*Publications of the Astronomical Society of Australia* (2025), 1–6

ARTICLE TYPE

# Random Number Generation from Pulsars

Hayder Tirmazi

Grove School of Engineering, City College of New York, New York, 10031, New York, United States of America
**Author for correspondence:** Hayder Tirmazi, Email: hayder.research@gmail.com.

**Abstract**

Pulsars exhibit signals with precise inter-arrival times that are on the order of milliseconds to seconds depending on the individual pulsar. There is subtle variation in the timing of pulsar signals, primarily due to the presence of gravitational waves, intrinsic variance in the period of the pulsar, and errors in the realization of Terrestrial Time (TT). Traditionally, these variations are dismissed as noise in high-precision timing experiments. In this paper, we show that these variations serve as a natural entropy source for the creation of Random Number Generators (RNG). We also explore the effects of using randomness extractors to increase the entropy of random bits extracted from Pulsar timing data. To evaluate the quality of the Pulsar RNG, we model its entropy as a *k*-source and use well-known cryptographic results to show its closeness to a theoretically ideal uniformly random source. To remain consistent with prior work, we also show that the Pulsar RNG passes well-known statistical tests such as the NIST test suite.

**Keywords:** pulsars, astronomy data analysis, cryptography, theoretical computer science

## 1. Introduction

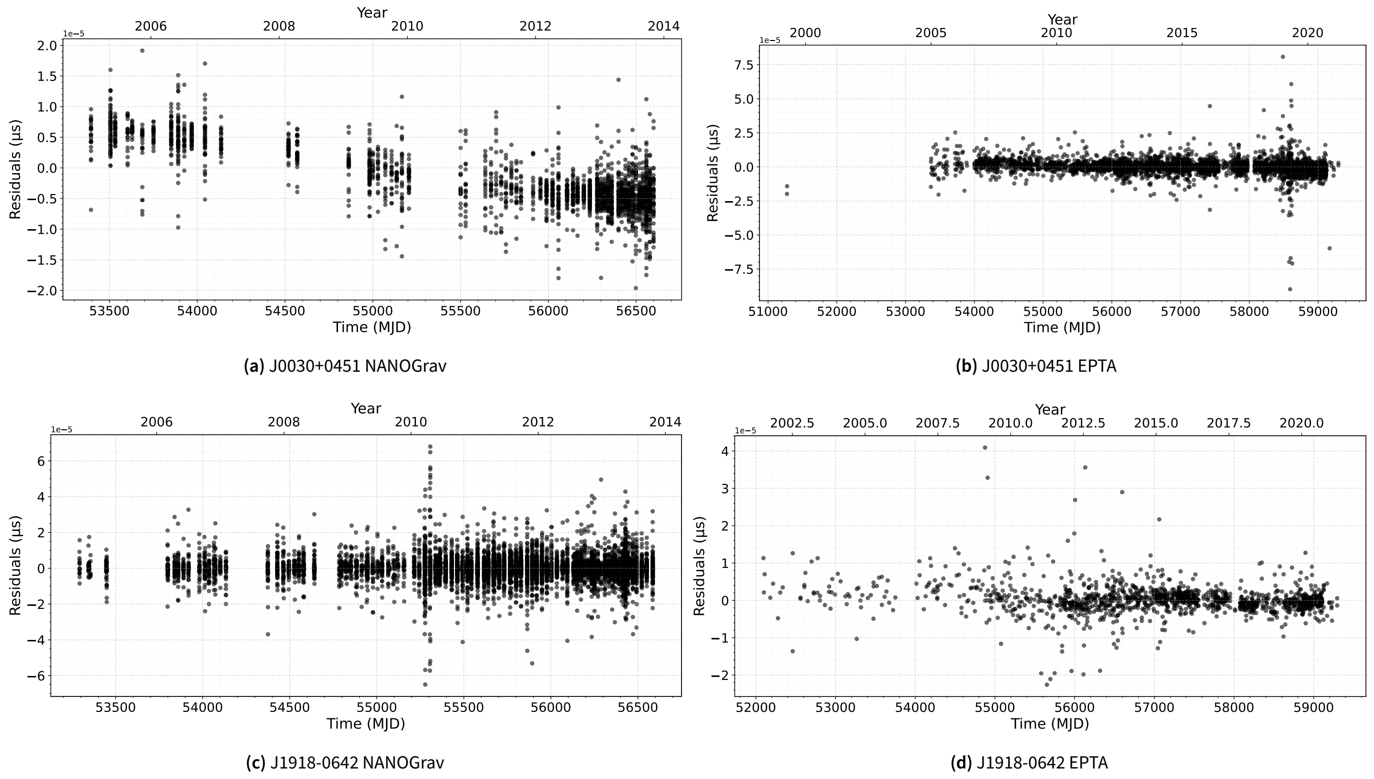
Random number generators (RNGs) are a fundamental part of modern cryptography (Katz & Lindell, 2014). They can be used to implement provably secure secret-key encryption schemes (Katz & Lindell, 2014; Pass & Shelat, 2010), digital signature schemes (Katz & Lindell, 2014), and the key generation step of public-key encryption schemes such as RSA (Rivest et al., 1978) and ElGamal (1985). True Random Number Generators (TRNGS) use noise in physical processes as a source of randomness. As an example, Intel's TRNG uses Johnson noise in resistors (Jun & Kocher, 1999). Pseudo-Random Number Generators (PRNGs) are initialized with a *seed* and use algorithms to produce numbers that seem random to adversaries that do not know the seed and are restricted to performing all their computations in probabilistic polynomial time Pass & Shelat (2010). The initial seed of a PRNG may be derived from a TRNG. Prior work on extracting randomness from astrophysical sources includes, in chronological order, hot pixels in astronomical imaging (Pimbblet & Bulmer, 2005), radio astronomy signal data noise (Chapman et al., 2016), cosmic microwave background radiation spectra (Lee & Cleaver, 2017), cosmic photon arrival times (Wu et al., 2017), and intrinsic flux density distribution of single pulsars (Dawson et al., 2022). Using a pulsar's flux density fluctuations as an entropy source has certain complicating factors including interstellar scintillation and instrumental noise such as unique Gaussian noise added by a telescope (Dawson et al., 2022).

