Math242 Lab 08: Series

Introduction

Determining if a sum

$$\sum_{n=0}^{\infty} a_n$$

converges can be re-phrased as determining if the sequence of partial sums

$$\left\{ \sum_{n=0}^{0} a_n, \sum_{n=0}^{1} a_n, \ldots, \sum_{n=0}^{N} a_n, \ldots \right\}$$

converges to a limit. Consider, for example, the infinite series

$$S = \sum_{n=1}^{\infty} \frac{1}{n} = \lim_{k \to \infty} \sum_{n=1}^{k} \frac{1}{n}.$$

We can construct a table of values for the first 101 partial sums of S and see if they seem to be approaching a limit. Instead of viewing the whole table, we will only view every 10th entry, as we did with the lab on sequences. Below we will make a function `s(n)` that computes the nth partial sum.

```
def s(n):
        # do a partial sum of the first n values of
        # \sum 1 / k
        k_values = list(range(1, n))
        partial_sum = sum([1 / k for k in k_values])
        return partial_sum

def print_table_sum(n_values, sum_values, step):
        # helper function which prints a pretty table
        print("n \t sum to n")
        print("----------------------")
        for index in range(0, len(n_values), step):
                print(n_values[index], "\t", sum_values[index])
```

Another way to see if a sequence may be converging or diverging is to plot the sequence of partial sums using ListPlot. We can do this with the first 101 terms of the above sequence:

```
  only put imports at the beggining of a python file!
import matplotlib.pyplot as plt

# plot
plt.plot(n_values, sum_values, ".")

# pretty it up
plt.xlabel("n")
plt.ylabel("s(n)")

# display the plot
plt.show()
```

Does this plot look like a familiar function? Based on the plot, does the sequence of partial sums appear to be converging or diverging?

Python alone does not have the tools to determine if the sum converges exactly. The `Sympy` library has the tools needed to do this, and gives Python the same capabilities as Mathematica

```
## Again, only have imports at the top of a python file
# this imports the symbolic k, which can be used as a variable
from sympy.abc import k
# import summations, and oo is infinity
from sympy import Sum, oo

a_k = 1 / k
# create an expression for the sum, notice the capital "S" in "Sum"
sum_expression = Sum(a_k, (k, 1, oo))
# doit() actually performs the operation through computation
print(sum_expression.doit())
```

which gives us "oo," indicating the series goes to infinity, and therefore does not converge.

To see everything put together, see the python file `template.py`. To install packages in the Thonny IDE, see https://www.youtube.com/watch?v=Oo-B98WWre_8.

Lab Questions

Question 1(a)

For each of the following series, perform the steps outlined below.

$$\sum_{k=1}^{\infty} \frac{1}{k^2}, \qquad \sum_{k=1}^{\infty} \left(\frac{\ln(k)}{k}\right)^2, \qquad \sum_{k=0}^{\infty} \frac{1}{k!}$$

i) Define a sequence of partial sums.

ii) Make an abridged table of values.

iii) (List)Plot the partial sums.

iv) Evaluate the sum of the series.

v) Considering the sequence of terms a_k so that the series is $\sum_{k}^{\infty} a_k$, determine if a_k converges or diverges, and if it converges, find its limit. (Note: you can answer this question by your result in the previous step! Why?)

Question 1(b)

Take your best guess at these questions (no right or wrong answers here!).

i) Looking at the plots and lists, which series appears to be converging most quickly?

ii) Which one appears to be converging most slowly?

Question 2

Let us look more carefully at the last series in Question (1). You should have found that the series converged to $e \approx 2.718$. This is a well-known series we will revisit later in the semester. Use the sample code below as a starting point to answer the questions.

```
t[n_] := (* write the code for the partial sum of the third series here *)
      errors = {};
      exact = (* write the exact value the series converges to here *)
      For[
      m = 1,
      m <= 10,
      m++,
      errors = Append[errors, Abs[t[m] - exact]]
      ]
      (* Output the table of error terms here--use approximate (decimal) values
   *)
```

i) Modify the sample code. It helps you define an empty list called **errors** and run an experiment for values of n ranging from 1 to 10.

   Recall: In Mathematica comments look like (* this is a comment *) and you will need to replace them with appropriate code.

   When you are finished, **errors** should contain 10 numbers, representing the difference between the nth partial sum and e.

ii) Use the command **ListLogLogPlot** to plot the errors on a $\log - \log$ plot. What do you notice?

iii) Does evaluating the partial sums seem to be a good way to find an approximation to e?

Question 3

i) Repeat Question 2 with the series

$$\sum_{k=1}^{\infty} \frac{1}{k^2}.$$

   The sum of this series is known to be $\dfrac{\pi^2}{6}$, so this should be used as your value for **exact**.

   It actually takes a lot of effort to prove the series converges to this value (you can read about the Basel Problem on Wikipedia if you are interested—and it illustrates why you are only asked to determine convergence or divergence for most series, not their limits).

ii) Does this series converge as quickly as the one in Question 2? Make sure your sum starts at $k = 1$ instead of $k = 0$.