Appendix B

Quantum, chaotic and fractal types of algorithmic convergence

The problems in this appendix complement the material discussed in the book. Section B.1 deals with the digit sum, in particular with its generating and cumulative functions. It leads to an interesting fractal convergence behavior featured in Figure B.1. Section B.2 discusses linear recurrences with non-fractal, chaotic convergence, illustrated in Figures B.2 and B.3.

B.1 Digit count generating function and fractal convergence

Let H_k be the number of 1 in the binary expansion of the positive integer k, also called the hamming weight of k. These weights are listed as the sequence A000120 in the online encyclopedia of integer sequences (OEIS), see here. The Hamming weight normalized cumulative function is pictured in Figure B.1 and defined as

$$S_H(n) = \frac{1}{n \log_2 n} \sum_{k=0}^{n} H_k, \quad n = 2, 3, 4, \ldots \tag{B.1}$$

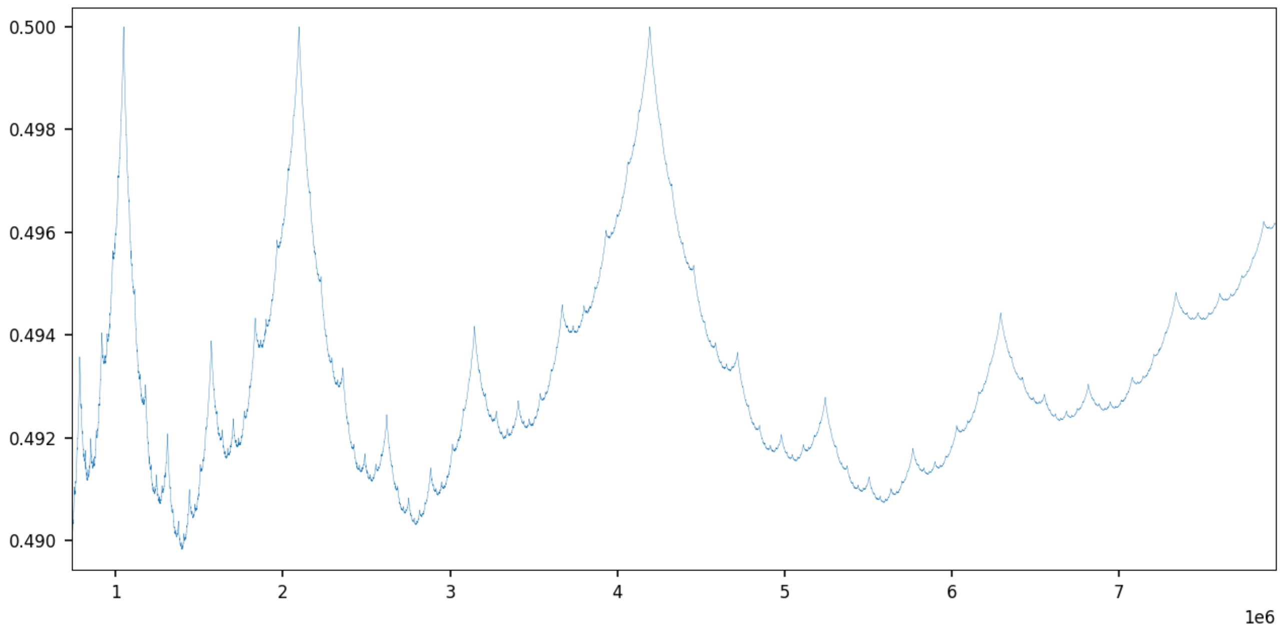

Figure B.1: Hamming weights normalized cumulative function $S_H(n)$ with $n \leq 8 \times 10^6$ on the X-axis

Clearly, $S_H(n) \to \frac{1}{2}$ as $n \to \infty$ with the peaks occurring when n is a power of 2. The curve is fractal-like: the graph between two successive peaks is identical to that between the two previous peaks, but stretched horizontally be a factor 2, and with minima increasing over time, thus causing some vertical compression as n increases. See below the code for the computations and plot generation. The code is also on GitHub, here.

```python
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
plt.rcParams['legend.fontsize'] = 'x-small'

def countSetBits(n):
   # compute H(n)
   count = 0
   while(n):
      count += n & 1
      n >>= 1
   return(count)

sum = 0
arr_k = []
arr_sum = []

for k in range(0,8000000):
   d = countSetBits(k)
   sum += d
   arr_k.append(k)
   if k < 2:
      arr_sum.append(0)
   else:
      arr_sum.append(sum/(k*np.log2(k)))
   if k % 10 == 0:
      print("sss", k, d, sum)

plt.plot(arr_k, arr_sum, linewidth = 0.2)
plt.show()
```

