Beautiful Mathematics to Innovative AI

Vincent Granville PhD, Co-Founder, BondingAI.io
vincent@bondingai.io | linkedin.com/in/vincentg/
September 2025

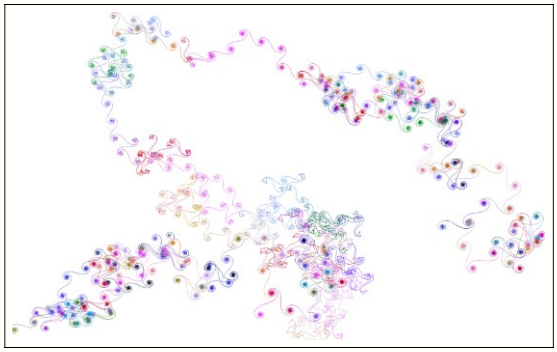
$$\int_0^\infty \frac{|\sin qx|^r - |\sin px|^r}{x}dx = \frac{1}{\sqrt{\pi}}\frac{\Gamma((1+r)/2)}{\Gamma(1+r/2)}\log\frac{q}{p}$$

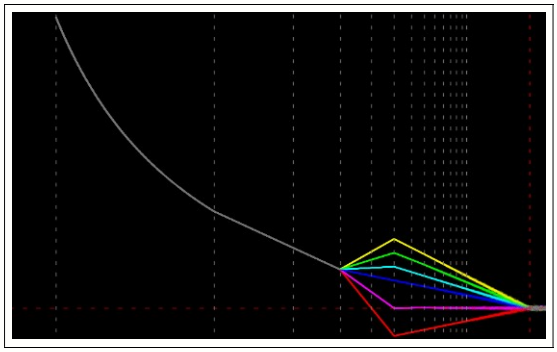
$$1 - \frac{1}{4} + \frac{1}{7} - \frac{1}{10} + \frac{1}{13} - \frac{1}{16} + \cdots = \frac{1}{3}\cdot\left(\frac{\pi}{\sqrt{3}} + \log 2\right)$$

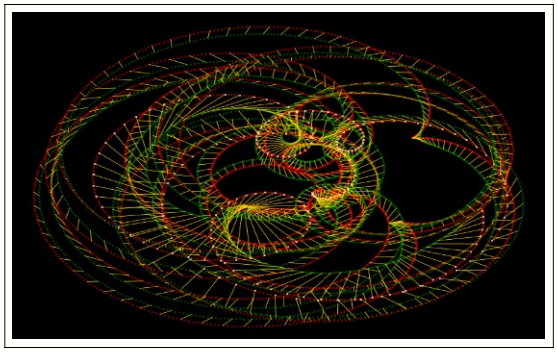
$$\lim_{n\to\infty}\left(\sum_{k=0}^{n}\frac{1}{3k+1} - \frac{\log n}{3}\right) = \frac{1}{6}\cdot\left(2\gamma + 3\log 3 + \frac{\pi}{\sqrt{3}}\right)$$

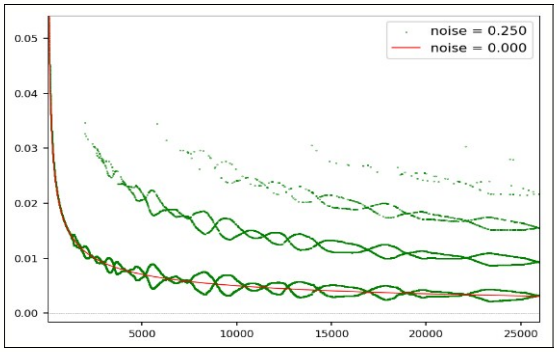
$$1 + \frac{x^3}{3!} + \frac{x^6}{6!} + \frac{x^9}{9!} + \cdots = \frac{1}{3}e^x + \frac{2}{3}e^{-x/2}\cdot\cos\left(\frac{\sqrt{3}}{2}x\right)$$

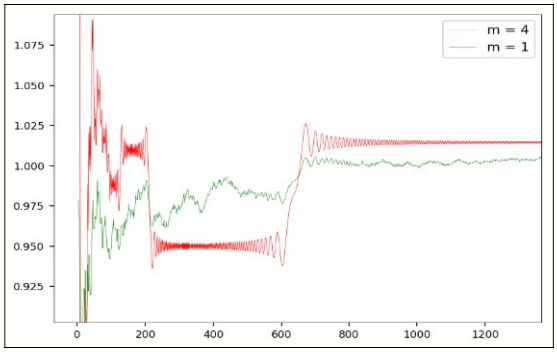
$$\frac{x}{1!} + \frac{x^4}{4!} + \frac{x^7}{7!} + \frac{x^{10}}{10!} + \cdots = \frac{1}{3}e^x - \frac{1}{3}\left[\cos\left(\frac{\sqrt{3}}{2}x\right) - \sqrt{3}\sin\left(\frac{\sqrt{3}}{2}x\right)\right]e^{-x/2}$$

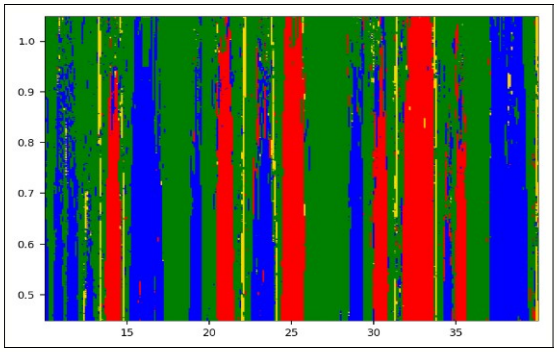


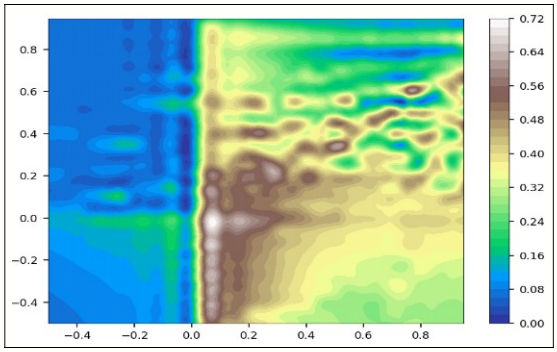












Chapter 4

Creative Problems in Number Theory

From the beginning, the Riemann Hypothesis has been strongly connected to prime numbers. Both are deeply intertwined, and all attempts to prove this famous conjecture – still unsolved today – heavily rely on analytic number theory where primes play a key role. But what if the connection was indirect, with even a stronger link to synthetic numbers that are not even integers? Could the Riemann Hypothesis be explained by something other than prime numbers? It would open new opportunities to solve this problem, based on generated numbers with the appropriate distribution, possibly moving the discussion from number theory to standard calculus and real analysis. This is the topic of section 4.1.

