Statistical Inference for Elementary Oscillator Based TRNGs

Maciej Skorski

University of Warsaw

CryptArchi 2023

Outline

Outline

Credits 🙏

- Viktor Fisher, for inspiration,
- Nathalie Bochard, for insights about empirical data,
- Laboratoire Hubert Curien, for research visits opportunities.

Background

- Jitter means stochastic perturbations to a signal phase
- Jitter is a hardware artefact exploited to generate randomness.
- In general, the jittered clock signal is [Baudet et al., 2011]

$$s(t) = f(\omega(t + \xi(t))), \tag{1}$$

  where t is the time argument, f is a pulse wave, ξ is a *jitter stochastic process* and t is time.
- In the elementary ring-oscillator, we will assume that $\xi(t)$ follows a Brownian motion (variance accumulates over time!)

Related Work

- [Baudet et al., 2011] model $\xi(t)$ as a random walk
- [Baudet et al., 2011] obtain approximations to finite-dimensional probabilities and the entropy rate
- [Fischer and Lubicz, 2014] proposed a method to estimate the random walk parameters in hardware
- ...

Approach

💡 Differently to [Baudet et al., 2011], let's work in the time domain!

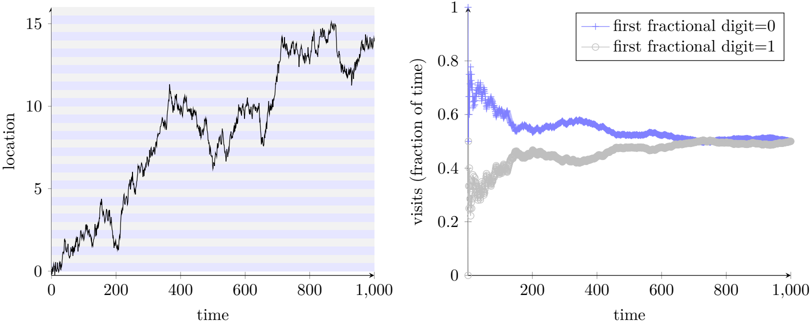


Figure 1: Blue or gray? Stochastic behaviour of the oscilator-based TRNG.

🧩 How to extract (nearly) random bits?

Outline

Problem Statement

For $t \geqslant 0$ we study the process

$$X_t \sim \mathsf{Brownian}(t|\mu, \sigma, x_0)$$
$$b_t = \begin{cases} 1 & X_t \bmod 1 < \frac{1}{2} \\ 0 & X_t \bmod 1 \geqslant \frac{1}{2} \end{cases} \tag{2}$$

and want to solve

Problem

What are the ***distribution and entropy*** *of bits generated by* (2)*?*

Objectives

- ✔️ Find finite-dimensional probabilities
- ✔️ Propose an efficient implementation
- 🚧 Find exact formulas for the entropy rate

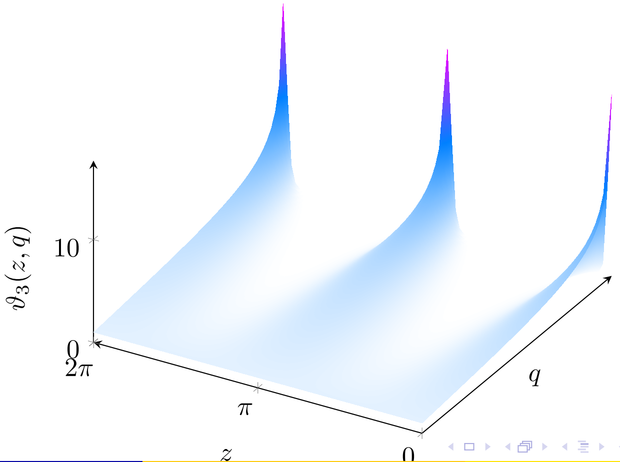
Outline

1. Introduction
2. Problem
3. Results
   - Finite-dimensional Inference
   - Implementation
4. Summary

Preliminaries

In our calculations we use the *third elliptic theta function*, defined as

$$\vartheta_3(z, q) \triangleq \sum_{k\in\mathbb{Z}} q^{k^2} \mathrm{e}^{2kiz}. \tag{3}$$



Outline

Binary Digits of Gaussian Distributions

Theorem (Gaussian Distribution Modulo 1)

Suppose that X follows the gaussian distribution with mean μ and variance σ^2. Let $Y = X \bmod 1$, then

$$\mathbf{P}\{Y = y\} = \vartheta_3(\pi(y-\mu), \mathrm{e}^{-2\pi^2\sigma^2}). \tag{4}$$

Note: techniques come from [Wilms, 1994].

Phase Finite-dimensional Distribution

The phase is a Markov chain distributed as follows:

Theorem (Distribution of Brownian Motion Modulo 1)

Let X_t be Brownian motion with drift μ and volatility σ, and $Y_t = X_t \mod 1$. Then for any $0 < t_0 < t_1 < \ldots < t_n$, any $x_0 \in \mathbb{R}$ and any $y_1, \ldots, y_n \in [0, 1)$ we have

$$\mathbf{P}\{y_{t_1}, \ldots, y_{t_n} | x_{t_0}\} = \prod_{i=1}^{n} \vartheta_3 \left(\pi(\Delta y_i - \mu \Delta t_i), \mathrm{e}^{-2\pi^2\sigma^2\Delta t_i} \right), \tag{5}$$

where we define $\Delta y_i = y_{t_i} - y_{t_{i-1}}$, $\Delta t_i = t_i - t_{i-1}$ and $y_0 = x_0$.