Pulsar timing variations provide an alternative entropy source that is structured yet unpredictable. Since timing variations rely on the precise arrival times of pulsar signals instead of their observed intensity, they are comparatively less affected by interstellar scintillation. The periodic nature of pulsar signals (Hobbs et al., 2019) allows for a well-defined baseline against which small, astrophysical variations introduce entropy in a measurable and reproducible way (Hobbs et al., 2010). Timing measurements are also inherently less sensitive to instrumental noise than flux density measurements for pulsars. To the best of our knowledge, this paper is the first to investigate the variation in inter-arrival times of pulsar signals as a novel entropy source for cryptographic random number generation. Prior work on random number generation from astrophysical sources notably including Dawson et al. (2022) and Pimbblet & Bulmer (2005) has relied primarily on black-box statistical testing to evaluate randomness quality. There are well-known concerns (Saarinen, 2022) with relying solely on such statistical tests without a proper theoretical analysis of the entropy source. In fact, the statistical tests endorsed by NIST can be passed even by weak PRNGs (Saarinen, 2022). Unlike prior work, our work also includes a theoretical analysis using known cryptographic techniques to complement our empirical findings.

The rest of this paper is structured as follows. In Section 2, we create a Pulsar RNG from observational data from two sources: the North American Nanohertz Observatory for Gravitational Waves (NANOGrav) (Matthews et al., 2016) and the European Pulsar Timing Array (EPTA) (EPTA et al., 2024). Section 3 evaluates the Pulsar RNG using a cryptographic analysis and statistical tests. Section 4 provides discussion relevant to the viability of the Pulsar RNG.