This chapter discusses state-of-the-art research about the Riemann zeta function, with a very innovative perspective on the subject. It also features interesting findings about twin primes and additive number theory, linked to a generalization of the Goldbach conjecture. Finally, I discuss the connection to quantum systems and sub-quantum states.

4.1 Synthetic primes, quantum states, and the Riemann Hypothesis

Using synthetic numbers – random ones, not even integers – I build a mathematical function with the exact same roots as the Riemann zeta function, on the critical line [Wiki]. This is at the core of the Riemann Hypothesis. It challenges the myth that the theory is regulated by prime numbers. In other words, you can do a deep dive on the topic, maybe even prove the conjecture, without ever discussing prime numbers. My arbitrary numbers form a synthetic multiplicative semigroup [Wiki] and have a specific distribution and properties. Yet, they do not depend on the exact values of the standard prime numbers.

The material presented here is accessible to readers with a modest mathematical background. Knowledge of advanced topics such as analytic continuation, Dirichlet series, or complex number theory, are not needed to gain a deep understanding of the mechanics behind the scenes.

4.1.1 Definitions

Let $\mathscr{P} = \{p_1, p_2, p_3 \dots\}$ be a set of strictly increasing real numbers called Beurling primes, with $p_1 = 2$. Also, let $\log p_1, \log p_2$ and so on be linearly independent over the set of rational numbers. Let $S_{\mathscr{P}}$ be the set of all product combinations

$$\omega = p_1^{a_1} p_2^{a_2} p_3^{a_3} \cdots$$

where $p_1, p_2, \cdots \in \mathscr{P}$ and a_1, a_2 and so on are positive integers, with all but a finite number of them equal to zero. I can now define the Dirichlet eta function [Wiki] attached to $\mathscr{P}$ as

$$\eta_{\mathscr{P}}(s) = \sum_{\omega \in S_{\mathscr{P}}} \delta(\omega) \cdot \frac{1}{\omega^s} = \delta(\omega_1) \cdot \frac{1}{\omega_1^s} + \delta(\omega_2) \cdot \frac{1}{\omega_2^s} + \dots \tag{4.1}$$

where $s = \sigma + it$ is a complex number. Here, $\delta(\omega) \in \{-1, +1\}$. In practice, the series (4.1) is not absolument convergent when $\sigma \leq 1$. Thus the ordering of the terms is critical. I define the order and the δ function later. For now, note that we always have $\omega_1 = 1, \omega_2 = 2, \delta(\omega_1) = 1$ and $\delta(\omega_2) = -1$. From there, one can define the associated Riemann zeta function as follows:

$$\zeta_{\mathscr{P}}(s) = \frac{\eta_{\mathscr{P}}(s)}{1 - 2^{1-s}} = \sum_{w \in S_{\mathscr{P}}} \frac{1}{\omega^s} = \prod_{p \in \mathscr{P}} \frac{1}{1 - p^{-s}} \tag{4.2}$$

where typically, both the series and Euler product [Wiki] in (4.2) diverge when $\sigma \leq 1$.

4.1.2 Building a Beurling eta function by deletion

This is the easiest case. First, you pick up primes numbers $p_{k_1}, p_{k_2}, p_{k_3}$ and so on from the set of standard odd primes $3, 5, 7, 11$ and so on. We must keep $p_1 = 2$ which is responsible for the negative terms in the resulting series (4.1); without it, all terms would be positive and the series would diverge. Then, from the standard eta function

$$\eta(s) = \sum_{k=1}^{\infty} (-1)^{k+1} \cdot \frac{1}{k^s}, \tag{4.3}$$

you remove all terms k^{-s} where k is a multiple of at least one of the p_{k_i}'s. The resulting set $\mathscr{P}$ contains all the standard primes except p_{k_1}, p_{k_2} and so on. Also, $\delta(w)$ in the resulting series (4.1) is equal to -1 if w is even, and to $+1$ otherwise. We have

$$\eta_{\mathscr{P}}(s) = \lambda_{\mathscr{P}} \cdot \eta(s), \quad \text{with } \lambda_{\mathscr{P}} = \left[\prod_i \frac{1}{1 - p_{k_i}^{-s}} \right]^{-1}. \tag{4.4}$$

One would expect that the function $\eta_{\mathscr{P}}$ has the same non-trivial roots as the standard Riemann zeta function, provided the following condition is met:

$$\sum_i \frac{1}{p_{k_i}^s} < \infty. \tag{4.5}$$

In short, condition (4.5) states that it works even if you remove infinitely mamy primes, as long as you don't remove too many of them.

4.1.3 Building a Beurling eta function by swapping

For simplicity, let us assume that we replace each standard odd prime p_k by a strictly positive real number q_k such that $\log q_1, \log q_2$ and so on are linearly independent on the set of rational numbers. Some of the q_k's (a few, none, or infinitely many) can be identical to the corresponding p_k's. You perform the swapping directly in series (4.3) to obtain (4.1), by replacing each k^{-s} in (4.3) with

$$\omega_k = \left(k \cdot \frac{q_1^{a_1} q_2^{a_2} \cdots}{p_1^{a_1} p_2^{a_2} \cdots} \right)^{-s}$$

where a_1, a_2 and so on are the *p*-adic valuations of k respectively for p_1, p_2 and so on. Again, $q_1 = p_1 = 2$ is unchanged, to guarantee convergence. The p-adic valuation of k for a prime p is the largest integer a such that p^a divides k. If k is not a multitple of p, then $a = 0$. The resulting set $\mathscr{P}$ consists of q_1, q_2 and so on while $S_{\mathscr{P}}$ contains the ω_k's. Now we have

$$\eta_{\mathscr{P}}(s) = \lambda_{\mathscr{P}} \cdot \eta(s), \quad \text{with } \lambda_{\mathscr{P}} = \prod_k \frac{1 - p_k^{-s}}{1 - q_k^{-s}} = \frac{\zeta_{\mathscr{P}}(s)}{\zeta(s)}. \tag{4.6}$$

This time, we need $0 < \lambda_{\mathscr{P}} < \infty$ for (4.6) to be valid, and to make sure that $\eta_{\mathscr{P}}$ has the same non-trivial roots as the standard Riemann zeta function. In particular, we want this to be true when $\sigma = \Re(s) = \frac{1}{2}$. Finally, this is only a brief sketch explaining the mechanism at play. In particular, the ratio of the two zeta functions on the right-hand side may be $0/0$ (undetermined) depending on s. However, there is less ambiguity if only a finite number of q_k are different from the corresponding p_k.