Now I discuss some interesting properties related to the generating functions attached to H_n. It is easy to prove that

$$\Lambda_n(\lambda, x) := \prod_{k=0}^{n-1}\Big(1 + e^{\lambda}x^{2^k}\Big) = \sum_{k=0}^{2^n-1} e^{\lambda H_k}x^k \tag{B.2}$$

Thus $\Lambda_n(0, x) = (1 - x^{2^n})/(1 - x)$. By virtue of (B.2), we also have

$$\Lambda_n(\lambda, 1) = \big(1 + e^{\lambda}\big)^n = \sum_{k=0}^{n}\binom{n}{k}e^{k\lambda} = \sum_{k=0}^{2^n-1} e^{\lambda H_k} \tag{B.3}$$

Now, taking the m-th derivative with respect to λ and then setting $\lambda = 0$, we obtain

$$\sum_{k=0}^{2^n-1}(H_k)^m = \sum_{k=0}^{n}\binom{n}{k}k^m = 2^n n!\sum_{k=0}^{m}\frac{S(m,k)}{2^k(n-k)!}, \quad m = 0, 1, 2\dots \tag{B.4}$$

where $S(\cdot,\cdot)$ denotes the Stirling numbers of the second kind. The left identity in (B.4) follows from (B.3) and is true even if m is not an integer. The expression on the right in (B.4) corresponds to the m-th moment of a Bernoulli distribution of parameters $(n, \frac{1}{2})$, multiplied by 2^n.

The standard form of the Hamming weights generating function, identical to that of binary digit sums, is studied in [34]. See also [37]. The main result is Theorem 1 in [34], stating that

$$\sum_{k=0}^{\infty} H_k x^k = \frac{1}{1-x}\sum_{m=0}^{\infty}\frac{x^{2^m} - 2x^{2^{m+1}} + x^{3\cdot 2^m}}{(1 - x^{2^m})(1 - x^{2^{m+1}})}. \tag{B.5}$$

The "digit sum" entry in Wolfram also features beautiful results, see here. Among others:

$$\sum_{k=1}^{\infty}\frac{H_k}{k(k+1)} = \log 2, \quad \sum_{k=1}^{\infty}\frac{(2k+1)H_k}{k^2(k+1)^2} = \frac{\pi^2}{9}$$

$$\sum_{k=1}^{\infty}\frac{H_k + H'_k}{2k(2k+1)} = \gamma, \quad \sum_{k=1}^{\infty}\frac{H_k - H'_k}{2k(2k+1)} = \log\frac{4}{\pi}$$

where γ is the Euler-Mascheroni constant and H'_k is the number of 0 in the binary expansion of k.

The Hamming weights H_n can be computed with the recursion $H_{2n} = H_n$, $H_{2n+1} = H_n + 1$ with $H_0 = 0$. A related sequence with many interesting properties is Stern's diatomic series (A_n) with $A_0 = 0$ and $A_1 = 1$, defined by the recursion $A_{2n} = A_n, A_{2n+1} = A_n + A_{n+1}$. Finally, the integer sequence defined by $C_n = \frac{1}{2}C_{n-1}$ if C_{n-1} even, $C_n = 3C_{n-1} + 1$ otherwise, is presumed to converge to 1 regardless of the initial condition $C_0 > 0$. This is known as the Collatz conjecture.

B.2 Other examples of chaotic convergence

For centuries, mathematicians worked on problems where the convergence is either smooth or does not happen. Now the concept of chaotic convergence is mainstream, popularized by the stochastic gradient descent in deep neural networks, central to LLMs. In this book, most cases also involve various types of chaotic convergence, for instance in Figures 4.16, 5.1, 6.7, 6.8, and 6.11. In some examples such as Figure B.1, the chaos exhibits a fractal structure. In other examples such as Figures 2.3, 2.7, 6.4, and 6.9, it looks like we have multiple curves converging to a same limit, when in reality there is only one, with jumps from one level to another every millisecond: this pattern is strikingly similar to quantum states. This section features more illustrations falling in these various categories.

