Numerical Analysis
Assignment 3

Chuan Lu

September 9, 2017

Problem 1. Problem 1.26

(a). According to Taylor series,

$$f(x) = \frac{e^x - e^{-x}}{2x} = \frac{1}{2x}\left(1 + x + \frac{x^2}{2} + \frac{x^3}{3!} - 1 + x - \frac{x^2}{2} + \frac{x^3}{3!} + O(x^5)\right) = 1 + \frac{x^3}{6} + O(x^4).$$

Thus

$$\lim_{x\to 0} f(x) = 1.$$

(b).

$$\begin{aligned} f(x) = \frac{\log(1-x) + xe^{\frac{x}{2}}}{x^3} &= \frac{1}{x^3}\left(-x - \frac{x^2}{2} - \frac{x^3}{3} - \frac{x^4}{4} + x\left(1 + \frac{x}{2} + \frac{x^2}{8} + \frac{x^3}{48}\right) + O(x^5)\right) \\ &= \frac{1}{x^3}\left(-\frac{5}{24}x^3 - \frac{11}{48}x^4 + O(x^5)\right) \\ &= -\frac{5}{24} - \frac{11}{48}x + O(x^2). \end{aligned}$$

Thus

$$\lim_{x\to 0} f(x) = -\frac{5}{24}.$$

Problem 2. Problem 1.31

Solution. The subroutine and the main program are shown as follows.

```
function res = smallest_to_largest(xx)
% Add an array from the smallest number to the largest number;
% `xx`: The input array, arranged from the largest to the smallest.
    res = single(0);
    for i = length(xx):-1:1
        res = res + xx(i);
    end
```

```
function res = largest_to_smallest(xx)
% Add an array from the largest number to the smallest number;
% `xx`: The input array, arranged from the largest to the smallest.
    res = single(0);
    for i = 1:length(xx)
        res = res + xx(i);
    end
```

```
function res = precise(xx)
% Add an array using double precision and chop/round the result to single
    precision;
% `xx`: The input array, arranged from the largest to the smallest.
    res = sum(xx);
    res = single(res);
```

```
% Main Script
n = 1e7;
xx = n:-1:1;
a = single(1./xx);
b = single(1./(xx.^2));
c = single(1./(xx.^3));
d = single(((-1).^xx)./xx);

a1 = smallest_to_largest(a);
a2 = largest_to_smallest(a);
a0 = precise(a);
disp([abs(a0-a1) abs(a0-a2)]);

b1 = smallest_to_largest(b);
b2 = largest_to_smallest(b);
b0 = precise(b);
disp([abs(b0-b1) abs(b0-b2)]);

c1 = smallest_to_largest(c);
c2 = largest_to_smallest(c);
c0 = precise(c);
disp([abs(c0-c1) abs(c0-c2)]);

d1 = smallest_to_largest(d);
d2 = largest_to_smallest(d);
d0 = precise(d);
disp([abs(d0-d1) abs(d0-d2)]);
```

Result. The output is as below.

```
>> main
   1.2897701e+00   7.4214935e-03

   2.0873547e-04   1.1920929e-07

   6.1988831e-06               0

   9.6559525e-06               0
```

Problem 3. Problem 1.32

Solution. In this computer system, machine epsilon is $\epsilon = \beta^{-n}$.
Then we have

$$\begin{aligned} p_m &= a_0 a_1 \cdots a_m (1+w) \\ &= a_0 a_1 \cdots a_m (1+\epsilon_1)(1+\epsilon_2)\cdots(1+\epsilon_m) \end{aligned}$$

Since we can ignore those high-order terms in the error, we have

$$|w| = \left| \sum_{i=1}^{m} \epsilon_i \right| \leq n\delta,$$

where δ is the upper bound of error terms, which means $\delta = \frac{\epsilon}{2}$ if the system uses rounding and $\delta = \epsilon$ if the system uses chopping.
When using rounding, we may suppose the error terms are independent variables satisfying a uniform distribution between $[-\delta, \delta]$, where $\delta = \frac{\epsilon}{2}$. Then

$$E[w] = n\bar{\epsilon},\ \bar{\epsilon} \sim \mathcal{N}\left(0, \frac{\delta^2}{3n}\right).$$

When using chopping, we can suppose $-\sigma \leq \epsilon_i \leq 0$, where $\sigma = \epsilon$. In the same way,

$$E[w] = n\bar{\epsilon},\ \bar{\epsilon} \sim \mathcal{N}\left(-\frac{\sigma}{2}, \frac{\sigma^2}{3n}\right).$$