A Mathematician's Cookbook:
How to Make the Perfect Pi

Oliver J Baynham

23 / 02 / 1996

1 What is Pi?

The number π is defined as the ratio of a circle's diameter to its circumference, and has been the forefront of much Mathematical awe and frustration for the past 3000 years; we Mathematicians stand humbly at the seemingly impossible probability of *naturally occurring infinity*.
Perhaps our fascination derives from the subconscious longing to answer that ever lingering question:
"Is Mathematics of human construct, or human discovery?"

2 Calculating Pi

Since ancient times, humans have deduced methods of approximating π in order to be used for conventional Mathematics. Starting with the Babylonians taking $\pi = 3$ for architectural projects of the time, and moving to more complex rationals such as Archimedes' 223/71, it wasn't until the early 17^{th} century when an exact formula was first discovered.

2.1 Gregory-Leibniz Series

The Gregory-Leibniz Series [1] is built on the Taylor expansion of $tan^{-1}(1)$, such that:

$$\frac{\pi}{4} = \sum_{n=0}^{\infty} \frac{(-1)^n}{2n+1} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + ...$$

This infinite sum, if calculated to infinity, will give us $\frac{\pi}{4}$ exactly. Unfortunately, calculating a sum to infinity is **impossible** (at least, for mere mortals), but by calculating to the nth term we can still obtain a reasonable approximation for π. I have attempted to model this with this code using Python in SageMathCloud.

```
def Leibniz(n):
        """
    A simple function to simulate a summation formula modeled on the Gregory-Leibniz Series.
    Argument is an integer (number of iterations).
    Output is a float.
    """
    k = var('k')
    a = sum(((-1) ^ (k)) / ((2 * k) + 1.0), k, 0, n)
    return a * 4
```

Upon setting n to be 1000, Sage returns 3.14259165434, which we can see is equal to pi to exactly 2 decimal places, with an absolute difference of roughly 0.000999. When $n = 100000$, Sage returns 3.14160265348, which is an even closer approximation, showing that the higher the number of iterations, the more accuracy in approximating. The following Sage code calculates the absolute difference between pi and the approximation for iterations between 500 and 10000 and writes the values to a graph.

```
listofvalues = []                       #Defining an empty list
for e in range(500, 10000):
#A for loop to calculate the absolute difference 9500 times for increasing iterations
    value = abs(Leibniz(e) - pi)
    listofvalues.append([e, value])

list_plot(listofvalues, axes_labels = ['Iterations', 'Absolute difference'])
```

As we can see in Figure 1 (overleaf), the difference between the approximation and the actual value of pi grows smaller with higher numbers of iterations. This function does, however, converge very slowly, and becomes slower the higher the number of iterations it must compute. Realistically we would need to compute 5 million iterations to calculate a mere 7 correct decimal places of pi [1], which with my current resources would take a stupidly long amount of time to do. This is why we must look at other, more precise methods of approximation.

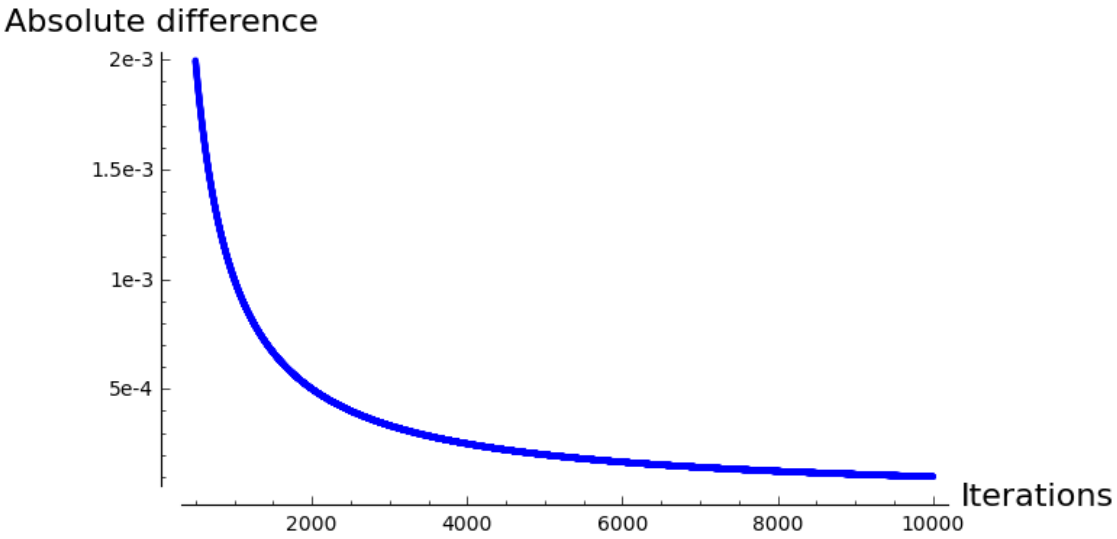

Figure 1: Absolute difference between Leibniz approximation and pi against number of iterations

2.2 Chudnovsky Algorithm

The Chudnovsky brothers are a dynamic duo born in the mid 20^{th} century, known for their incomparable breakthroughs in Mathematical computation and calculation involving home-built super computers and complex algorithms. Many of these algorithms were formulas for π, discovered after yearning helplessly to find some organisation in the irregularity of its decimal places, [2] which won them world records towards the end of the century. One of these such algorithms is still used to calculate further decimal places of pi, and this is the *Chudnovsky Algorithm*.

$$\frac{1}{\pi} = 12\sum_{n=0}^{\infty}\frac{(-1)^n(6n)!(13591409+545140134n)}{(3n)!(n!)^3 640320^{3k+3/2}}$$