For examples of chaotic convergence in the context of explainable deep neural networks, see Figure 4.6 featuring stochastic gradient descent, Figure 4.14 featuring swarm optimization, and Figure 4.2 featuring adaptive loss function all in my book on no-Blackbox LLM architectures [21]. Yet another type is asymptotic periodicity where x_n converges to multiple values depending on the residue class of n modulo some integer. See section 5.2.1.

B.2.1 Smooth convergence but with multiple branches

The first example deals with the recursion $x_{n+3} = 2x_{n+2} - 16x_{n+1} + 4x_n$ with initial conditions $x_0 = 0$, $x_1 = 1$, and $x_2 = 1$. We are interested in the limit

$$\rho := \lim_{n\to\infty} |x_n|^{1/n} \tag{B.6}$$

if it exists. It does in this example. To compute it, proceed as follows:

- Find the roots of the characteristic polynomial, in this case the cubic equation $x^3 = 2x^2 - 16x + 4$.
- Identify the root with the largest norm. Here we have two complex conjugate roots with the same norm, and one real root. The complex roots have the largest norm.
- Then ρ is the square root of the norm in question. More specifically,

$$\rho = \sqrt{\frac{12\omega}{2\omega - \omega^2 + 44}}, \quad \text{with } \omega = \sqrt[3]{82 + 6\sqrt{2553}} \tag{B.7}$$

First, I used AI to find ρ. The LLMs that I tested correctly identify $r \approx 0.2572$ as the real root of the cubic equation, and $\rho \approx 3.9436$ as the solution. Perplexity even mentioned that $\rho = 2\,||r||^{-1/2}$, which is correct. But they all failed when providing the exact values for the roots, thus the exact value for ρ was also incorrect. To be fair, I used the free version of these tools. Then, I used the `solve` function from the SymPy Python library. It is based on symbolic mathematics to find exact solutions. That's how I obtained (B.7).

The most interesting part is how $\rho_n = |x_n|^{1/n}$ converges to ρ as $n \to \infty$, see Figure B.2 where ρ_n on the Y-axis has a different color depending on $n \bmod 7$, with n on the X-axis. We jump from one branch to another each time n is incremented, yet eventually ρ_n converges to n. The associated Python code is listed in section B.2.2.

B.2.2 Chaotic convergence with multiple branches

I now discuss the case $x_{n+2} = |x_{n+1} - 3x_n|$ with initial conditions $x_0 = 1$ and $x_1 = 1$. Again, we are interested in the same limit ρ defined by (B.6). At first glance, this case seems easier as the characteristic polynomial is now of degree 2 instead of 3. However, the absolute value in the recursion makes it different with a dramatic shift in behavior as seen in Figure B.3, by contrast to Figure B.2. Both plots show $|x_n|^{1/n}$ on the Y-axis with n on the X-axis.

It is reminiscent of the problem about random Fibonacci sequences discussed in section 7.4 in [21]. In that example, the equivalent of the ratio $r_n = x_{n+1}/x_n$ asymptotically oscillates between 5 different values. Taking the geometric mean of these values yields the correct, exact value for ρ. However the situation is a lot more complicated here. Even though empirical evidence points to the limit $\rho \approx 1.58010$, it is not obvious to prove convergence, let alone putting an exact value on ρ. Interestingly, some LLMs erroneously believe that the answer is $\rho = \sqrt{3}$.

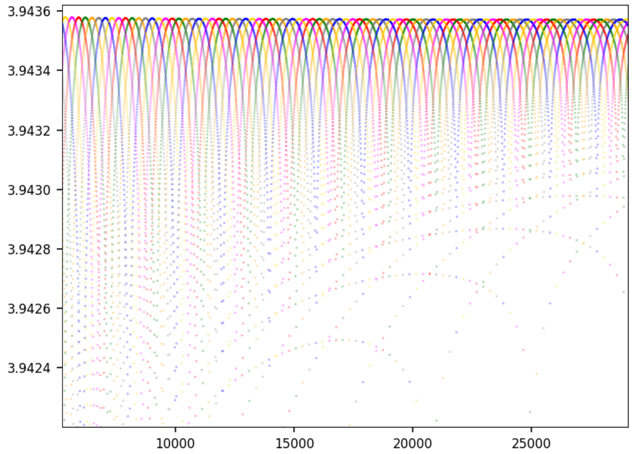

Figure B.2: Constant jumps in $|x_n|^{1/n}$ gives the illusion of 7 curves but there is only one (section B.2.1).