Also, the Dirichlet series obtained by changing the terms in (4.3) as per the algorithm discussed, switching from $\eta(s)$ to $\eta_{\mathscr{P}}(s)$ without re-ordering, must converge at the target value s. Otherwise, the results may not be valid. The proof of convergence may not be obvious in all cases.

More broadly, two Beurling eta functions, one based on a set $\mathscr{P}$ and the other one on a set $\mathscr{P}'$, have the same non-trivial roots if the ratio

$$\lambda_{\mathscr{P}|\mathscr{P}'} = \prod_k \frac{1 - {q'_k}^{-s}}{1 - q_k^{-s}} = \frac{\zeta_{\mathscr{P}}(s)}{\zeta_{\mathscr{P}'}(s)}, \quad q_k \in \mathscr{P},\ q'_k \in \mathscr{P}'$$

satisfies $0 < \lambda_{\mathscr{P}|\mathscr{P}'} < \infty$ or equivalently, $0 < \lambda_{\mathscr{P}'|\mathscr{P}} < \infty$.

I provide illustrations in sections 4.1.4 and 4.3.1. In particular, Figure 4.2 shows that $s_3 = \frac{1}{2} + 25.01085758\, i$, the 3rd zero of ζ, is also a zero of $\eta_{\mathscr{P}}$ when $\mathscr{P}$ consists of all the primes except $3, 5, 7$ (see section 4.1.2). This is true for all zeros of ζ on the critical line. Figure 4.1 shows that replacing primes by other numbers (see section 4.1.3) also preserves the zeros without adding new ones. In this example, primes have been modified as follows: $3 \mapsto 4.051$, $5 \mapsto 7.916$, $7 \mapsto 9.114$. See Part 3 of the code in section 4.1.4.

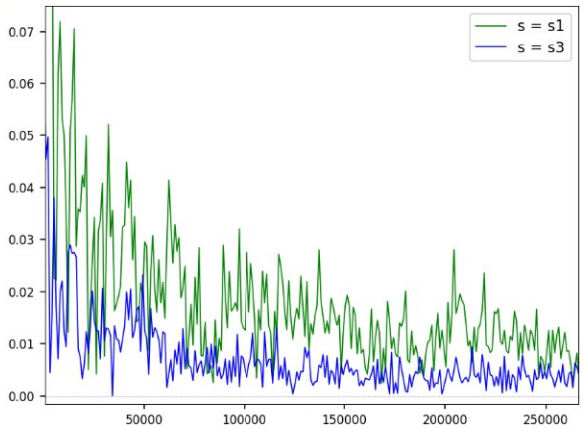


Figure 4.1: $||\eta_{\mathscr{P}}(s)|| \to 0$ as $n \to \infty$; s_1, s_3 are 1st and 3rd roots of ζ, with n on X axis and special $\mathscr{P}$.

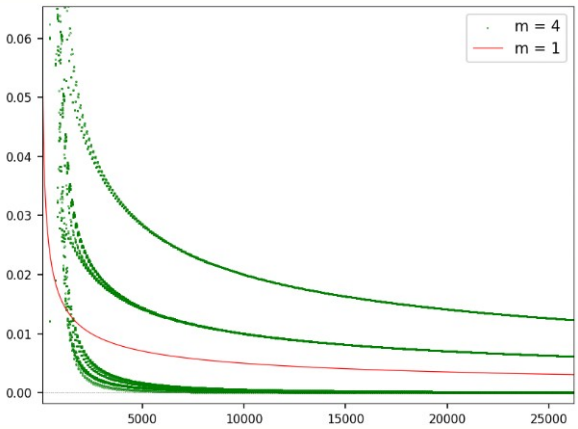


Figure 4.2: Same as figure 4.1, with a different $\mathscr{P}$ and $s = s_3$. The red curve is the standard $\eta(s_3)$.

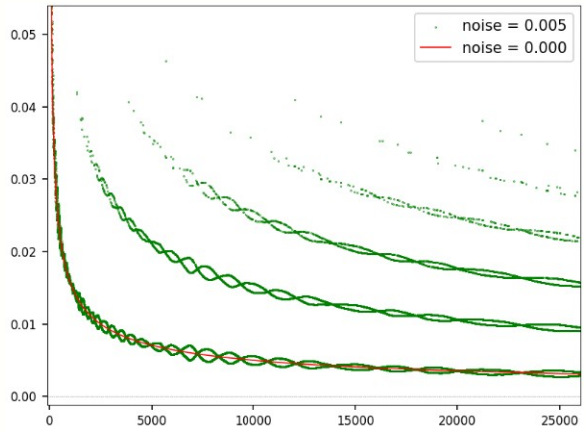


Figure 4.3: $||\eta_{\mathscr{P}}(s_3)|| \to 0$ as $n \to \infty$; p_k is replaced in $\mathscr{P}$ by $q_k = p_k + \lambda_k \cdot k^{1/4}$ if $k > 1$

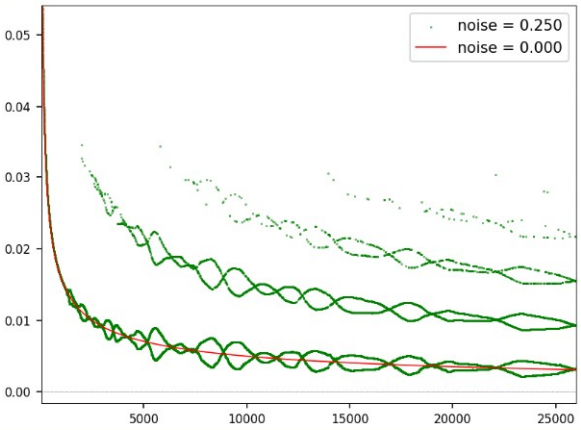


Figure 4.4: Same as Figure 4.3 but with much larger λ_k and $q_k = p_k$ if $k < 200$

