Curtin University

DEPARTMENT OF MATHEMATICS & STATISTICS

ACCELERATED MATHEMATICS UNITS MATH1017 AND MATH1021

Lab Sheet 5

This week we want you to code the *Maclaurin series* for each of the functions sin and cos. This week we are only interested in the domain $[0, \pi/4]$. Despite each of these series being convergent for all real x, errors accumulate for large $|x|$. Next week we'll use symmetries of sin to use what you code this week, to provide a good coding solution for $\sin(x)$ for all $x \in \mathbb{R}$.

Background

What you will learn next semester is that an *alternating series* $\sum_{n=1}^{\infty} a_n$ – one for which the terms a_n alternate in sign – converges if the magnitude of the terms decreases with limit 0. Moreover, if that condition holds, then the error in truncating the series at a given point is less than the magnitude of the next term. Thus, the criterion used for terminating the summation of the Maclaurin series of the exponential function last week, will actually be valid for both $\sin(x)$ and $\cos(x)$ on $[0, \pi/4]$, so long as the ratio $|a_{n+1}/a_n| < 1$, at the point where the summation terminates. You will find that n does not need to be very large for that condition to be true.

The Maclaurin series for $\sin(x)$ and $\cos(x)$ are as follows:

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \cdots + (-1)^n \frac{x^{2n-1}}{(2n-1)!} + \cdots$$
$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \cdots + (-1)^{n-1} \frac{x^{2n}}{(2n)!} + \cdots$$

Ingredients: `proc`, `for` (and/or `while`).

The idea is to keep adding terms of the series until the last term added has magnitude less than some "tolerance", which can either be "hard-wired" (i.e. unchangeable) in the body of your `proc` or can be a second argument. You might like to call your `proc` for $\sin(x)$,

`sinx`

(you can't call it `sin` because this is the actual function that does it in Maple).

Here is a suggested approach.

1. As with `expx`, first write a `for` loop to calculate
$$x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!}.$$
You will need to keep track of two things as you go round the loop, the sum so far (suggested variable: `s` – don't use `sum`, it's a Maple keyword), and the term you are up to (`ai` might be a nice choice). Note that you can get the next `ai` quite easily from the previous `ai`. It may help to realise that $2i - 1$ is the i^{th} odd number.
As with `expx`, while you are getting this loop correct, leave `x` uninstantiated and just change the `to` limit gradually up to 5. It will be easiest if you think of x as the first term, and using the count variable `i`, the `i`th term as $(-1)^i x^{2i-1}/(2i-1)!$. Note, that you will not need factorial if, as suggested above, the next `ai` is computed from the previous `ai` appropriately.
Possible hints. You may want to have a variable `oi` for the `i`th odd number, that you update each time round the loop, and you may want to update `ai` after you've added it to `s`, instead of before.

2. In order to make Maple treat `s` as a floating point number, make sure you make its initial value a number followed by a decimal point. (Your `s` initialisation line should probably be: `s := 0.;`.)

3. Again exploit Maple's `for` design, to make the loop a `while` at the same time, by replacing

   ```
   to ...
   ```

   with

   ```
   while ai > ...
   ```

   for some appropriate `...`.

4. Embed your `for` in a `proc`. You should then have local variables `s`, `ai` and count variable `i`, and possibly, your "tolerance" variable (you may instead like it to be changeable and passed as a 2nd argument to your `proc`). Note that `1e-12` is an easy way to represent the floating point number 1.0×10^{-12}.

5. Run your `sinx` function (i.e. `proc`) with argument `x` set to 0.7, and compare it with Maple's `sin(0.7)`. You should get good agreement to within the tolerance you set.

6. Now redo the above for $\cos(x)$. You may like to think of 1 as its zeroth term, and set `s := 1.;` initially (outside the loop), and think of $2i$ as the i^{th} even number.
   Possible hints. You may want to have a variable `ei` for the `i`$^{\text{th}}$ even number, that you update each time round the loop, and you may want to update `ai` before you add it to `s`.