## 2. Generating Random Bits

We use measurement data from two pulsars, PSR J0030+0451 and PSR J1918-0642. These two pulsars are present in both the NANOGrav 9-year dataset release (Jun & Kocher, 1999) and the EPTA DR2 dataset release (EPTA et al., 2024). Our Pulsar RNG extracts timing residuals from these datasets using PINT (Luo et al., 2021) v1.1.1. Let $L$ be the list of pulsar residuals where $L_i$ is the ith element. We first normalize the residual



(**a**) J0030+0451 NANOGrav

(**b**) J0030+0451 EPTA

(**c**) J1918-0642 NANOGrav

(**d**) J1918-0642 EPTA

**Figure 1.** Timing Variations in $\mu s$ for the J0030+0451 and J1918-0642 pulsars with Modified Julian Date (MJD) and Year plotted on the $x$ axis

values to create list $N$ in the usual way ($N_i = \frac{L_i - \min(L)}{\max(L) - \min(L)}$). We then investigate three quantification techniques on the list $N$ to convert it to a list of random bits $R$.

1. A simple threshold: $R_i = 1$ if $N_i \geq 0.5$, otherwise $R_i = 0$
2. 8-bit Gray coding (Doran, 2007)
3. Using the 8-bit Gray coded value as a seed for a SHA-512 hash (Penard & Van Werkhoven, 2008)

Figure 2 shows the measured dataset entropy in bits per byte of these three quantification methods. Note that by *entropy* throughout this paper we mean information entropy, also known as Shannon entropy. We measure all dataset entropy results in this paper using the `ent` tool (Walker, 2008).

Using threshold as a quantification method requires a careful choice of where to put the threshold based on each distribution. Our method of uniformly using $\tau = 0.5$ as the threshold provides vastly different results for, as an example, the PSR J1918-0642 data on the EPTA dataset as opposed to the NANOGrav dataset. This is due to $\tau = 0.5$ not providing an equal direction of the EPTA data. This can be verified in Figure 3 which shows normalized residuals ($N$) for both datasets. Notice that while the points on the NANOGrav data are roughly equally divided by a cutoff line at 0.5 (marked by a dotted line in the figure), most points in the EPTA dataset are below 0.5.

We next investigate the effect of three different randomness extractors on our results. Randomness Extractors are functions that take as input 1) a comparatively small uniformly-random

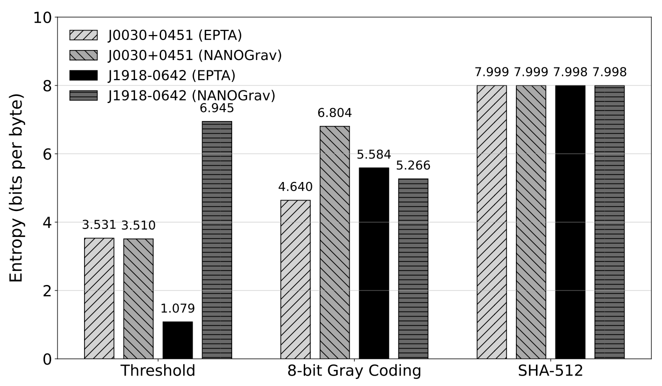


**Figure 2.** Entropy of Different Quantification Methods for 2 Pulsars across EPTA and NANOGrav data

seed and 2) a comparatively weak entropy source, for example, radioactive decay (Walker, 2001) or in our case Pulsar timing variation. Randomness Extractors output random bits that appear to computationally bound adversaries as being independent from the input entropy source and uniformly randomly distributed. Note that prior astrophysics-based RNG papers including Pimbblet & Bulmer (2005) refer to randomness extractors as debiasing or deskewing algorithms. We test two simple *ad hoc* randomness extractors, XOR-ing several subsequent bits (Stipčević & Koç, 2014) and Von Neumann (1963). We also test a Randomness extractor based on SHAKE-256 from the SHA-3 family of cryptographic hash functions. Figure 4 shows our results. While using a cryptographic hash

