Primorials in Pi

Jabari Zakiya

Email: jzakiya@gmail.com

Abstract: Since at least 1734 (when Euler solved the Basel problem), it's been known for positive even integers s, Zeta(s) can be written in terms of the even powers of Pi. I manipulate the Euler Zeta function form and find lurking (hidden) in it an exquisite and elegant formula for Pi. Thus, not only does the Euler Zeta function have embedded in it Pi, Pi has embedded in its construction the primorials of primes.

Introduction

For most people π , i.e. 3.14159…, is the most well known math constant they can recite to at least a few digits. There are many algorithms [6] that can generate its digits, with varying speed. Using ***Prime Generator Theory (PGT)*** we can derive an exquisite formula for its computation that's been hiding in plain sight (for centuries) that heretofore hadn't been noticed, missed by even the great Euler, who probably had the first chance (best mindset) to notice it, but didn't. And its starts with his Zeta function.

Zeta function $\zeta(s)$

In contemporary math the Euler/Riemann Zeta function expression is usually written in this form:

$$\zeta(s) = \sum_{n=1}^{\infty} n^{-s} = \prod_{p} \frac{1}{1 - p^{-s}}$$

But Euler wrote it like this:

$$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s} = \prod_{p} \frac{p^s}{p^s - 1}$$

Written in primorial form it's:

$$\zeta(s) = \prod_{p} \frac{p^s}{p^s - 1} = \frac{p_n^s\#}{(p_n^s - 1)\#}$$

For *s* = 2 we get:

$$\zeta(2) = \frac{p_n^2\#}{(p_n^2 - 1)\#}$$

But $\zeta(2) = \pi^2/6$, and $p_n^2\#$ is $(p_n\#)^2$, which now gives us this exquisite formula for π.

$$\frac{\pi^2}{6} = \frac{(p_n\#)^2}{(p_n^2 - 1)\#}$$

$$\pi = \frac{\sqrt{6}\; p_n\#}{(\sqrt{p_n^2 - 1})\#} = (3\#)^{1/2} \frac{p_n\#}{(p_n^2 - 1)^{1/2}\#}$$

And now we see a simple formula for π ***hidden in the background of the Zeta function!*** We *see* we can represent (and calculate) π strictly with primorials, i.e. *consecutive prime factors*. We'll further see not only does π lurk within the $\zeta(2k)$ values, but the primorials lurk within the construction of π.

But we don't have to stop with $\zeta(2)$, as each expression for $\zeta(2k)$ has a factor of π^{2k} in it.

For $s = 2k$:

$$\zeta(2k) = (-1)^{k+1}\frac{B_{2k}2^{2k}}{2(2k)!}\pi^{2k}$$

The B_{2k} are the $2k$-th Bernoulli numbers. Here are the first 8 expressions for $\zeta(2k)$ [5].

$$\zeta(2) = \frac{\pi^2}{6} \qquad \zeta(4) = \frac{\pi^4}{90} \qquad \zeta(6) = \frac{\pi^6}{945} \qquad \zeta(8) = \frac{\pi^8}{9450}$$

$$\zeta(10) = \frac{\pi^{10}}{93555} \qquad \zeta(12) = \frac{691\pi^{12}}{638512875} \qquad \zeta(14) = \frac{2\pi^{14}}{18243225} \qquad \zeta(16) = \frac{3617\pi^{16}}{325641566250}$$

I'll show we can compute π to increasing accuracy with primorials, using its generalized form:

$$\pi = C_{z2k}^{1/2k}\prod_p \frac{p_n}{(p_n^{2k}-1)^{1/2k}} = C_{z2k}^{1/2k}\frac{p_n\#}{(p_n^{2k}-1)^{1/2k}\#}$$

where the C_{z2k} are the constant rational inverse coefficients of π^{2k} from the $\zeta(2k)$ expressions.

$$C_{z2} = \frac{6}{1} = 6 \qquad C_{z4} = \frac{90}{1} = 90 \qquad C_{z6} = \frac{945}{1} = 945 \qquad C_{z8} = \frac{9450}{1} = 9450$$

$$C_{z10} = \frac{93555}{1} = 93555 \qquad C_{z12} = \frac{638512875}{691} \qquad C_{z14} = \frac{18243225}{2} \qquad C_{z16} = \frac{325641566250}{3617}$$

With the C_{z2k} having form:

$$C_{z2k} = (-1)^{k+1}\frac{(2k)!}{2^{2k-1}B_{2k}}$$

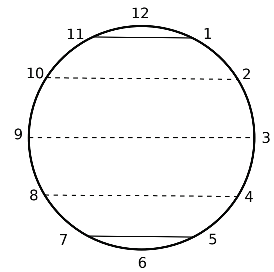
What we will *discover* is that the C_{z2k} coefficients have embedded within them the value of π, to increasing digits of accuracy. From their starting approximations for π, the primorials boost the number of accurate digits higher, until they max out their capacity to create more. We'll also *discover* that from the numerator factorization of the C_{z2k} we can reconstruct their representations as factors of primorials.