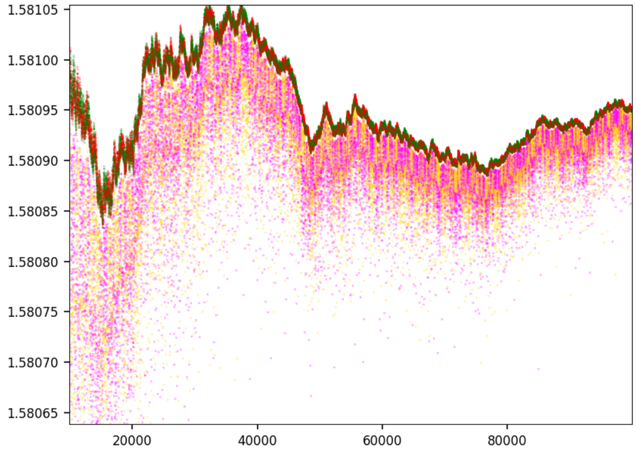

Figure B.3: 4-colors quantum regime similar to that of Figure B.2 but now with chaos (section B.2.2).

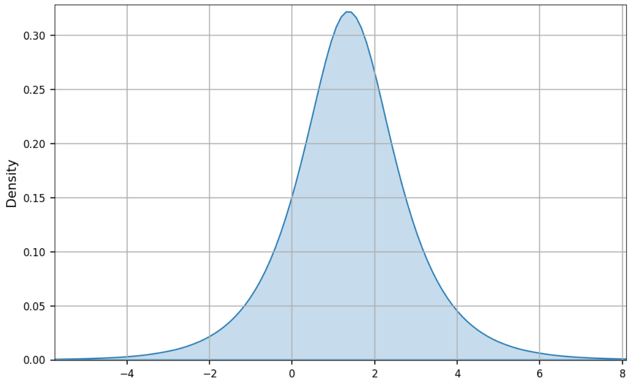

Figure B.4: Empirical PDF for $\log |x_{n+1}/x_n|$, case featured in Figure B.2

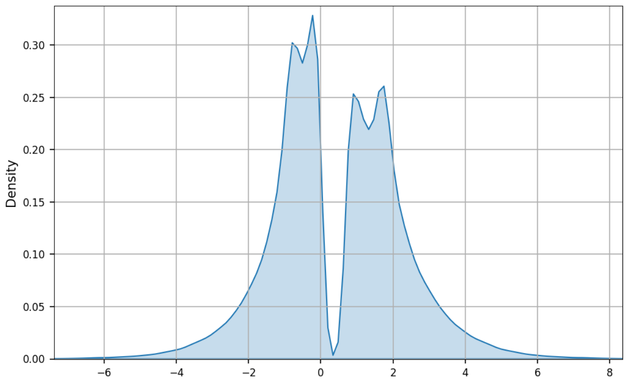

Figure B.5: Empirical PDF for $\log |x_{n+1}/x_n|$, case featured in Figure B.3

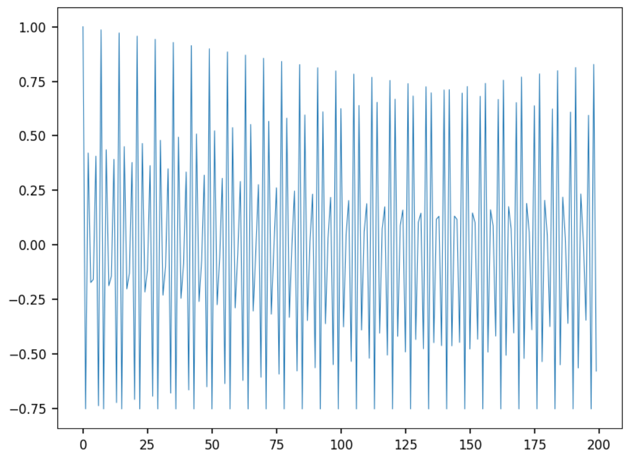

Figure B.6: Autocorrel function for $\log |x_{n+1}/x_n|$, case featured in Figure B.2