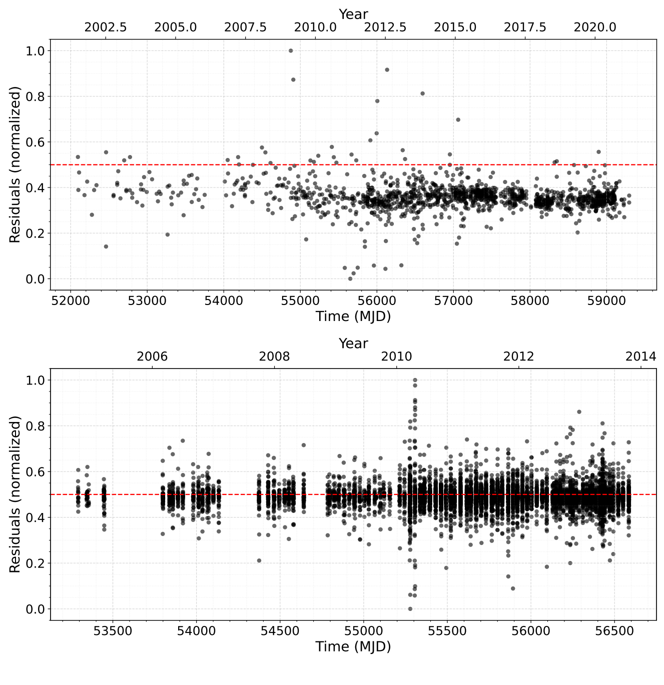


**Figure 3.** Normalized PSR J1918-0642 residuals on EPTA data (above) and NANOGrav data (below).

yields the highest entropy, it is interesting to note that even an ad-hoc random extractor like Von Neumann provides considerable entropy gains.

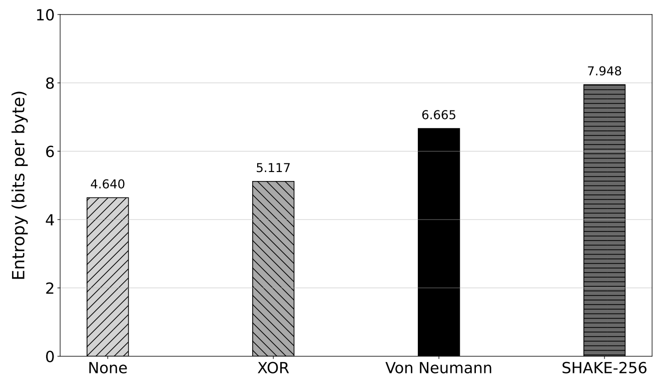


**Figure 4.** Entropy for different randomness extractors on data from PSR J0030+0451 (EPTA).

## 3. Evaluation

A strict mathematical proof of absolute randomness is considered impossible (Stipčević & Koç, 2014). To analyze TRNGs, we must rely on assumptions based on the fundamental postulates of physics (Stipčević & Koç, 2014) in combination with our mathematical analysis. We define randomness extractors and *k*-sources using standard cryptographic definitions (See Appendix 1 for details). We use our definitions to show the suitability of Pulsar RNG under a reasonable physical assumption. We assume that pulsar timing variations exhibit non-trivial entropy and can be modeled as a k-source (Assumption 1). From a theoretical standpoint, this assumption aligns with existing stochastic models Antonelli et al. (2023) of pulsar timing variations due to non-deterministic phenomena such as glitches (Zubieta, E. et al., 2024), as well as the presence of Gravitational Waves (Agazie et al., 2023).

We also empirically verify our assumption based on Pulsar data from NANOGrav and EPTA. We show our empirical results in Table 1. We generate binary arrays from 10 different Pulsars, 5 in the NANOGrav dataset and 5 in the EPTA dataset respectively. Then we measure the min-entropy (Definition 3) of generated binary arrays in units of bits per bit. By a non-trivial min-entropy, we mean a min-entropy value significantly larger than 0. By definition, the min-entropy over a binary array will be in the range $[0, 1]$ in bits per bit. We rely on the 8-bit Gray coding method for quantification that we discussed in Section 2.