Geometric Interpretation using PGT

Let's see how to geometrically understand this conceptually, from the perspective of *PGT*.

As explained in [1], [2], [3] *Prime Generators* break the number line into modular groups of size $p_n\#$ integers, which contain $(p_n{-}1)\#$ integer residues, along which all the primes not a factor of $p_n\#$ exist. As we increase the modular group size by p_n we increase the number of residues by $(p_n - 1)$. This has the effect of squeezing the primes into a smaller and smaller percentage of the integer number space. It's essentially the same process Euler used to squeeze out all the composites in the reciprocal integer form of the Zeta function to create his multiplicative prime (primorial) form.

Useful for our purposes here, we can model the periodicity of the modular groups with a clock.



Using our generator clock model we can conceptualize the geometric meaning of the expression for π.

$$\pi = C_{z2k}^{1/2k} \frac{p_n\#}{(p_n^{2k}-1)^{1/2k}\#}$$

From geometry: $$\pi = \frac{c}{d} = \frac{c}{2r}$$

where *r* = *c*/2π = *c*/τ, with (tau) τ = 2π. Thus when we take generators of length $p_n\#$ integers, and fold them into, and model them as clocks (modular circles), c = $p_n\#$ is the circumference of these circles, which increase by factors of p_n for each larger generator. Thus we get these geometric relationships:

$$c = p_n\# \qquad d = \frac{(p_n^{2k}-1)^{1/2k}\#}{C_{z2k}^{1/2k}} \qquad r = \frac{(p_n^{2k}-1)^{1/2k}\#}{2C_{z2k}^{1/2k}}$$

$$c^{2k} = p_n^{2k}\# \qquad d^{2k} = \frac{(p_n^{2k}-1)\#}{C_{z2k}} \qquad r^{2k} = \frac{(p_n^{2k}-1)\#}{2^{2k}C_{z2k}}$$

Thus we see the modular diameters and radii expressions are the (principal) 2*k*-th roots of primorial expressions. Thinking about this more extensively, this suggests there may be complex roots, which we know come as ***complex conjugate pairs***. This would be consistent with the fact that the generator residues come as ***modular complement pairs***. We'll also see for the p_n, d_n ~ p_n/π_{z2k} and r_n ~ p_n/τ_{z2k}.

I've only scratched the surface here, but I'll suspend going further down this rabbit hole of analysis, as it's diverging from the principal purpose of this paper. However, it presents itself as an interesting area of math to explore and develop, and I encourage others to vigorously pursue it if desired.

Numerical Analysis

Compared to other methods for generating π, the presented method is much simpler to understand and remember. And from a Number Theory point of view, it also has a conceptual and numerically pleasing elegance, which I will show and explain. To demonstrate its utility and performance, I provide Ruby code to generate some results of its accuracy and convergence speed for the first few C_{z2k} coefficients.

From this form of the formula: $$\pi = C_{z2k}^{1/2k} \prod_p \frac{p_n}{(p_n^{2k}-1)^{1/2k}}$$

We expand it into: $$\pi = C_{z2k}^{1/2k} \cdot \frac{2}{(2^{2k}-1)^{1/2k}} \cdot \frac{3}{(3^{2k}-1)^{1/2k}} \cdot \frac{5}{(5^{2k}-1)^{1/2k}} \cdots$$

In fact, this is the form of the algorithm the Ruby code uses to numerically compute it.

Notice in the factors $(p_n^{2k}-1)^{1/2k}$ we're raising each p_n to a power $2k$, then bringing one less than that value back down to be almost (but less than) p_n. Using $p_2 = 3$ as an example, we can see the process.

$$(3^2-1)^{1/2} = (9-1)^{1/2} = 8^{1/2} = 2.82842...$$
$$(3^4-1)^{1/4} = (81-1)^{1/4} = 80^{1/4} = 2.990697...$$
$$(3^6-1)^{1/6} = (729-1)^{1/6} = 728^{1/6} = 2.99931...$$
$$(3^8-1)^{1/8} = (6561-1)^{1/8} = 6560^{1/8} = 2.99994...$$

As $2k$ increases $(p_n^{2k}-1)^{1/2k}$ becomes increasingly closer to p_n, If we set p_{n-} to be $(p_n^{2k}-1)^{1/2k}$ then the primorial ratios p_n/p_{n-} are always > 1 but can be made arbitrarily close to 1, as $2k \to \infty$.