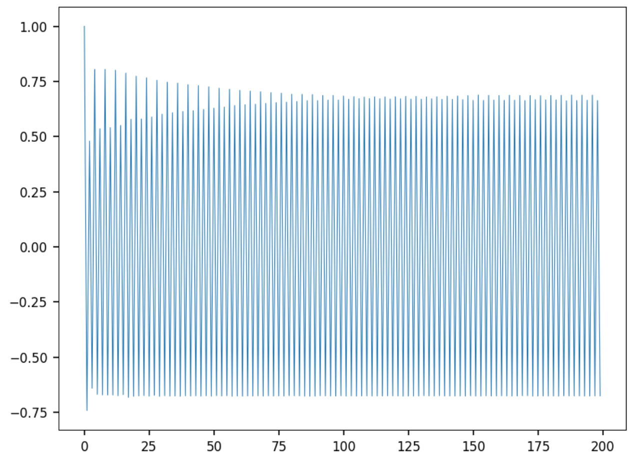

Figure B.7: Autocorrel function for $\log |x_{n+1}/x_n|$, case featured in Figure B.3

The curve $\rho_n = |x_n|^{1/n}$ has 4 chaotic bands. For a specific n, the band that ρ_n belongs to depends on n mod 4. The green and red bands are interlaced but distinct. The yellow and pink bands are somewhat more separated. This is further confirmed when looking at the empirical autocorrelation function pictured in Figure B.7, showing the correlation between the sequences $\log |x_{n+1}/x_n|$ and $\log |x_{n+k+1}/x_{n+k}|$ for $0 \leq k < 200$. A cycle of length 4 is clearly visible. By contrast, the case studied in section B.2.1 has a cycle of length 7, as pictured in Figure B.6, which also explains the 7 branches in Figure B.2.

Finally, I looked at the empirical probability density function (EPDF in Python), computed on the first 10^5 values of $\log |x_{n+1}/x_n|$, skipping the first 10,000 ones. Not surprisingly, the EPDF is smooth and classic for the case featured in Figure B.2, but bumpy with 4 main peaks and a deep valley for the example discussed here. You can see the contrast by comparing Figures B.4 and B.5 respectively. To conclude, the dark upper boundary in Figure B.3 looks like a Brownian motion to the layman, but it is in fact very different. Ours converges (the variance tends to zero as n increases) while the variance of a Brownian motion keeps growing. However, this is a small issue: rescaling would easily fix it. The main difference is that our curve is significantly more chaotic, and thus suitable as a model when increased chaos is needed. Metrics measuring the amount of chaos are discussed in section 2.3 in my book on chaotic dynamical systems [14].

Below is the Python code used to do the analyses and producing the plots in sections B.2.1 and B.2.2. The code is also on GitHub, here. The `mode` parameter allows you to choose between the recursion in section B.2.1 and that in section B.2.2.

```
import numpy as np
import gmpy2
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
plt.rcParams['legend.fontsize'] = 'x-small'

kmax = 3000
ctx = gmpy2.get_context()
ctx.precision = kmax
ndigits = ctx.precision
z = gmpy2.log(2)

N = 50000
mode = 'quantum' # options: 'quantum' or 'chaotic'

x0 = gmpy2.mpfr(0)
x1 = gmpy2.mpfr(1)
x2 = gmpy2.mpfr(1)
arr_k = []
arr_x = []
arr_col = []
arr_log_rho = []

def set_color(k, mode):

   if mode == 'quantum':
      if k % 7 == 0:
         color='red'
      elif k % 7 == 1:
         color='blue'
      elif k % 7 == 2:
         color='green'
      elif k % 7 == 3:
         color='gold'
      elif k % 7 == 4:
         color='orange'
      elif k % 7 == 5:
         color='magenta'
      elif k % 7 == 6:
         color='gray'

   elif mode == 'chaotic':
      if k % 4 == 0:
         color = 'red'
      elif k % 4 == 1:
         color = 'gold'
      elif k % 4 == 2:
         color = 'green'
      elif k % 4 == 3:
         color = 'magenta'
      else:
         color = 'gray'
   else:
```

```
        print("Unsuported mode:", mode)
        exit()
    return(color)


#--- Main

for k in range(2,N):
    color = set_color(k, mode)
    if mode == 'quantum':
        x = 2*x2 - 16*x1 + 4*x0
    elif mode == 'chaotic':
        x = abs(x2 - 3*x1)
    v = gmpy2.log(abs(x))/k
    w = gmpy2.exp(v)
    log_rho = np.log(abs(float(x/x2)))
    if k % 1 == 0 and k > N/10:
        print("%6d log_rho: %8.5f" % (k, log_rho))
        arr_k.append(k)
        arr_x.append(w)
        arr_col.append(color)
        arr_log_rho.append(log_rho)
    x0 = x1
    x1 = x2
    x2 = x

plt.scatter(arr_k, arr_x, c=arr_col, s=0.02)
plt.show()


#--- Plot EPDFs

np_log_rho = np.array(arr_log_rho)
import seaborn as sns
plt.figure(figsize=(8, 5))
##sns.ecdfplot(np_rho)
sns.kdeplot(data=np_log_rho, fill=True)
plt.grid(True)
plt.show()

meanlog = np.mean(np_log_rho)
mean = np.exp(meanlog)
print("\nRho:", mean)
print()


#--- Compute autocorrels for log(x/x2)

nobs = len(np_log_rho)
arr_lag = []
arr_autocorrel = []

for lag in range(200):
    mean1 = np.mean(np_log_rho[0: nobs-lag])
    mean2 = np.mean(np_log_rho[lag:nobs])
    std1 = np.std(np_log_rho[0: nobs-lag])
    std2 = np.std(np_log_rho[lag:nobs])
    dotprod = np.dot(np_log_rho[0: nobs-lag], np_log_rho[lag:nobs])
    dotprod /= (nobs-lag)
    autocorrel = (dotprod - mean1*mean2)/(std1*std2)
    arr_lag.append(lag)
    arr_autocorrel.append(autocorrel)
    print("Autcorrel lag %3d: %8.5f" %(lag, autocorrel))

plt.plot(arr_lag, arr_autocorrel, linewidth = 0.5)
plt.show()
```