**Table 1.** Empirical results validating non-trivial (significantly larger than 0) min-entropy for 10 pulsars, 5 from NANOGrav and 5 from EPTA. Note that the maximum possible min-entropy is 1.

| Pulsar | Dataset | Min Entropy (bits per bit) |
|---|---|---|
| PSR J0030+0451 | EPTA | 0.974 |
| PSR J1918-0642 | EPTA | 0.826 |
| PSR J2124-3358 | EPTA | 0.801 |
| PSR J1843-1113 | EPTA | 0.911 |
| PSR J2322+2057 | EPTA | 0.699 |
| PSR J1832-0836 | NANOGrav | 0.679 |
| PSR J2302+4442 | NANOGrav | 0.909 |
| PSR J0030+0451 | NANOGrav | 0.882 |
| PSR J1918-0642 | NANOGrav | 0.739 |
| PSR J1012+5307 | NANOGrav | 0.798 |

### 3.1 *Cryptographic Guarantees*

We show that our Pulsar RNG satisfies the conditions of a strong extractor under the Leftover Hash Lemma. Randomness extractors are cryptographic primitives that can transform an entropy source with bias into a (in practice) uniformly random distribution. The Leftover Hash Lemma formally proves that a universal hashing family can extract nearly uniform bits from a *k*-source. For the hash function that performs this debiasing in Pulsar RNG, we use SHAKE-256 from the SHA-3 family of cryptographic hash functions. The formal proof is provided in Appendix 1. Informally, this result implies that random bits generated by Pulsar RNGs are statistically close to random bits sampled from some ideal uniformly random distribution. More precisely, the statistical distance between the output of Pulsar RNG and a uniformly random distribution is bounded by a suitable $\varepsilon$.

### 3.2 *Statistical Tests*

Previous analyses of RNGs derived from astrophysical sources rely on black-box statistical tests such as the NIST SP800-22b test (Bassham III et al., 2010), `ent` (Walker, 2008), `diehard`, and `dieharder` (Brown et al., 2018). In Section 3.2, we show that our Pulsar RNG performs well when evaluated using such statistical tests and provide a discussion regarding debiasing and mixing methods. We note, however, that presenting results for these black-box statistical tests as sole evidence for the suitability of cryptographic RNGs is inaccurate (Saarinen, 2022). Even weak (insecure) PRNGs can pass these tests (Saarinen, 2022). Therefore, we recommend using our statistical test results only as complementary evidence to our theoretical claims. We show NIST SP800-22b results for the complete version of our Pulsar RNG, including SHA-512 quantification and SHAKE-256 randomness extraction, on the NIST Statistical Testing suite. We test 1 million generated bits evaluated as 10 bitstreams of 100K bits each. Table 2 shows the results for PST J0030+0451 on EPTA Data.

**Table 2.** Statistical Test Results for Pulsar RNG on PSR J0030+0451 (EPTA).

| NIST test | Proportion | P-value | Pass |
|---|---|---|---|
| Frequency | 10/10 | 0.911413 | Y |
| BlockFrequency | 10/10 | 0.911413 | Y |
| CumulativeSums | 10/10 | 0.534146 | Y |
| Runs | 10/10 | 0.213309 | Y |
| LongestRun | 10/10 | 0.534146 | Y |
| Rank | 10/10 | 0.534146 | Y |
| FFT | 10/10 | 0.534146 | Y |
| ApproximateEntropy | 10/10 | 0.122325 | Y |
| Serial | 10/10 | 0.017912 | Y |
| LinearComplexity | 10/10 | 0.004301 | Y |

The Pulsar RNG passes all tests in NIST SP800-22b. NIST SP800-22b compares a given bit stream to the null hypothesis of a uniformly random distribution of binary bits (Dawson et al., 2022). The frequency test checks the fraction of 0s and 1s in the bit stream. The block frequency test checks the same fraction but for segments or *blocks* of the bit stream. The cumulative sum test checks whether the cumulative sum of the bits in the bit stream follows a random walk. The runs test checks the maximum length of consecutive 0s or 1s. The longest runs of ones test checks the maximum length of consecutive 1s in blocks of the bit stream. The Fast Fourier Transform (Heideman et al., 1984) test, FFT for short, checks if there are any repeating patterns in the bit stream. The Approximate Entropy test checks the frequency of all possible overlapping m-bit patterns across the entire sequence. The Serial test focuses on the frequency of all possible overlapping m-bit patterns in the bit stream. Lastly, the Linear Complexity test focuses on the length of a linear feedback shift register (LFSR) to determine whether or not the sequence is complex enough to be considered random (Bassham III et al., 2010).