Thus as $2k \to \infty$: $$\prod_p \frac{p_n}{p_{n-}} = \frac{2}{1.999...} \cdot \frac{3}{2.999...} \cdot \frac{5}{4.999...} \cdot \frac{7}{6.999...} \cdots \to 1.0000...$$

So if the primorial ratios are marching in unison toward 1 where do we get π from? Well, there's only one place left its digits can come from. And this is what we ***discover***, apparently missed by even Euler.

$$C_{z2}^{1/2} = 6^{1/2} = 2.449489...$$
$$C_{z4}^{1/4} = 90^{1/4} = 3.080070...$$
$$C_{z6}^{1/6} = 945^{1/6} = 3.132602...$$
$$C_{z8}^{1/8} = 9450^{1/8} = 3.139995...$$
$$C_{z10}^{1/10} = 93555^{1/10} = 3.141280...$$
$$C_{z12}^{1/12} = (638512875/691)^{1/12} = 3.141528...$$

We can *theoretically* get arbitrary convergence with a few (or just $p_1 = 2$) primes. However in the real world, at least with using personal computers, calculators, etc, we will soon hit the wall in reaching the limit on the number of digits floating point implementations can accurately represent. But that is an implementation issue true for all numerical (floating point) operations performing computations with small numbers. However, there are numerical implementations and hardware that address these issues.

Below is Ruby code to generate π to 15 digits (when capable) using the coefficients for $C_{z2} - C_{z16}$.

```
Code:
require "primes/utils"          # Load primes-utils RubyGem
                                 # To install on system do: $ gem install primes-utils
def pi_Z2k(k2, cz2k, primes)
  pi, exp = 1.0, 1.0/k2
  primes.each do |p|
    pi *= p / (p**k2 - 1)**exp
  end
  pi * cz2k**exp
end

# Example inputs for Zeta(8)
nth = 18                        # Select number of primes to use
nth_prime = nth.nthprime        # Set prime value of nth prime
n_primes = nth_prime.primes     # Generate array of first n primes
k2, cz2k = 8, 9450              # Set Zeta(8) parameters

puts "\nUsing #{nth} primes up to #{nth_prime}"
pi = pi_Z2k(k2, cz2k, n_primes)                    # Using 18 primes uo to 61
puts "pi_Z#{k2} = #{pi} \n"                        # pi_Z8 = 3.141592653589792
```

This table shows the speed of convergence up to `pi_Z16`. On my laptop using Ruby, I was able to get up to 15 significant digits of accuracy until the fractions got too small to generate more accurate digits.

Pi digits	pi_Z2	pi_Z4	pi_Z6	pi_Z8	pi_Z10	pi_Z12	pi_Z14	pi_Z16
	m primes	m primes	m primes	m primes	m primes	m primes	m primes	m primes
3.	2							
3.1	5	1						
3.14	38		1					
3.141	76	3						
3.1415	301	5	2	1	1			
3.14159	516	10		2				
3.141592	16,663	14	4	3		1		
3.1415926	142,215	26	6		2		1	1
3.14159265	1,534,367	51	9	4	3	2		
3.141592653		80	11	5				
3.1415926535		132	15	6	4	3	2	
3.14159265358		240	21	8	5			2
3.141592653589		481	30	10	6	4	3	
3.1415926535897		837	40	13	7	5		3
3.14159265358979				18		6	4	4

As there are an unending number of $\zeta(2k)$ values of this form (use [7] for more), you can theoretically generate as many accurate digits of π you'd want with just the first few primes. It will be interesting to see if this method can be implemented to generate the next record number of digits for π, or used as a standard benchmark to test the numerical accuracy and speed of super and quantum computers, etc.

