Curtin University
DEPARTMENT OF MATHEMATICS & STATISTICS

ACCELERATED MATHEMATICS UNITS MATH1017 AND MATH1021

Lab Sheet 4

As stated last week, we want you to code a function to evaluate e^x this week, and since some of you will have to wait some time for Maple to start up – hopefully, this problem won't last too much longer – you may as well spend that time usefully with the following background information. If you find the background too verbose, skip to **Summary** and **Example** and see if you already follow that. If Maple has started up before you've finished reading the background, read it later. It's more important to get on with the lab.

Background

You may have noticed that *Maclaurin series* was listed in the Lecture Schedule. This semester we can only expose you to it, since before you see that, you should know what a *convergent sequence* is, and how the *convergence* of an *infinite series* is defined in terms of a particular *sequence*.

Firstly, one way of thinking about a *sequence* is as a *function* whose domain is $\mathbb{N}$, rather than intervals that are subsets of $\mathbb{R}$. The usual notation we use reflects this; instead of writing

$$f(1), f(2), \ldots, f(n), \ldots$$

for some function f, we would write using subscript notation:

$$f_1, f_2, \ldots, f_n, \ldots$$

and we often use the letter 'a' rather than 'f'. You will be familiar with the Fibonacci sequence: $1, 1, 2, 3, 5, 8, 13, \ldots$.

A *convergent sequence* is one with a *limit*. Since, a sequence is really just a function with domain $\mathbb{N}$, you would expect its formal definition to be like that of functions with an adjustment for the change from a *continuous* domain to a *discrete* one. So typically, where we often write $f(x)$ for a function, we will write a_n to represent a sequence. In the literature an "infinite sequence", i.e. a sequence with an infinite number of element is often written as a set,

$$\{a_n \mid n \in \mathbb{N}\}.$$

This is unfortunate notation, as a *set* is not *ordered*, and yet an order is implied. A better notation is

$$(a_n)_{n=1}^{\infty}.$$

Maple allow you to write a finite sequence with a formula with `seq` and essentially uses `(...)` but the brackets "disappear":

```
seq(n^2, n = 1 .. 10)
```

gives the sequence $(n^2)_{n=1}^{10}$ and displays as

$$1, 4, 9, 16, 25, 36, 49, 64, 81, 100$$

Definition. A *sequence* a_n is said to **converge** to limit L, which we write formally as

$$\lim_{n\to\infty} a_n = L$$

or, informally as,

$$a_n \to L \text{ as } n \to \infty$$

if

$$\forall\varepsilon > 0\ (\exists N \in \mathbb{N} \text{ s.t. } n > N \implies |a_n - L| < \varepsilon).$$

As with function limits it's when ε is small where it matters. So one should read the above sentence this way:

> For every positive ε, no matter how small, there exists a "point" in the sequence (determined by the element a_N) such that every element beyond a_N is within ε of L.

Of course the two sequences mentioned so far are not *convergent sequences*, but the sequence

$$3.1, 3.14, 3.141, 3.1415, 3.14159, \ldots$$

where the n^{th} element is π to n decimal places, converges to π, and if $\varepsilon = 10^{-k}$ we can choose $N = k$ to show that $a_n \to \pi$ as $n \to \infty$.

Now the numbers that form a sequence are called **terms**. So an informal, but accurate way of describing the usual way we write a sequence is that, we write down "its terms in order and separate them by commas". An informal way of thinking about a **series**, is that it's just a *sequence* with all the *comma*s replaced by + signs. So from a **Geometric sequence**

$$\left(2^{-(n-1)}\right)_{n=1}^{\infty} = (1, \tfrac{1}{2}, \tfrac{1}{4}, \ldots, 2^{-(n-1)}, \ldots)$$

we can form a **Geometric series**

$$\sum_{n=1}^{\infty} \frac{1}{2^{n-1}} = 1 + \tfrac{1}{2} + \tfrac{1}{4} + \cdots + 2^{-(n-1)} + \cdots$$

In order to define *convergence* of a *series*, we first define the *sequence of partial sums* of a series. If

$$\sum_{i=1}^{\infty} a_i$$

is the series, then its **sequence of partial sums** is:

$$a_1, a_1 + a_2, \ldots, \sum_{i=1}^{n} a_i, \ldots$$

i.e. the n^{th} term of the sequence of sum of a series is the sum of the first n terms of that series.