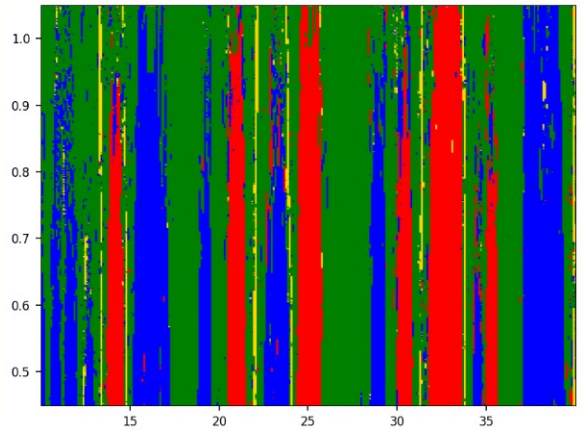


Figure 4.5: $\eta_{\mathscr{P}}(\sigma, t)$ basins of attraction, $10 < t < 40$ (X axis) and $0.45 < \sigma < 1.05$ (Y axis)

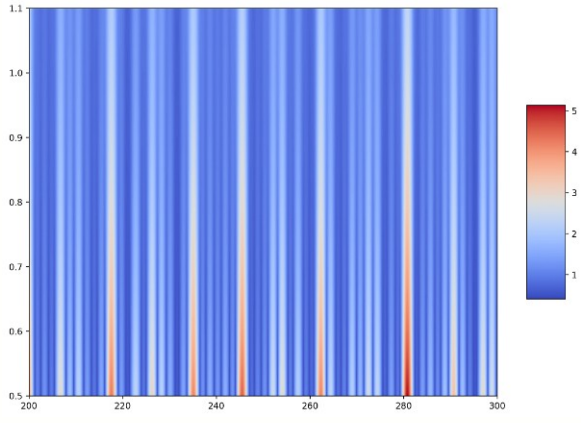


Figure 4.6: $||\zeta(\sigma, t)||$ with $200 < t < 300$ (X axis) and $0.5 < \sigma < 1.1$ (Y axis)

4.1.4 Applications and Python code

Here I discuss applications of (4.1). For a complex-valued function $\eta(s) = f(s) + ig(s)$ with complex argument s, the the notation $||\eta(s)|| = f^2(s) + g^2(s)$ stands for the norm evaluated at s. It is equal to zero if and only if

$\eta(s) = 0$, that is, if s is a root (also called zero) of η. I use the notation $s = \sigma + it$ where σ, t are respectively the real and imaginary parts of s. Also, $s_k = \sigma_k + it_k$ denotes the k-th zero of the Riemann zeta function $\zeta(s)$ with $\sigma_k = \frac{1}{2}$ and $t_k > 0$, for $k = 1, 2$ and so on. The red curve in Figures 4.2, 4.3 and 4.4 represents the standard Dirichlet eta function $\eta(s)$ evaluated at $s = s_3$ using the first n terms in series (4.3), with n on the X axis. The green and blue curves represent transformed versions of η, denoted as $\eta_{\mathscr{P}}(s)$, also supposed to have the exact same roots as $\eta(s)$ or $\zeta(s)$ at $\sigma = \frac{1}{2}$. In particular:

- In Figure 4.1, the set $\mathscr{P}$ consists of the standard primes $2, 3, 5, 7, 11$ and so on except that I replaced $p_2 = 3$ by $q_2 = 4.051$, $p_3 = 5$ by $q_3 = 7.916$, and $p_4 = 7$ by $q_4 = 9.114$. The green and blue curves correspond to $||\eta_{\mathscr{P}}(s)||$ evaluated respectively at $s = s_1$ and $s = s_3$, using the first n terms in (4.1) with n on the X axis.
- In Figure 4.2, for the green curve, the set $\mathscr{P}$ consists of all the standard primes, expect that I removed $p_2 = 3$, $p_3 = 5$ and $p_4 = 7$. Note that as n increases by one unit, $||\eta_{\mathscr{P}}(s_3)||$ (its approximation based on the first n terms) jumps from one level to another. At a given n, the current level is called the quantum state and depends on n mod 3. There are sub-levels that you can only see by zooming in, called sub-quantum states. At the lowest level (closest to the X axis), as n increases, $||\eta_{\mathscr{P}}(s_3)||$ converges to zero faster than the red curve $||\eta(s_3)||$. The word quantum convergence makes sense in this context.
- In Figure 4.3, for the green curve, $\mathscr{P} = \{q_1, q_2, \dots\}$ is a brand new set of random real numbers with $q_1 = 2$ and $q_k = p_k + \lambda_k \cdot k^{\alpha}$ for $k > 1$, where p_2, p_3 and so on are the standard primes and $\alpha = \frac{1}{4}$. Here, the λ_k's are independent uniform deviates on $[-\tau, \tau]$ with $\tau = 0.005$. The parameter τ is called the *noise*. This time, we have infinitely many quantum states though we only see the first few ones. We still have convergence of $||\eta_{\mathscr{P}}(s_3)||$ to zero as n increases.
- Figure 4.4 is identical to Figure 4.3 except that the noise threshold is $\tau = 0.25$ and $q_k = p_k$ if $k < 200$. The latter is accomplished by setting `start=200` in the code.

Convergence of the green curve $||\eta_{\mathscr{P}}(s_3)||$ to zero as n increases in Figures 4.3 and 4.4 is not a trivial problem. It seems like it requires $0 < \lambda_{\mathscr{P}} < \infty$ where $\lambda_{\mathscr{P}}$ is defined in formula (4.6). If this is correct, then we would have convergence when the following condition is met at $s = s_3 = \frac{1}{2} + it_3$:

$$\left| \frac{1}{q_k^s} - \frac{1}{p_k^s} \right| = O\Big(\frac{1}{k^{\beta}}\Big) \text{ as } k \to \infty, \tag{4.7}$$

for some constant $\beta > 1$. Here p_k is the k-th standard prime, thus $p_k \sim k \log k$. Also $q_k = p_k + \lambda_k k^{\alpha}$ with $|\lambda_k|$ bounded. Thus (4.7) is satisfied if $\alpha < \frac{1}{2}$, which is the case both in Figure 4.3 and 4.4.