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]. 12, 14, 23, 32, 45

[2] Christoph Aistleitner et al. Normal numbers: Arithmetic, computational and probabilistic aspects. 2016. Workshop [Link]. 12, 23, 32, 45

[3] Adel Alamadhi, Michel Planat, and Patrick Solé. Chebyshev's bias and generalized Riemann hypothesis. *Preprint*, pages 1–9, 2011. arXiv:1112.2398 [Link]. 84

[4] Gökalp Alpan and Maxim Zinchenko. Lower bounds for weighted Chebyshev and orthogonal polynomials. *Preprint*, 2024. arXiv:2408.11496v [Link]. 56

[5] David Bailey, Jonathan Borwein, and Neil Calkin. *Experimental Mathematics in Action*. A K Peters, 2007. 11, 23, 32, 45, 61

[6] Verónica Becher, A. Marchionna, and G. Tenenbaum. Simply normal numbers with digit dependencies. *Mathematika*, 69:988–991, 2023. arXiv:2304.06850 [Link]. 12, 23, 32, 45

[7] Frederik Broucke. On zero-density estimates for Beurling zeta functions. *Preprint*, pages 1–24, 2024. arXiv:2409:1051v1 [Link]. 77

[8] James Dolan. Carrying is a 2-cocycle. *Preprint*, pages 1–9, 2023. [Link]. 12, 23, 32, 45

[9] David Doty, Jack H. Lutz, and Satyadev Nandakumar. Finite-state dimension and real arithmetic. *Information and Computation*, 205:1640–1651, 2007. arXiv:cs/0602032 [Link]. 70

[10] 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]. 12, 23, 32, 45

[11] P. M. Gauthier. Approximating the Riemann zeta-function by polynomials with restricted zeros. *Canadian Mathematical Bulletin*, 62(3):475–478, 2018. [Link]. 95

[12] Vincent Granville. *Stochastic Processes and Simulations: A Machine Learning Perspective*. MLT, 2022. [Link]. 77, 94

[13] Vincent Granville. *Synthetic Data and Generative AI*. MLT, 2022. [Link]. 77

[14] Vincent Granville. *Gentle Introduction To Chaotic Dynamical Systems*. MLT, 2023. [Link]. 7, 10, 11, 12, 14, 15, 18, 23, 32, 44, 45, 57, 61, 69, 70, 77, 78, 83, 84, 107

[15] Vincent Granville. *Building Disruptive AI & LLM Technology from Scratch*. MLT, 2024. [Link]. 15, 23, 32, 45, 100