Once again this formula is based on an infinite sum, but by simulating it in Python with this code we can gain a reasonably accurate approximation for π by taking the nth term.

```
R = RealField(300)
#Using the RealField, I can choose the number of decimal points for my answers

def Chudnovsky(n):
    """
    A function to simulate the Chudnovsky algorithm for pi approximation, consisting mainly of
    a sum from 0 to the iteration argument n.

    Argument n is an integer.
    Output will be a float
    """
    k = var('k')
    x = ((-1)**k * factorial(6*k) * (13591409 + 545140134 * k))
    y = (factorial(3*k) * (factorial(k))**3 * (640320 ** (3 * k + 3/2)))
    a = R(sum(x / y, k, 0, n))
    return R(1 / (12 * a))
```

Note in the function how, for clarity, I have separated the numerator and denominator of the algorithm into variables x and y; given the variable k, this kind of algebraic manipulation would not work on many other Python editors - hence Sage is far superior when it comes to coding the more complex algorithms. Now, if I set $n = 0$, we get a rather surprising result: 3.141592653589734207668 4..., which we can see is a correct approximation for pi to ***13 decimal places***. This means that the function gives a better approximation of pi with no iterations than the Gregory-Leibniz Series in section 2.1 would give with over 5 million iterations.
In comparison, with $n = 1$, the approximation is correct to 27 decimal places. When $n = 2$, that number is 41, and so on, and so forth. In fact, since the accuracy increases by so much with each singular operation, drawing a graph like in Figure 1 is near impossible. Cependant, by working out the absolute difference between each approximation and the actual value of pi, I have written this code to determine the number of correct decimal places, as seen in the table below.

Iterations	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Decimal Places	13	27	41	55	70	84	98	112	127	141	155	169	183	198	212	226	240	254

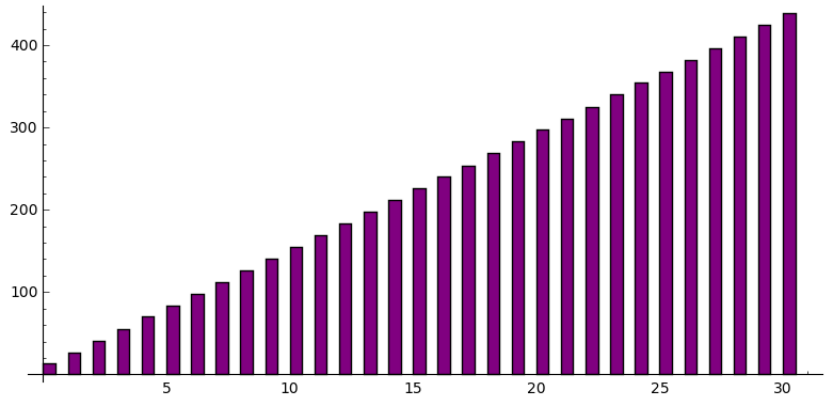

Figure 2: Number of correct decimal places against number of iterations

As a more visual interpretation, Figure 2 to the left shows how the number of correct decimal places of each approximation increases in almost perfect uniform as the number of iterations is increased. The chart makes it easier to see how the algorithm won 4 world records between 2009 and 2013 for approximating pi, the last of which being to 12.1 trillion decimal places.
But by now this may have got you thinking, *what does this all mean*? What is pi but an incomprehensibly long list of random numbers, continuing forever into the depressingly confusing yet surprisingly intriguing nothingness of infinity?
Perhaps if there were some way to write code that could visualise this monstrosity of a mathematical constant...

3 But Seriously: What is Pi?

"*It's a moon,*
A wheel revolving on golden earth, and lotus blossoms.
Mountains embrace windmills, and it all reflects this number, pi." [3]

A Piku *(a poem whose line's syllables are 3-14-15)* in Pilish *(words whose letters correspond to the digits of pi)* shows just how fascinated some people are by the beauty that lies in certain Mathematical areas. To embrace this, I have attempted to write some Python code that uses the Turtle library to visualise our favourite constant. After importing pi to 1000 decimal places, I converted each digit to an integer in a list, and used this to draw pictures. The full code can be found here, but the most important part is the following for loop that draws the circle.

```
for i in range(1001):
    color = colorsys.hsv_to_rgb(i / 100.1 * 0.1, 1.0, 1.0)
    t.color(color)              #Changes the pen colour - colorsys iterates 1000 times to
    t.rt(90)                    #give all colours of the rainbow.
    t.pendown()
    if data[i] != 0:            #Turtle moves forward only if digit is not 0.
        t.fd(data[i] * 36.9)
    t.penup()                   #Turtle does not draw while moving back on itself.
    t.bk(data[i] * 36.9)
    t.lt(90)                    #Turtle turns and moves forward on circle.
    t.fd((666 * math.pi) / 1001.0)
    t.rt(360 / 1001.0)
```

I am afraid to admit that this ends our brief journey into the whats, hows, and whys of the most tempestuous, scintillating, babelicious, unfaltering Mathematical constant, π - and thus I leave you with this short witticism: Whenever you are feeling down in the dumps, just remember that even though life may be ephemeral, blessed be that π is not.

References

[1] Crypto-Stanford. Gregory-Leibniz Series. https://crypto.stanford.edu/pbc/notes/pi/glseries.html.

[2] Boris Gourvitch. The World of Pi - Chudnovsky Brothers. http://www.pi314.net/eng/chudnovsky.php.

[3] Museum of Mathematics. 2014 Pi Day Competition Winners. http://momath.org/home/2014-pi-day-winners/.