## 4. Discussion

We have demonstrated the viability of pulsar timing variations as an entropy source for RNGs. Theoretically, we have proved the existence of pulsar-based strong randomness extractors based on reasonable physical assumptions. Experimentally, we have verified the quality of our Pulsar RNG using various standard statistical tests. When compared to TRNGs based on noise in electronic devices, such as Johnson noise in resistors (Tyson, 2013), Pulsar RNGs are immune to local temperature fluctuations and other local environmental factors. Pulsar timing variation data is also publicly available from many sources including the North American Nanohertz Observatory for Gravitational Waves (Agazie et al., 2023), the European Pulsar Timing Array (EPTA et al., 2024), the Chinese Pulsar Timing Array Xu et al. (2023), and the Parkes Pulsar Timing Array (Manchester et al., 2013) in Australia. Unlike most Quantum RNGs (Ma et al., 2016), our Pulsar RNG does not require specialized hardware and uses this publicly available data.

In addition to cryptography, our Pulsar RNG is also suitable for many other applications. RNGs are used in Monte Carlo simulations to generate random variates from the underlying distributions of input variables (Raychaudhuri, 2008). This random variate generation process is, in fact, the core of the Monte Carlo simulation. RNGs are also used to implement probabilistic data structures such as Bloom Filters (Bloom, 1970), Skip Lists (Pugh, 1990), and Sketches (Cormode & Muthukrishnan, 2005). Other uses of RNGs include Machine Learning algorithms (Mitchell, 1997) and even artwork (Bauer, 1998). We have released a fully open-source Python implementation of our Pulsar RNG. Our implementation contains a usable tool to generate random numbers from pulsar data under multiple configurations. The tool currently supports NANOGrav and EPTA data but our modular implementation makes the tool easy to extend for other public datasets. In addition to our tool, we have also open sources all our data processing scripts, randomness extraction methods, and evaluation code. Lastly, we have also publicly released the raw bitstreams we generated to allow an independent verification of our results. We have made all the discussed artifacts available at github.com/jadidbourbaki/pulsar_rng.

Many open problems emerge from this work. We observe (Table 1) that different pulsars yield different entropy. There are over 3000 known pulsars and a comprehensive study would provide a better understanding of the min-entropy and entropy distributions of pulsar timing variations in generation. The deployment of pullsar-based RNGs in real-work applications will also demonstrate practical advantages or challenges our analysis does not address.

Funding Statement The authors have no relevant funding sources to declare.

Competing Interests The authors have no relevant competing interests.

Data Availability Statement The datasets we analyze in this paper are publicly available from the NANOGrav and EPTA collaborations. The open-source Pulsar RNG tool, the extracted bitstreams, the code used to process pulsar timing data and evaluate randomness properties, are all available at github.com/jadidbourbaki/pulsar_rng

**Ethical Standards** The research meets all ethical guidelines, including adherence to the legal requirements of the study country.

**Author Contributions** Conceptualization: H.T. Methodology: H.T. Data curation: H.T. Data vvisualization H.T. Writing original draft: H.T. All authors approved the final submitted draft.

## References

Agazie, G., Anumarlapudi, A., Archibald, A. M., et al. 2023, The Astrophysical Journal Letters, 951, L8

Antonelli, M., Basu, A., & Haskell, B. 2023, Monthly Notices of the Royal Astronomical Society, 520, 2813

Bassham III, L. E., Rukhin, A. L., Soto, J., et al. 2010, Sp 800-22 rev. 1a. a statistical test suite for random and pseudorandom number generators for cryptographic applications

Bauer, A. 1998, Gallery of random art, https://www.random-art.org/

Bloom, B. H. 1970, Commun. ACM, 13, 422–426

Brown, R. G., Eddelbuettel, D., & Bauer, D. 2018, Duke University Physics Department Durham, NC, 27708

Chapman, E., Grewar, J., & Natusch, T. 2016, Australian Information Security Management Conference