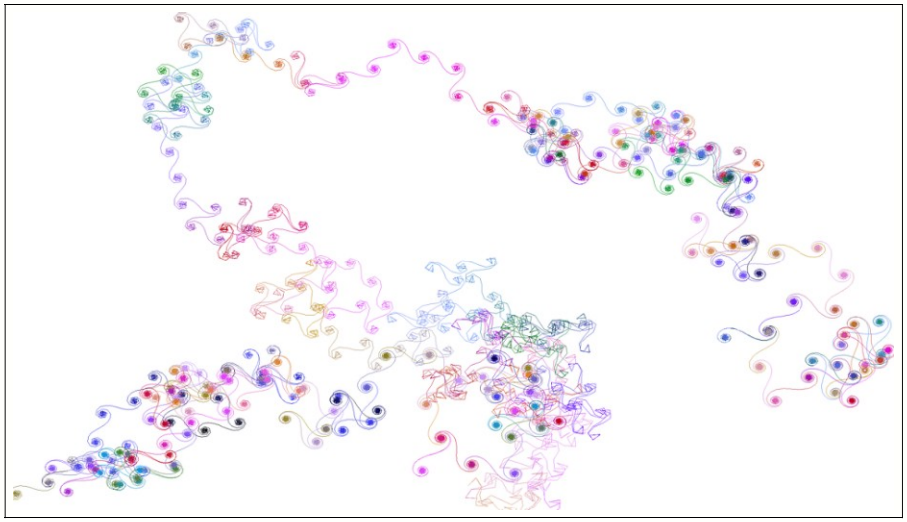


Figure 4.7: Typical convergence path for series (4.3) representing $\eta(s)$ in the complex plane, when the imaginary part t of $s = \sigma + it$ is large

This leads to an interesting question. Let $f(k)$ be a real-valued function defined on the integers $k > 0$ with $f(1) = 0$. Define $q_k = p_k + f(k)$ and $\mathscr{P} = \{q_1, q_2, \cdots\}$. Again, p_1, p_2, p_3 and so on are the standard primes. Is there a function f such that the q_k are chaos-free (unlike the standard primes) with $\eta_{\mathscr{P}}(s)$ and $\eta(s)$ having the

same zeros? By chaos-free, I mean $q_{k+1} - q_k$ smoothly increasing when k is large enough, rather than erratically going up and down infinitely many times as for $p_{k+1} - p_k$. Of course, f would have to be chaotic to remove the chaos in the standard primes. Cramér's conjecture [Wiki] states that $p_{k+1} - p_k < (\log k)^2$ for k large enough. On the other end, $f(k) = k^\alpha$ preserves the roots of η if $\alpha < \frac{1}{2}$. However we need a bigger α to counteract the largest and smallest prime gaps, thus the question remains open.

Another way to somewhat reduce the chaos in standard primes while preserving the roots of η is to choose $q_k = (p_{k-1} + p_k + p_{k+1})/3$ for $k > 200$, with $q_k = p_k$ for $k \leq 200$. The modification can be implemented in the function `build_swap_primes` in the code, along with setting `start=200`. In the code p_k is represented by `arr_primes[k]`. The resulting $||\eta_{\mathscr{P}}(s_3)||$ convergence path to zero is similar to that in Figure 4.4.

The convergence of series (4.1) representing $\eta_{\mathscr{P}}(s)$ is not a trivial topic, as shown by the green curve in Figure 4.4. It seems that the convergence of (4.3) attached to the standard eta function $\eta(s)$ is easier to establish, if you look at the red curve in the same figure. However, this is true only for values of $s = \sigma + it$ where t is small. When t is large and $0 < \sigma < 1$, the red curve also becomes quite chaotic: see Figure 4.8. This is also illustrated in Figure 4.7 where the path starts with the first term in (4.3), and each move corresponds to adding one more term in the series until convergence to $\eta(s)$. See video here showing the path evolving to $\eta(s)$ as the number n of computed terms increases, starting with two different values of s in parallel. Figure 4.7 is described in section 3.4.2 in my book on chaotic systems [12], along with the code to produce the video.

For convergence acceleration techniques applied to this example, see section 4.3 and exercise 25 in chapter 5 in [10]. Using synthetic numbers to generate $\eta_{\mathscr{P}}$ functions with desirable properties, is further discussed in chapter 17 in [11].

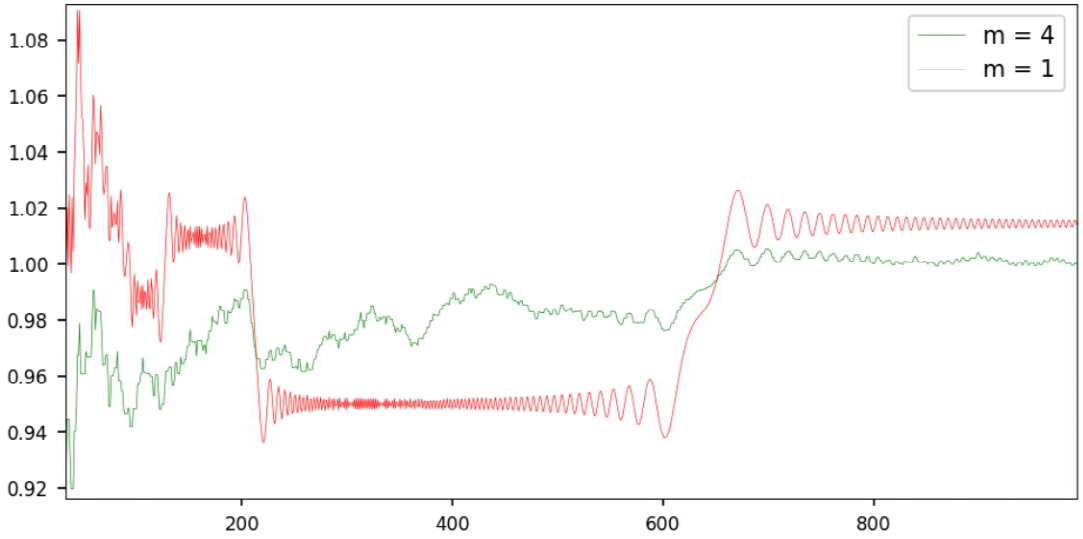


Figure 4.8: Same as Figure 4.2 with $\eta_{\mathscr{P}}(s)$ in green and $\eta(s)$ in red evaluated at $s = 0.95 + it$ with $t = 2000$, rather than at $s = s_3$. The large t causes the jumps in $\eta(s)$ as n increases on the X-axis.