Conclusion

Since at least 1734, when Euler solved the Basel problem, it's been known for positive even integers *s* the Zeta function $\zeta(s)$ can be written in terms of the even powers of π. Using *Prime Generator Theory (PGT)* as my conceptual framework, I manipulated the Euler Zeta function form to show we can create an elegant generalized formula to represent and compute π, which apparently even Euler missed. I then show the coefficients 2*k*-th roots are approximations to π, with increasing accuracy as *s* increases. They are then boosted (increased) by primorial ratios to achieve higher accuracy. I provide software, with a table of numerical results, to show this. Finally, I provide a list of the first 25 coefficients, also written in their full primorial forms. Hopefully, the approach and findings presented here can serve as a starting point for a better conceptual understanding of the odd *s* values of $\zeta(s)$, with similar revealing results.

References

[1] The Use of Prime Generators to Implement Fast Twin Primes Sieve of Zakiya (SoZ), Applications to Number Theory, and Implications to the Riemann Hypothesis, Jabari Zakiya, 2018/19
https://vixra.org/pdf/2006.0054v1.pdf
https://www.academia.edu/37952623/
The_Use_of_Prime_Generators_to_Implement_Fast_Twin_Primes_Sieve_of_Zakiya_SoZ_Applications_to_Number_Theory_and_Implications_for_the_Riemann_Hypotheses

[2] On The Infinity of Twin Primes and other K-tuples, Jabari Zakiya, 2019/20
https://vixra.org/pdf/2006.0053v2.pdf
https://www.academia.edu/41024027/On_The_Infinity_of_Twin_Primes_and_other_K_tuples

[3] (Simplest) Proof of Twin Primes and Polignac's Conjectures, Jabari Zakiya, video, 2021
https://www.youtube.com/watch?v=HCUiPknHtfY

[4] The Twin Primes Segmented Sieve of Zakiya (SSoZ) Explained, Jabari Zakiya, 2022
https://vixra.org/pdf/2206.0100v1.pdf
https://www.academia.edu/81206391/Twin_Primes_Segmented_Sieve_of_Zakiya_SSoZ_Explained

[5] Particular values of the Riemann Zeta Function
https://en.wikipedia.org/wiki/Particular_values_of_the_Riemann_zeta_function

[6] Coded different methods to Calculate PI
https://github.com/joaojcorreia/Calculate-pi

[7] WolframAlpha Compute Engine
https://www.wolframalpha.com/input/?i=mathematica

[8] primes-utils Rubygem
https://rubygems.org/gems/primes-utils

[9] PRIMES-UTILS HANDBOOK, Jabari Zakiya, 2016
https://www.academia.edu/19786419/PRIMES_UTILS_HANDBOOK

List of Primorials in Pi from C_{z2k} constants

$C_{z2} = 6 = 3\#$ $\qquad C_{z2}^{1/2} = 2.449489742783178$

$C_{z4} = 90 = \frac{3\#5\#}{2\#}$ $\qquad C_{z4}^{1/4} = 3.080070288241023$

$C_{z6} = 945 = \frac{(3\#)^2\ 7\#}{(2\#)^3}$ $\qquad C_{z6}^{1/6} = 3.132602581012435$

$C_{z8} = 9450 = \frac{3\#5\#7\#}{(2\#)^2}$ $\qquad C_{z8}^{1/8} = 3.1399951412959073$

$C_{z10} = 93555 = \frac{(3\#)^4\ 11\#}{(2\#)^5}$ $\qquad C_{z10}^{1/10} = 3.1412803693973714$

$C_{z12} = \frac{638512875}{691} = \frac{(3\#)^3\ 5\#7\#13\#}{(2\#)^6\ 691}$ $\qquad C_{z12}^{1/12} = 3.1415282368670168$

$C_{z14} = \frac{18243225}{2} = \frac{(3\#)^4\ 5\#13\#}{(2\#)^7}$ $\qquad C_{z14}^{1/14} = 3.1415789099913694$

$C_{z16} = \frac{325641566250}{3617} = \frac{(3\#)^3(5\#)^2\ 7\#17\#}{(2\#)^6\ 3617}$ $\qquad C_{z16}^{1/16} = 3.1415896529495364$

$C_{z18} = \frac{38979295480125}{43867} = \frac{(3\#)^6(7\#)^2\ 19\#}{(2\#)^9\ 43867}$ $\qquad C_{z18}^{1/18} = 3.1415919871238964$

$C_{z20} = \frac{1531329465290625}{174611} = \frac{(3\#)^4(5\#)^3\ 11\#19\#}{(2\#)^9\ 175611}$ $\qquad C_{z20}^{1/20} = 3.1415925037418626$