Summary

A **sequence**,

$$a_1, a_2, \ldots, a_n, \ldots$$

converges to a **limit** L, if

$$\forall \varepsilon > 0\ (\exists N \in \mathbb{N} \text{ s.t. } n > N \implies |a_n - L| < \varepsilon).$$

A **series**,

$$a_1 + a_2 + \cdots + a_n + \cdots$$

converges to a **limit** M, if its **sequence of partial sums**

$$s_1, s_2, \ldots, s_n, \ldots$$

converges to M, where the n^{th} term

$$s_n = a_1 + a_2 + \cdots + a_n.$$

Example

From a *geometric sequence*,

$$a, ar, ar^2, \ldots, ar^{n-1}, \ldots$$

we have a corresponding *geometric series*

$$a + ar + ar^2 + \cdots + ar^{n-1} + \cdots$$

whose *sequence of partial sums*

$$s_1, s_2, \ldots, s_n, \ldots$$

has general term

$$s_n = a + ar + ar^2 + \cdots + ar^{n-1} = \frac{a(1-r^n)}{1-r}$$

which converges only if $r^n \to 0$, which is only if $|r| < 1$, in which case the limiting sum s_∞ is

$$\frac{a}{1-r}.$$

The aim today is to write some code that returns an approximate evaluation of the *Maclaurin series* for e^x,

$$e^x = 1 + x + \frac{x^2}{2!} + \cdots + \frac{x^n}{n!} + \cdots$$

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`

```
expx
```

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

Here is a suggested approach.

1. First write a `for` loop to calculate
$$1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^{10}}{10!}.$$
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`.
While you are getting this loop correct, leave `x` uninstantiated and just change the `to` limit gradually up to 10. It will be easiest if you think of 1 as the zeroth term, and using the count variable `i`, the $\texttt{i}^{\text{th}}$ term as `x^i/i!`. Note, that you will not need factorial if, as suggested above, the next `ai` is computed from the previous `ai` appropriately. (What do you have to multiply `x^i/i!` by, to get `x^(i+1)/(i+1)!`?)

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. Depending, on how you've done your loop you will want your `s` initialisation line to be either: `s := 0.;` or `s := 1.;`

3. You can 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 `expx` function (i.e. `proc`) with argument `x` set to 2, and compare it with Maple's `exp(2.)` (with the decimal point, Maple will return a floating point number, rather than e^2). Try some other values up to 10 or so or more.

6. Now try `x` negative. Look at your `while` clause, it might not be the right test with `x` negative. Fix it, if necessary.
   Hint. You may need `abs`. (To find out what that does, do: `?abs`.)
   Compare your `expx` with Maple's `exp` for `x` values: `-1.`, `-2.`, `-5.`, `-10.`, `-20.`
   The first of these values shouldn't be too different, but the last one, in particular, shows something has gone seriously wrong.
   Hint. The series has terms with alternating signs. The first few terms have nonzero digits for the first few terms, but eventually all the early places in `s` become 0, which means the eventual significant decimal places are the least significant digits of the first few terms, which are not all kept due to rounding.

7. To fix the problem, identified for `x` negative, modify your algorithm so that for negative `x` it essentially returns `1/expx(-x,...)`, andcheck that this indeed fixes the problem.

8. At some point, to get the next `ai` from the previous `ai` one is multiplying by something less than 1. Indeed, this will be the case at the point that `ai` is less than "tolerance". Calling the ratio r that one multiplies `ai` to get the next `ai`, observe that the error in truncating the series at this point is no more than the geometric series

$$a + ar + ar^2 + \cdots$$

   for some choice of a, i.e. one can estimate the error in truncation to be no more than $a/(1-r)$. Compare that error with "tolerance". Does it explain the actual difference between what `expx` gives and what Maple's `exp` gives?