The link to quantum states is further discussed in section 4.2, where I introduce quantum derivatives and remarkable approximations to $\eta(s)$. See also how sub-quantum states are involved in the binary digits of special math constants such as e, in chapter 5 and in my book "0 and 1: From Elemental Math to Quantum AI" [17]. For articles about Beurling primes, see [6, 30] and chapter 17 in [11] for additional references. Before displaying the Python code, I conclude with two interesting problems:

- **Reordering the terms** in series (4.1). When turning standard primes p_k into new numbers q_k, the terms in the resulting series may no longer be decreasing in absolute value if you keep the original ordering (alternating between positive and negative terms). You may re-order by absolute value – assuming it does not alter the value of $\eta_{\mathscr{P}}(s)$ – but then the series may no longer be alternating: see example (4.14). In the code, this is accomplished with the setting `mode='sorted'`. Still, you should keep the same ratio of positive and negative terms. Question: can you find a set $\mathscr{P}$ such that the sequence of term signs (the + and -) looks random like Bernoulli trials, after the re-ordering in question?

- **Basins of attractions**. In Figure 4.5, each pixel represents a complex number $s_* = \sigma + it$, with σ on the Y-axis and t on the X-axis. For each s_*, I solve $\eta_{\mathscr{P}}(s) = 0$ using an iterative algorithm that starts with $s = s_*$ as the initial guess for the root. Also, I use an approximation of $\eta_{\mathscr{P}}$ based on the first n terms in its series (4.1). Given s_*, the root found is denoted as $g_n(\sigma_*)$. When $n = \infty$, the roots have real part

equal either to $\frac{1}{2}$ (red dot) or 1 (green dot), according to the Riemann Hypothesis. Blue dots correspond to other roots, while the yellow color indicates lack of convergence to a solution. As n grows, the areas in blue shrink to an empty set. Each color determines a basin of attraction in the root system. Such basins are studied in the context of chaotic dynamical systems [12].

Conclusion: even if starting with s_* far away from the critical line $\sigma = \frac{1}{2}$, you can retrieve the roots on that line. Figure 4.6 confirms this fact: even on the line $\sigma = 1$, you can "feel" the existence of a root at $\sigma = \frac{1}{2}$ by looking at $||\zeta(1 + it)||$, which exhibit minima at values of t such that $\zeta(\frac{1}{2} + it) = 0$. For details, see section 1.2.2 in [15]. The computation of the roots is done in section 4 in the code below, using `fsolve` from the Scipy library.

The code below is also on GitHub, here. The `pre_compute` function reduces run time. For $s = \sigma + it$ with a very large t, you can improve accuracy by replacing the computation of the `vexp` array in the `eta` and `eta2` functions, with `vexp=np.exp(np.mod(t*alog[1:n],2*np.pi)*1j)`. Since you only need the cosine and sine of the angles in the `t*alog[1:n]` array, taking the values element-wise modulo 2π does not changes the result yet increases accuracy if t is very large.