[16] Vincent Granville. *State of the Art GenAI & LLMs, Creative Projects & Solutions*. MLT, 2024. [Link]. 20, 84

[17] Vincent Granville. *Statistical Optimization for AI and Machine Learning*. MLT, 2024. [Link]. 78

[18] Vincent Granville. *Synthetic Data and Generative AI*. Elsevier, 2024. [Link]. 82

[19] Vincent Granville. *Blueprint: Next-Gen Enterprise RAG & LLM 2.0 – Nvidia PDFs Use Case*. 2025. MLT [Link]. 100

[20] Vincent Granville. Simple, efficient, secure, accurate enterprise AI xLLM 2.0 architecture & operating system. 2025. BondingAI internal report `bdai-scores.pdf`, July 2025. 100

[21] Vincent Granville. *No-Blackbox, Secure, Efficient AI and xLLM Solutions*. MLT, 2026. [Link]. 95, 100, 105

[22] Vincent Granville and Richard L Smith. Disaggregation of rainfall time series via Gibbs sampling. *NISS Technical Report*, pages 1–21, 1996. [Link]. 94

[23] Emil Grosswald. Oscillation theorems of arithmetical functions. *Transactions of the American Mathematical Society*, 126:1–28, 1967. [Link]. 84

[24] Adam J. Harper. Moments of random multiplicative functions, II: High moments. *Algebra and Number Theory*, 13(10):2277–2321, 2019. [Link]. 85

[25] 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]. 85

[26] Adam J. Harper. Almost sure large fluctuations of random multiplicative functions. *Preprint*, pages 1–38, 2021. arXiv [Link]. 85

[27] Christopher Lutsko, Athanasios Sourmelidis, and Niclas Technau. Pair correlation of the fractional parts of αn^θ. *Journal of the European Mathematical Society*, 27:4069–4082, 2025. arXiv:2106.09800 [Link]. 71

[28] 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. MLT [Link]. 32, 45

[29] 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]. 12, 23, 32, 45

[30] Mohammadamin Moradi et al. Data-driven model discovery with Kolmogorov-Arnold networks. *Preprint*, pages 1–6, 2024. arXiv:2409.15167 [Link]. 24, 32, 45

[31] 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]. 23, 32, 45

[32] Alan Oppenheim and Ronald W. Schafer. *Discrete–Time Signal Processing*. Prentice Hall, 1999. 97

[33] Michel Planat and Patrick Solé. Efficient prime counting and the Chebyshev primes. *Preprint*, pages 1–15, 2011. arXiv:1109.6489 [Link]. 84

[34] Frank Ruskey. Generating functions for the digital sum and other digit counting sequences. *Journal of Integer Sequences*, 12:1–9, 2009. [Link]. 104

[35] Klaus Schiefermayr and Maxim Zinchenko. Norm estimates for Chebyshev polynomials, i. *Journal of Approximation Theory*, 265, 2021. [Link]. 56

[36] Jan-Christoph Schlage-Putcha and Jasson Vindas. The prime number theorem for Beurlings generalized numbers – new cases. pages 1–26, 2011. [Link]. 77

[37] Maxwell Schneider and Robert Schneider. Digit sums and generating functions. *Preprint*, pages 1–10, 2018. arXiv:1807.06710 [Link]. 104

[38] Terence Tao. Biases between consecutive primes. *Tao's blog*, 2016. [Link]. 84

[39] Yury V. Tiumentsev and Mikhail V. Egorchev. *Neural Network Modeling and Identification of Dynamical Systems*. Elsevier, 2019. 23, 32, 45

[40] Chukwudubem Umeano and Oleksandr Kyriienko. Ground state-based quantum feature maps. *Preprint*, pages 1–8, 2024. arXiv:2024.07174 [Link]. 12, 23, 32, 45

[41] Joseph Vandehey. On the binary digits of $\sqrt{2}$. *Preprint*, pages 1–6, 2017. arXiv:1711.01722 [Link]. 11, 23, 32, 45, 61

[42] Troy Vasiga and Jeffrey Shallit. On the iteration of certain quadratic maps over GF(p). *Discrete Mathematics*, 277:219–240, 2004. [Link]. 23, 32, 44

[43] Henry S. Warren. *Hacker's Delight*. Addison-Wesley Professional, second edition, 2012. 12, 23, 32, 45

[44] 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]. 23, 32, 45