomial using $h(x)=exp(x)$.

```
function [ ff, result ,error] = fun( z,a,n )
syms x real;
f=exp(x);
sum=subs(f,'x',a);
prod=1;
for j=1:n
```

```
prod=prod*(z-a)/j;
sum=sum+prod*subs(diff(f,x,j),'x',a);
end
result=sum;
ff=(exp(z)-1)/z;
error=abs(result-ff);
```

Then we recall the function for several different values of x.

```
ID=fopen('text.txt','w');
fprintf(ID,' f  `f error relative\n', 'relative error')

for x=[0.1, 10^(-5), 10^(-10), 10^(-15), 10^(-17), 10^(-20)]
   [ff, result,error]=fun(x,0,2);
    fprintf(ID,' %1.2d, %1.10f,%1.10f, %1.10f \n', x,ff, result, error);
end
```

The results are shown of the table below. We observe that while x approaches 0, error tends to became 0. However, when x get smaller than 10^{-15} then $f(x)$ is equal to 0 which is incorrect, since $\lim_{x->0} \frac{exp(x)-1}{x} = 1$.

x	f	Taylor approx.	Error
1.00e-01	1.0517091808	1.1050000000	0.0532908192
1.00e-05	1.0000050000	1.0000100001	0.0000050000
1.00e-10	1.0000000827	1.0000000001	0.0000000826
1.00e-15	1.1102230246	1.0000000000	0.1102230246
1.00e-17	0.0000000000	1.0000000000	1.0000000000
1.00e-20	0.0000000000	1.0000000000	1.0000000000