In Figure 4.8, $\mathscr{P}$ is the set of standard primes without $3, 5, 7$. The green curve represents $||\eta_{\mathscr{P}}(s)||/||\lambda_{\mathscr{P}}||$ based on the Dirichlet series evaluated at $s = 0.95 + 2000\,i$ and truncated to n terms, with n on the X-axis. The normalization by the factor $1/||\lambda_{\mathscr{P}}||$ guarantees that both curves converge to the same value when $n \to \infty$. As per (4.4), $\lambda_{\mathscr{P}} = (1 - 3^{-s})(1 - 5^{-s})(1 - 7^{-s})$. The limit is $||\eta(s)|| = 1.0144236\ldots$ when $n = \infty$, confirmed here by Wolfram. Note that n needs to be much larger than 1000 to see both curves converge to that value.

```
from scipy.optimize import fsolve
import numpy as np

import matplotlib.pyplot as plt
import matplotlib as mpl

mpl.rcParams['axes.linewidth'] = 0.5
plt.rcParams['xtick.labelsize'] = 8
plt.rcParams['ytick.labelsize'] = 8

#--- 1. Precomputations

def pre_compute(swap_primes, del_primes, N, mode):

   p_alog = np.zeros(N)
   p_arr_omega = np.zeros(N)
   p_arr_delta = np.zeros(N)
   flag = 1

   for k in range(1, N):

      if k % 100 == 0:
         print("pre_compute:", k)
      hash_pval = {}
      new_k = k
      for p in swap_primes:
         pval = 0
         while new_k % p == 0:
            new_k = new_k // p
            pval += 1
         hash_pval[p] = pval

      for p in hash_pval:
         new_k *= (swap_primes[p])**hash_pval[p]

      p_alog[k] = np.log(new_k)
      p_arr_omega[k] = new_k
      p_arr_delta[k] = flag
      flag = -flag

      for p in del_primes:
         if k % p == 0:
            p_arr_delta[k] = 0

   if mode == 'sorted':
      ranks = np.abs(p_arr_omega).argsort().argsort()
      alog = np.zeros(N)
      arr_omega = np.zeros(N)
      arr_delta = np.zeros(N)
      for k in range(len(arr_omega)):
```

```
            rank = ranks[k]
            alog[rank] = p_alog[k]
            arr_omega[rank] = p_arr_omega[k]
            arr_delta[rank] = p_arr_delta[k]
        return(alog, arr_omega, arr_delta)
    else:
        return(p_alog, p_arr_omega, p_arr_delta)


def eta2(s, params):

    # t is imaginary part
    sigma = s[0]
    t = s[1]
    n = params[0]
    alog = params[1]
    arr_omega = params[2]
    arr_delta = params[3]
    vexp = np.exp(t*alog[1:n]*1j)
    vpow = arr_delta[1:n] * arr_omega[1:n]**(-sigma)
    sum = np.dot(vexp, vpow)
    norm = np.linalg.norm(sum)
    return(norm)

#--- 2. Removing primes in eta function

sigma = 0.5
t = 25.010858 # 0.5 + it is 3rd root of zeta
s = [sigma, t]
n0 = 4
n1 = 3500
N = 500001
swap_primes = {}
mode = 'unsorted' # options: 'sorted', 'unsorted'

del_primes1 = (3, 5, 7)
m1 = len(del_primes1) + 1
alog, arr_omega, arr_delta = pre_compute(swap_primes, del_primes1, N, mode)
params1 = [0, alog, arr_omega, arr_delta]

del_primes2 = ()
m2 = len(del_primes2) + 1
alog, arr_omega, arr_delta = pre_compute(swap_primes, del_primes2, N, mode)
params2 = [0, alog, arr_omega, arr_delta]

arr_x = []
arr_y1 = []
arr_y2 = []
for n in range(n0, n1):
    params1[0] = n
    params2[0] = n
    arr_x.append(n-1)
    arr_y1.append(eta2(s, params1))
    arr_y2.append(eta2(s, params2))

plt.scatter(arr_x, arr_y1, c='green', s = 0.1, label="m = %d" %m1)
plt.plot(arr_x, arr_y2, c='red', linewidth = 0.6, label="m = %d" %m2)
plt.axhline(y=0.0, color='black', linestyle='dotted', linewidth = 0.3)
plt.legend(loc="upper right")
plt.show()

#--- 3. Changing primes in eta function

def eta(s, params):
    # s[0] = real part, s[1] = imaginary part
    n = params[0]
    alog = params[1]
    arr_omega = params[2]
    arr_delta = params[3]
    vexp = np.exp(s[1]*alog[1:n]*1j)
    vpow = arr_delta[1:n] * arr_omega[1:n]**(-s[0])
    sum = np.dot(vexp, vpow)
    return [sum.real, sum.imag]

swap_primes = {
              3: 4.051,
```

```
               5: 7.916,
               7: 9.114
            }
del_primes = ()
mode = 'unsorted' # options: 'sorted', 'unsorted'
alog, arr_omega, arr_delta = pre_compute(swap_primes, del_primes, N, mode)
arr_x = []
arr_y = []
arr_z = []

sigma = 0.5
t1 = 14.13472514
t3 = 25.01085758
s1 = [sigma, t1]
s3 = [sigma, t3]
params = [0, alog, arr_omega, arr_delta]
for n in range(400, 300000, 1000):
   arr_x.append(n-1)
   params[0] = n
   arr_y.append(eta2(s1, params))
   arr_z.append(eta2(s3, params))

plt.plot(arr_x, arr_y, c='green', linewidth = 0.8, label="s = s1")
plt.plot(arr_x, arr_z, c='blue', linewidth = 0.8, label="s = s3")
plt.axhline(y=0.0, color='black', linestyle='dotted', linewidth = 0.3)
plt.legend(loc="upper right")
plt.show()

#--- 4. Basins of attraction

areal = []
aimag = []
acolor = []
n = 499
found1 = 0
found2 = 0
nfc = 0
eps = 0.05

params = [n, alog, arr_omega, arr_delta]
for real in np.arange(0.45, 1.05, 0.001):
   print("Real: %7.4f" %(real))
   for imag in np.arange(10.00, 40.01, 0.1):
      init = np.array([real, imag])
      root = fsolve(eta, init, params)
      areal.append(real)
      aimag.append(imag)
      sigma = min(1, root[0])
      nfc += 1
      if 0 < abs(root[0] - 0.5) < eps:
         found1 += 1
         acolor.append('red')
      elif 0 < abs(root[0] - 1.0) < eps:
         found2 += 1
         acolor.append('green')
      elif 0 < root[0] < 1:
         acolor.append('blue')
      else:
         acolor.append('gold')

plt.scatter(aimag, areal, s = 0.2, c=acolor)
plt.show()
print()
print("Found: left: %5.3f right: %5.3f" % (found1/nfc, found2/nfc))

#--- 5. Replace all primes by random numbers

from sympy import primerange, isprime

def build_swap_primes(nprimes, start, noise):

   arr_primes = list(primerange(3, nprimes))
   swap_primes = {}
   seed = 503
   np.random.seed(seed)
   N_max = len(arr_primes)
```

```
    for k in range(start, N_max):
       prime = arr_primes[k]
       u = np.random.uniform(-noise, noise)
       swap_primes[prime] = prime + u*(k**0.25)
    print("N_max: ", N_max)
    return(swap_primes, N_max)

del_primes = ()
nprimes = 300000
sigma = 0.5
t = 25.010858 # 0.5 + it is 3rd root of zeta
s = [sigma, t]

noise1 = 0.005
start = 0
swap_primes1, N_max = build_swap_primes(nprimes, start, noise1)
mode = 'sorted' # options: 'sorted', 'unsorted'
N = N_max
n0 = 4
n1 = N_max

alog, arr_omega, arr_delta = pre_compute(swap_primes1, del_primes, N, mode)
params1 = [0, alog, arr_omega, arr_delta]

noise2 = 0.000
swap_primes2, N_max = build_swap_primes(nprimes, start, noise2)
alog, arr_omega, arr_delta = pre_compute(swap_primes2, del_primes, N, mode)
params2 = [0, alog, arr_omega, arr_delta]

arr_x = []
arr_y1 = []
arr_y2 = []
for n in range(n0, n1):
   params1[0] = n
   params2[0] = n
   arr_x.append(n)
   arr_y1.append(eta2(s, params1))
   arr_y2.append(eta2(s, params2))

plt.scatter(arr_x, arr_y1, c='green', label="noise = %5.3f" %noise1, s=0.1) ### linewidth = 0.3)
plt.plot(arr_x, arr_y2, c='red', label="noise = %5.3f" %noise2, linewidth = 0.8)
plt.axhline(y=0.0, color='black', linestyle='dotted', linewidth = 0.3)
plt.legend(loc="upper right")
plt.show()
```

4.2 Quantum derivatives, GenAI, and the Riemann Hypothesis

If you are wondering how close we are to proving the Generalized Riemann Hypothesis (GRH), you should read on. The purpose of this project is to uncover intriguing patterns in prime numbers, and gain new insights on the GRH. I stripped off all the unnecessary math, focusing on the depth and implications of the material discussed here. You will also learn to use the remarkable MPmath library for scientific computing. This is a cool project for people who love math, with the opportunity to work on state-of-the-art research even if you don't have a PhD in number theory.

Many of my discoveries were made possible thanks to pattern detection algorithms (in short, AI) before leading to actual proofs, or disproofs. This data-driven, bottom-up approach is known as experimental math. It contrasts with the top-down, classic theoretical academic framework. The potential of AI and its computing power should not be underestimated to make progress on the most difficult mathematical problems. It offers a big competitive advantage over professional mathematicians focusing on theory exclusively.

My approach is unusual as it is based on the Euler product. The benefit is that you immediately know when the target function, say the Riemann zeta function $\zeta(s)$, has a root or not, wherever the product converges. Also, these products represent analytic function [Wiki] wherever they converge.

I use the standard notation in the complex plane: $s = \sigma + it$, where σ, t are respectively the real and imaginary parts. I focus on the real part only (thus $t = 0$) because of the following result: if for some $s = \sigma_0$, the product converges, then it converges for all $s = \sigma + it$ with $\sigma > \sigma_0$. Now let's define the Euler product.

