# MECH2700 Engineering Analysis I, 2015
# Exercise Sheet 1

## Approximating $\pi$ using a series expansion

This exercise is designed to work through the process of editing and running two Python programs using the Engineering Faculty computers that run MS-Windows. It is assumed that you have logged into your lab computer and that you have access to a **H:** drive which is the your storage area on the file server. You can do the same on your own computer if you download an install a suitable Python3 programming package. Presently, `Anaconda` from Continuum Analytics looks to be a good, all-in-one package.

### Python as a calculator

First, we will use the Python environment as a calculator and attempt to estimate the numerical value of $\pi$ using only simple arithmetic operations.The numerical value of $\pi$ is determined using a Taylor series based on inverse trigonometric or trigonometric formulae. Machin's formula is one of the more famous ways of calculating $\pi$.

- Research and state Machin's formula in your workbook. Good sources include `www.wikipedia.org` or `www.google.com.au`. Be sure to state your source(s).
- State the Taylor (or Maclaurin) series for $atan(x)$.
- Write down an approximation for $\pi$ using Machin's formula and approximating each $atan$ by the first three terms of its series.

The following example demonstrates a program that uses Machin's formula with the arctangent approximated by its first three terms (This is what you wrote out on the previous page).

Step 1. Make a **mech2700** directory in your **H:** drive. This will be your working directory for the exercise.

Step 2. Start the *IDLE3* programming environment and you should be greeted with an interactive session containing something like the following:

```
Python 3.4.0 (default, Apr 11 2014, 13:05:18)
[GCC 4.8.2] on linux
Type "copyright", "credits" or "license()" for more information.
>>>
```

The triple-angle-bracket prompt at the bottom of the text indicates that the Python interpreter is waiting for your input.

Step 3. Instead of giving the Python interpreter commands directly, open a new editing window using the menu entry *File→New Window* and enter the code listed here. Remember to customise the introductory string to reflect your name and the date that you actually typed the code. If you save the text with the menu command *File→Save as...* and supply the file name as **h:\mech2700\machin1.py**, the edit window will be labelled (in the top bar) with that file name and the content will be syntax-highlighted in colour. Note that, depending on network conditions, it may be faster to do all of your work on drive `C:` which is the local hard-disk. The `C:` drive will be wiped clean each night so you should move your work files to network drive `H:` at the end of your session.

```
# file: machin1.py
"""
MECH2700: Using Python as a calculator.

Compute and display an estimate of pi using Machin's formula.
Use only three terms in the arctangent expansions.
Check against another estimate obtained via the math library.

P. A. Jacobs
School of Engineering, UQ.

2014-07-08 Python3 compatibility.
"""

from math import *

print("Begin program Machin1...")
a1 = 1.0 / 5
a2 = pow(a1,3) / 3
a3 = pow(a1,5) / 5
b1 = 1.0 / 239
b2 = pow(b1,3) / 3
b3 = pow(b1,5) / 5
pi_1 = 4 * (4 * (a1 - a2 + a3) - (b1 - b2 + b3))
print("Truncated series estimate is ", pi_1)
pi_2 = 4.0 * atan(1.0)
print("Math library estimate is     ", pi_2)
print("Difference is                ", pi_1 - pi_2)
print("End of program Machin1.")
```

Step 4. Run your Python script using the *Run→Run module* menu entry in the edit window and copy (or paste) the result of the original interpreter session into your workbook.

Comment on the accuracy of the approximation of $\pi$ to the value in the math library. How would you increase the accuracy of the approximation?

## Programming Python

We will now generalise the calculation by allowing the arc-tangent series to be truncated at some number of terms that will be specified at run-time rather than at program design time.

If you are given a function **atanseries(**$x$, $n$**)**, which approximates **arctan(**$x$**)** using **n** terms of the arctangent Taylor series, write out an approximation for $\pi$ using Machin's formula and the **atanseries(**$x$, $n$**)** using **k** terms of the arctangent series.

$\pi \approx$ ____________________________________________

Step 1. Create a new file `machin2.py` and enter something like the following code:

```
# file: machin2.py
"""
Sample Python program for MECH2700

Computes and displays an estimate of pi using Machin's
and a series expansion for the arctangent function.
Check against another estimate obtained via the math library.

P. A. Jacobs
School of Engineering, UQ.

2014-07-08 Python3 compatibility.
"""

import math

#------------------------------------------------------------------
def atan_series(x, n):
    "Computes atan(x) with a truncated series expansion of n terms."
    xpower = x
    my_sum = x
    sign = 1
    for i in range(1,n):
        xpower = xpower * x * x
        sign = -1 * sign
        term = sign * xpower / (2 * i + 1)
        my_sum = my_sum + term
    return my_sum

def machin(n):
    "Computes pi using Machin's formula."
    return 4 * (4 * atan_series(0.2,n) - atan_series(1.0/239,n))

#-----------------------------------------------------------------
print("Begin program Machin2...")
text = input("Enter a value for the number of terms: ")
i = int(text)
pi_1 = machin(i)
print("Truncated series estimate is ", pi_1, " for n=", i)
pi_2 = 4.0 * math.atan(1.0)
print("Math library estimate is     ", pi_2)
print("Difference is                ", pi_1 - pi_2)
print("End of program Machin2.")
```

Be careful to get the indentation of the lines exactly as shown; the block structure of the code is indicated to the interpreter via this indentation. Each indent level here is 4 characters. IDLE3 will handle the details for you if you press *TAB* while the cursor is at the start of the line. Note that the *BACKSPACE* key will take you back an indent level of 4 spaces.

Step 2. Execute the program mentioned in the previous section. This time, you will need to supply an integer value at the program's prompt.

Step 3. Print the output. This can be done via the *File→Print window* menu entry. You may also save the window's contents. Paste or copy the windows content into your workbook.

## Notes and Questions

1. Python allows us to inspect its namespace. In the interpreter window, try the command `dir()` to see the list of named objects that are visible at the top level. After running a module as we did above for `machin2.py`, You should see name of the function 'machin' as one of the items in the list. Try the command `help(machin)` to see what information is available. Paste or copy the output into your workbook.

2. In the math library there is a function `fabs` and a constant `pi`. Using the help, write down the syntax of the function `fabs`

3. Construct in one line a python command that will return the error in the approximation of $\pi$, using `math.fabs` and `math.pi` and the function `machin` from `machin2.py`.

4. Complete the following table using the python command from the previous question. (Of course, do this in your workbook.)

Table 1: Approximation of $\pi$ using Machin's formula

| No of Terms | 3 | 5 | 7 | 9 | 11 | 13 | 15 |
|---|---|---|---|---|---|---|---|
| Error | | | | | | | |

5. Machin's formula is not the only way of approximating $\pi$, but has better performance than other methods. This question will compare it to another approximation for $\pi$, that uses arctangent directly.

$$\pi = 4 \arctan(1.0) \tag{1}$$

Fill out the following table of errors directly using `4*arctan_series(1.0, n)` as the approximation for $\pi$, where `n` is the number of terms to keep in the expansion. Remember that error can be defined as $|\pi_{approx} - \pi|$.

Table 2: Approximation of $\pi$ using an alternative formula

| No of Terms | 3 | 5 | 7 | 9 | 11 | 13 | 15 |
|---|---|---|---|---|---|---|---|
| Error | | | | | | | |