$C_{z22} = \frac{13447856940643125}{155366} = \frac{(3\#)^6\ 5\#(7\#)^2\ 23\#}{(2\#)^{10}\ 155366}$ $\qquad C_{z22}^{1/22} = 3.1415926195391455$

$C_{z24} = \frac{201919571963756521875}{236364091} = \frac{(3\#)^6\ 5\#(7\#)^2\ 13\#23\#}{(2\#)^{11}\ 236364091}$ $\qquad C_{z24}^{1/24} = 3.1415926457870995$

$C_{z26} = \frac{11094481976030578125}{1315862} = \frac{(3\#)^5(5\#)^3\ 7\#11\#23\#}{(2\#)^{11}\ 1315862}$ $\qquad C_{z26}^{1/26} = 3.141592651789231$

$C_{z28} = \frac{564653660170076273671875}{6785560294} = \frac{(3\#)^7(5\#)^4\ 7\#13\#29\#}{(2\#)^{14}\ 678556094}$ $\qquad C_{z28}^{1/28} = 3.1415926531718115$

$$C_{z30} = \frac{5660878804669082674070015625}{6892673020804} \qquad C_{z30}^{1/30} = 3.141592653492265$$

$$= (3\#)^9\ 5\#(7\#)^2\ 11\#13\#31\#/(2\#)^{15}\ 6892673020804$$

$$C_{z32} = \frac{6249022057102234120726640625 0}{7709321041217} \qquad C_{z32}^{1/32} = 3.141592653566935$$

$$= (3\#)^7(5\#)^4(7\#)^2\ 17\#31\#/(2\#)^{14}\ 7709321041217$$

$$C_{z34} = \frac{1213045458143374858729289062 5}{151628697551} \qquad C_{z34}^{1/34} = 3.1415926535844148$$

$$= (3\#)^9(5\#)^3\ 7\#11\#13\#31\#/(2\#)^{16}\ 151628697551$$

$$C_{z36} = \frac{20777977561866588586487628662044921875}{2631527155305347 7373} \qquad C_{z36}^{1/36} = 3.141592653588523$$

$$= (3\#)^9(5\#)^3(7\#)^3\ 13\#19\#37\#/(2\#)^{18}\ 26315271553053477373$$

$$C_{z38} = \frac{2403467618492375776343276883984375}{308420411983322} \qquad C_{z38}^{1/38} = 3.1415926535894925$$

$$= (3\#)^{10}(5\#)^3(7\#)^2\ 11\#17\#37\#/(2\#)^{18}\ 308420411983322$$

$$C_{z40} = \frac{200804311722896388267984011283905566 40625}{261082718496449122051} \qquad C_{z40}^{1/40} = 3.141592653589722$$

$$= (3\#)^9(5\#)^5\ 7\#11\#13\#19\#41\#/(2\#)^{19}\ 261082718496449122051$$

$$C_{z42} = \frac{23077891898189601277125944278646674277 34375}{3040195287836141605382} \qquad C_{z42}^{1/42} = 3.1415926535897762$$

$$= (3\#)^{11}(5\#)^2(7\#)^4\ 13\#19\#43\#/(2\#)^{20}\ 3040195287836141605382$$

$$C_{z44} = \frac{37913679547025773526706908457776679169921875}{5060594468963822588186} \qquad C_{z44}^{1/44} = 3.141592653589789$$

$$= (3\#)^{10}(5\#)^4(7\#)^3\ 13\#23\#43\#/(2\#)^{20}\ 5060594468963822588186$$

$$C_{z46} = \frac{76701022144483010530333584806102125294628906 25}{10373062810328907187442 8} \qquad C_{z46}^{1/46} = 3.1415926535897922$$

$$= (3\#)^{12}(5\#)^4(7\#)^2\ 11\#13\#19\#47\#/(2\#)^{22}\ 103730628103289071874428$$

$$C_{z48} = \frac{40936486033842749965196989214788795801622866699218 75}{5609403368997817686249127547} \qquad C_{z48}^{1/48} = 3.1415926535897927$$

$$= (3\#)^{12}(5\#)^4(7\#)^3\ 13\#17\#23\#47\#/(2\#)^{23}\ 5609403368997817686249127547$$

$$C_{z50} = \frac{285258771457546764463363635252374414183254365234375}{3960457641928637185699820 2} \qquad C_{z50}^{1/50} = 3.141592653589793$$

$$= (3\#)^{13}(5\#)^2(7\#)^3(11\#)^2\ 13\#23\#47\#/(2\#)^{23}\ 39604576419286371856998202$$