Bibliography

[1] Franklin T. Adams-Watters and Frank Ruskey. Generating functions for the digital sum and other digit counting sequences. *Journal of Integer Sequences*, 12:1–9, 2009. [Link]. 41

[2] Christoph Aistleitner et al. Normal numbers: Arithmetic, computational and probabilistic aspects. 2016. Workshop [Link]. 41

[3] Adel Alamadhi, Michel Planat, and Patrick Solé. Chebyshev's bias and generalized Riemann hypothesis. *Preprint*, pages 1–9, 2011. arXiv:1112.2398 [Link]. 26

[4] David Bailey, Jonathan Borwein, and Neil Calkin. *Experimental Mathematics in Action*. A K Peters, 2007. 41

[5] Verónica Becher, A. Marchionna, and G. Tenenbaum. Simply normal numbers with digit dependencies. *Mathematika*, 69:988–991, 2023. arXiv:2304.06850 [Link]. 41

[6] Frederik Broucke. On zero-density estimates for beurling zeta functions. *Preprint*, pages 1–24, 2024. arXiv:2409:1051v1 [Link]. 19

[7] Yilan Chen et al. On the equivalence between neural network and support vector machines. *Proceedings of the 35th Conference on Neural Information Processing Systems*, pages 1–13, 2021. NeurIPS 2021 [Link]. 60, 85

[8] James Dolan. Carrying is a 2-cocycle. *Preprint*, pages 1–9, 2023. [Link]. 41

[9] Faiza Firdousi, Syeda Iram Batool, and Muhammad Amin. A novel construction scheme for nonlinear component based on quantum map. *International Journal of Theoretical Physics*, 58:3871–3898, 2019. [Link]. 41

[10] Vincent Granville. *Stochastic Processes and Simulations: A Machine Learning Perspective*. MLTechniques.com, 2022. [Link]. 19

[11] Vincent Granville. *Synthetic Data and Generative AI*. MLTechniques.com, 2022. [Link]. 19

[12] Vincent Granville. *Gentle Introduction To Chaotic Dynamical Systems*. MLTechniques.com, 2023. [Link]. 19, 20, 25, 41, 57, 85

[13] Vincent Granville. *Building Disruptive AI & LLM Technology from Scratch*. MLTechniques.com, 2024. [Link]. 41, 53, 54, 60, 76, 79, 85

[14] Vincent Granville. *State of the Art in GenAI & LLMs – Creative Projects, with Solutions*. MLTechniques.com, 2024. [Link]. 26, 55, 60, 85

[15] Vincent Granville. *Statistical Optimization for AI and Machine Learning*. MLTechniques.com, 2024. [Link]. 20, 55, 58, 59, 75, 76, 85

[16] Vincent Granville. *Synthetic Data and Generative AI*. Elsevier, 2024. [Link]. 24, 55, 58, 73, 85

[17] Vincent Granville. *0 and 1 – From Elemental Math to Quantum AI*. MLTechniques.com, 2025. [Link]. 19, 36, 37, 38, 40, 41, 47, 57, 85

[18] Vincent Granville. *Blueprint: Next-Gen Enterprise RAG & LLM 2.0 – Nvidia PDFs Use Case*. 2025. MLTechniques.com [Link]. 79

[19] Emil Grosswald. Oscillation theorems of arithmetical functions. *Transactions of the American Mathematical Society*, 126:1–28, 1967. [Link]. 26

[20] Adam J. Harper. Moments of random multiplicative functions, II: High moments. *Algebra and Number Theory*, 13(10):2277–2321, 2019. [Link]. 26

[21] Adam J. Harper. Moments of random multiplicative functions, I: Low moments, better than squareroot cancellation, and critical multiplicative chaos. *Forum of Mathematics, Pi*, 8:1–95, 2020. [Link]. 26

[22] Adam J. Harper. Almost sure large fluctuations of random multiplicative functions. *Preprint*, pages 1–38, 2021. arXiv [Link]. 26

[23] Yingcong Li et al. Mechanics of next token prediction with self-attention. *Proceedings of the 27th International Conference on Artificial Intelligence and Statistics*, pages 1–43, 2024. arXiv:2403:08081 [Link]. 60, 85

[24] Yue Li, Hongxia Wang, and Mauro Barni. A survey of deep neural network watermarking techniques. *Preprint*, pages 1–22, 2021. arXiv:2103:09274 [Link]. 85

[25] M. Madritsch and J. Thuswaldner. The level of distribution of the sum-of-digits function of linear recurrence number systems. *Journal de Théorie des Nombres de Bordeaux*, 34:449–482, 2022. MLTechniques.com [Link]. 41

[26] Vaibhav Mohanty et al. Maximum mutational robustness in genotype–phenotype maps follows a self-similar blancmange-like curve. *The Royal Society Publishing*, pages 1–16, 2023. [Link]. 41

[27] Mohammadamin Moradi et al. Data-driven model discovery with Kolmogorov-Arnold networks. *Preprint*, pages 1–6, 2024. arXiv:2409.15167 [Link]. 41

[28] K.S. Narendra and K. Parthasarathy. Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, 1:4–27, 1990. [Link]. 41

[29] Michel Planat and Patrick Solé. Efficient prime counting and the Chebyshev primes. *Preprint*, pages 1–15, 2011. arXiv:1109.6489 [Link]. 26

[30] Jan-Christoph Schlage-Putcha and Jasson Vindas. The prime number theorem for Beurlings generalized numbers – new cases. pages 1–26, 2011. [Link]. 19

[31] Terence Tao. Biases between consecutive primes. *Tao's blog*, 2016. [Link]. 26

[32] Yury V. Tiumentsev and Mikhail V. Egorchev. *Neural Network Modeling and Identification of Dynamical Systems*. Elsevier, 2019. 41

[33] Chukwudubem Umeano and Oleksandr Kyriienko. Ground state-based quantum feature maps. *Preprint*, pages 1–8, 2024. arXiv:2024.07174 [Link]. 41

[34] Joseph Vandehey. On the binary digits of $\sqrt{2}$. *Preprint*, pages 1–6, 2017. arXiv:1711.01722 [Link]. 41

[35] Troy Vasiga and Jeffrey Shallit. On the iteration of certain quadratic maps over GF(p). *Discrete Mathematics*, 277:219–240, 2004. [Link]. 41

[36] Henry S. Warren. *Hacker's Delight*. Addison-Wesley Professional, second edition, 2012. 41

[37] Rose Yu and Rui Wang. Learning dynamical systems from data: An introduction to physics-guided deep learning. *Proceedings of the National Academy of Sciences of the United States of America*, 121, 2024. [Link]. 41