Review of C-Programming: Creating Sine Function with Taylor Series

Numerical Programming

mod .2023.08.18

Pre-requisite

1) Do Tutorials : Preparation for NP

2) Complete Assignment_0: Power and Factorial

3) Review: Taylor Series

Numerical Programming Tip: Coding (Σ / Π)

- **General case for Σ, Π**

mathematical formula

$$y = \sum_{k=N_1}^{N_{\text{end}}} g(x,k)$$

Pseudo code

```
y = 0
for k = N_1 to N_end
    y = y + g(x,k)
end
```

mathematical formula

$$y = \prod_{k=N_1}^{N_{\text{end}}} g(x,k)$$

Pseudo code

```
y = 1
for k = N_1 to N_end
    y = y * g(x,k)
end
```

Numerical Programming Tip: Coding (Σ / Π)

- **Examples for Σ**

mathematical formula

(1)

$$y = \sum_{k=1}^{N} k = 1 + 2 + 3 + \cdots + (N-1) + N$$

programming

```
y = 0
for k = 1 to N
    y = y + k
end
```

mathematical formula

(2)

$$y = \sum_{k=1}^{N} x \cdot k = x + 2x + 3x + \cdots + (N-1)x + Nx$$

given value

programming

```
y = 0
for k = 1 to N
    y = y + (x · k)
end
```

Numerical Programming Tip: Coding (Σ / Π)

- **Examples for Π**

mathematical formula

(1) power()

$$y = x^N = x \cdot x \cdot x \ldots \cdot x = \prod_{k=1}^{N} x$$

pseudocode

```
y = 1
for k = 1 to N
    y = y * (x)
end
```

mathematical formula

(2) factorial ()

$$y = N! = 1 \cdot 2 \cdot \ldots (N-1) \cdot N = \prod_{k=1}^{N} k$$

pseudocode

```
y = 1
for k = 1 to N
    y = y * k
end
```

Sin(x) with Taylor Series

Part 1) Create a simple function that returns the output of sine x.

- Taylor series for $\mathbf{sin(x)}$ (where $x_0 = 0, \sin(x_0) = 0$)

Impossible to calculate

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k+1}}{(2k+1)!}$$

$$\rightarrow \quad \sin(x) \approx \sum_{k=0}^{N-1} (-1)^k \frac{x^{2k+1}}{(2k+1)!} = S_N$$

- Program stop condition

  when $k > N_{max}$, N_{max}=20

- Using the above, you should create the function below

radian unit

```
double sinTaylor(double _x)
```

Sin(x) with Taylor Series

- **Algorithm for sinTaylor(x)**

Get "x" (user input [rad])
Initialize

$N_{\max} = 20$

S_N=0

$$S_N = \sum_{k=0}^{N_{\max}} (-1)^k \frac{x^{2k+1}}{(2k+1)!}$$

return S_N

This is the most difficult part in the algorithm .
Lets create a code for Σ (sigma)

You will meet sigma frequently in the class

Sin(x) with Taylor Series

- **Taylor series for sin(x)**

mathematical formula

$$\sin(x) = y \approx \sum_{k=0}^{N-1} (-1)^k \frac{x^{2k+1}}{(2k+1)!}$$

$$= \underbrace{x}_{k=0} - \underbrace{\frac{x^3}{3!}}_{k=1} + \underbrace{\frac{x^5}{5!}}_{k=2} + \cdots + \underbrace{(-1)^{N-1} \frac{x^{2(N-1)+1}}{(2(N-1)+1)!}}_{k=N-1}$$

programming

Function sin(x)
    $y = 0$
    for $k = 0$ to $N-1$
        $y = y + (-1)^k \dfrac{x^{2k+1}}{(2k+1)!}$
    end
return y

Sin(x) with Taylor Series

- **Taylor series for sin(x)**

$$\sin(x) \approx \sum_{k=0}^{N-1} (-1)^k \frac{x^{2k+1}}{(2k+1)!}$$

Get "x" (user input [rad])

Initialize $N_{\max} = 20, \quad S_N$=0

$$S_N = \sum_{k=0}^{N_{\max}} (-1)^k \frac{x^{2k+1}}{(2k+1)!}$$

return S_N

$S_N = 0$

for $k = 0$ to $N - 1$

$$S_N = S_N + (-1)^k \frac{x^{2k+1}}{(2k+1)!}$$

end

```
sinTaylor(x)
{
    int Nmax = 20;
    double S_N = 0;

    for (int k = 0; k < Nmax; k++)
        S_k=pow(-1, k) * pow(x, 2 * k + 1) / factorial(2 * k + 1);
        S_N = S_N + S_k

    return S_N;
}
```

Exercise 1

Exercise 1: Create sin(x) with Taylor series

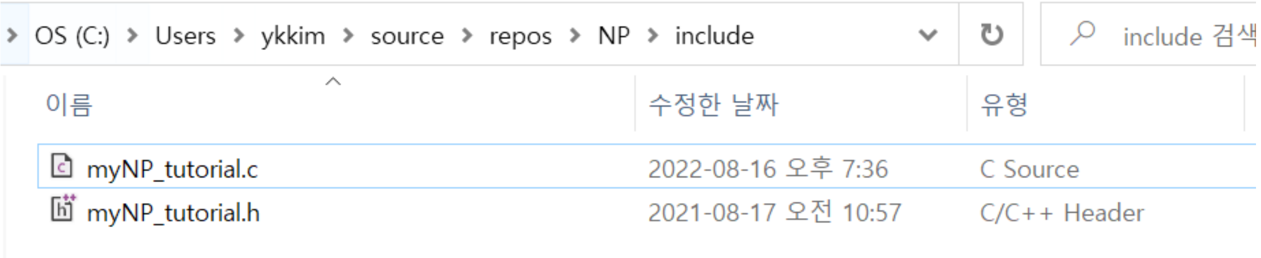
1. Create a new project " **TU_TaylorSeries**" with Visual Studio
2. Create the new source file and name it as "**C_taylorSeries_exercise.c**"
3. Copy the source code from this link: C_taylorSeries_exercise.c
4. Fill the definition of **sinTaylor(rad)** in the main source.
5. Compare your answer and calculate the absolute error
   $\sin(\pi/3) = 0.86602540378$
6. Create **sindTaylor(deg)** for degree unit input and output. (use sinTaylor(rad))

Use your own **power()** and **factorial()** function

Exercise 2

Exercise 2: Define your sinTaylor(x) in header file

1. Create a new project “ **TU_TaylorSeries_Part2**” with Visual Studio
2. Create the new source file and name it as “**C_taylorSeries_exercise_part2.c**”
3. Copy the source code from this link: C_taylorSeries_exercise_part2.c
4. Create a new header file named as **myNP_tutorial.h** and **myNP_tutorial.c**
   - These files can be downloaded from the link
   - These files should be saved in “ \include\” folder.



5. Your **sinTaylor(rad)** of Exercise 1 should be declared and defined in the header file.
6. Run and check the answer