Theorem (Output Bits Distribution)

For any $0 < t_1 < \ldots < t_n$ *and any* $y_0 \in [0,1)$

$$\mathbf{P}\{b_{t_n},\ldots,b_{t_1}|y_0\} = 2^{-n}\mathbf{E}\left[\prod_{i=1}^{n}\vartheta_3(\pi(\epsilon_i U_i - \epsilon_{i-1}U_{i-1} - \mu\Delta t_i), \mathrm{e}^{-2\pi^2\sigma^2\Delta t_i})|U_0 = y_0\right], \quad (6)$$

where we denote $\epsilon_i = (-1)^{1-b_{t_i}}$ *for* $i = 1,\ldots,n$, $\epsilon_0 = 1$ *and* U_i *for* $i = 0,\ldots,n$ *are independent random variables uniformly distributed on* $[0, 1/2)$.

Example: Distributions of Bit Patterns

init. distribution bits pattern	cold start $x_0 = (0.0)$	cold start $x_0 = (0.5)$	stationary
000	0.083597	0.189357	0.136284
001	0.081000	0.183290	0.131952
010	0.068965	0.130280	0.099813
011	0.090121	0.173388	0.131952
100	0.173388	0.090121	0.131952
101	0.130280	0.068965	0.099813
110	0.183290	0.081000	0.131952
111	0.189357	0.083597	0.136284

Table 1: Probability of bit blocks in a toy oscillator-based TRNG with $\mu = 0.2$, $\sigma = 0.25$, $t_0 = 0$, $\Delta t_n = 1$ evaluated under the cold start ($x_0 = 0$, or $x_0 = 0.5$) and the stationary distributions. The stationary case reveals bias in frequencies of blocks, even though a single bit is unbiased.

Outline

Dependencies

```
import numpy as np
import mpmath as mpm
from scipy.integrate import nquad
```

Listing 1: Dependencies

Jacobi's Theta

```
def jtheta3(z,q):
    """
        Jacobi Theta-3 function.
        Note: The scaling convention follows NIST (dlmf.nist.gov)
    """
    def fn(z,q):
        jtheta3_fn = lambda z,q: mpm.jtheta(n=3,z=z,q=q)
        jtheta3_fn = np.vectorize(jtheta3_fn,otypes=(np.cfloat,))
        return jtheta3_fn(z,q)

    return fn(z,q)
```

Listing 2: Jacobi theta-function ϑ_3.

Phase Log-Probability

```python
def brownian_mod1_logp(mu,sigma,t,y):
    """Evaluate the finite-dimensional log-probabilities of Brownian
    Args:
        mu: drift of Brownian motion
        sigma: velocity of Brownian motion
        t: sampling times: start time at index 0
        y: positions to evaluate density: start location at index 0
    """
    y = np.array(y)
    t_diff = np.diff(t,axis=0)
    y_diff = np.diff(y,axis=0)
    # evaluate probability with theta function
    z = np.pi*(y_diff-mu*t_diff)
    q = np.exp(-2*np.pi**2*t_diff*sigma**2)
    return np.log(jtheta3(z,q)).sum(0)
```

Listing 3: Density of Brownian Motion modulo 1

```python
def bits_prob(bits,t,mu,sigma,x0=0):
    """Evaluate probability of a bit sequence.

    Args:
        bits (array of booleans): Sequence of bits
        t (array of reals): Sampling times
        mu (real): Brownian motion drift
        sigma (real): Brownian motion velocity
        x0 (real, optional): Initial position. Defaults to 0.
    """

    def fn(*y):
        return np.exp(brownian_mod1_logp(mu,sigma,t,y=((x0,)+y)))
    ranges = [(0,1/2) if b==1 else (1/2,1) for b in bits]
    return nquad(fn,ranges)
```

Listing 4: Probability of output bits, starting from the chosen location.

Outline

Wrap-Up & Future Work

Achieved:

- elegant analytical formulas for finite-diensional probabilities of phase and raw bits.
- numerical implementation

Pending:

- 🤔 formulas for limiting entropies (Shannon Entropy, Min-Entropy, Smooth Min-Entropy).

Thank you!

References I

Abramowitz, M. and Stegun, I. A. (1964).
Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables.
Dover, New York, ninth dover printing, tenth gpo printing edition.

Baudet, M., Lubicz, D., Micolod, J., and Tassiaux, A. (2011).
On the security of oscillator-based random number generators.
Journal of cryptology, 24(2):398–425.

Fischer, V. and Lubicz, D. (2014).
Embedded evaluation of randomness in oscillator based elementary trng.
In *Cryptographic Hardware and Embedded Systems–CHES 2014: 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings 16*, pages 527–543. Springer.

Weisstein, E. W. (2000).
Jacobi theta functions.
https://mathworld. wolfram. com/.

References II

Wilms, R. J. G. (1994).
Fractional parts of random variables: limit theorems and infinite divisibility.