Cormode, G., & Muthukrishnan, S. 2005, Journal of Algorithms, 55, 58

Dawson, J., Hobbs, G., Gao, Y., et al. 2022, Astronomy and Computing, 38, 100549

Doran, R. W. 2007, The gray code, Tech. rep., Citeseer

ElGamal, T. 1985, in Advances in Cryptology, ed. G. R. Blakley & D. Chaum (Berlin, Heidelberg: Springer Berlin Heidelberg), 10–18

EPTA, InPTA, et al. 2024, Astronomy and Astrophysics, 690, A118

Heideman, M., Johnson, D., & Burrus, C. 1984, IEEE Assp Magazine, 1, 14

Hobbs, G., Archibald, A., Arzoumanian, Z., et al. 2010, Classical and Quantum Gravity, 27, 084013

Hobbs, G., Guo, L., Caballero, R. N., et al. 2019, Monthly Notices of the Royal Astronomical Society, 491, 5951–5965

Impagliazzo, R., Levin, L. A., & Luby, M. 1989, in Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing, STOC '89 (New York, NY, USA: Association for Computing Machinery), 12–24

Jun, B., & Kocher, P. 1999, Cryptography Research Inc. white paper, 27, 66

Katz, J., & Lindell, Y. 2014, Introduction to Modern Cryptography, Second Edition, 2nd edn. (n/a: Chapman & Hall/CRC)

Lee, J. S., & Cleaver, G. B. 2017, Heliyon, 3, e00422

Luo, J., Ransom, S., Demorest, P., et al. 2021, ApJ, 911, 45

Ma, X., Yuan, X., Cao, Z., Qi, B., & Zhang, Z. 2016, npj Quantum Information, 2, 1

Manchester, R. N., Hobbs, G., Bailes, M., et al. 2013, PASA, 30, e017

Matthews, A. M., Nice, D. J., Fonseca, E., et al. 2016, The Astrophysical Journal, 818, 92

Mitchell, T. 1997, Machine learning, 7, 2

Pass, R., & Shelat, A. 2010, A Course in Cryptography, Lecture Notes, available at: https://www.cs.cornell.edu/courses/cs4830/2010fa/lecnotes.pdf

Penard, W., & Van Werkhoven, T. 2008, Cryptography in context, 1

Pimbblet, K. A., & Bulmer, M. 2005, Publications of the Astronomical Society of Australia, 22, 1–5

Pugh, W. 1990, Commun. ACM, 33, 668–676

Raychaudhuri, S. 2008, in 2008 Winter simulation conference, IEEE, 91–100

Reyzin, L. 2011, Lecture Notes for CS 937: Advanced Topics in Cryptography, spring 2011. Available online: https://www.cs.bu.edu/~reyzin/teaching/s11cs937/notes-leo-1.pdf

Rivest, R. L., Shamir, A., & Adleman, L. 1978, Commun. ACM, 21, 120–126

Saarinen, M.-J. O. 2022, in 2022 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW) (Los Alamitos, CA, USA: IEEE Computer Society), 31–37

Stipčević, M., & Koç, Ç. K. 2014, in Open problems in mathematics and computational science (Springer), 275–315

Tyson, T. 2013, Thermal Johnson Noise Generated by a Resistor, https://123.physics.ucdavis.edu/week_2_files/Johnson_noise_intro.pdf

Von Neumann, J. 1963, John von Neumann, Collected Works, 5, 768

Walker, J. 2001, Online: http://www. fourmilab. ch/hotbits

—. 2008, ENT: A Pseudorandom Number Sequence Test Program. Fourmilab: Switzerland, 2008

Wu, C., Bai, B., Liu, Y., et al. 2017, Phys. Rev. Lett., 118, 140402

Xu, H., Chen, S., Guo, Y., et al. 2023, Research in Astronomy and Astrophysics, 23, 075024

Zubieta, E., García, F., del Palacio, S., et al. 2024, A&A, 689, A191

## Appendix 1. Formal Definitions & Proofs

Given set $S$, we write $x \xleftarrow{\$} S$ to mean that $x$ is sampled uniformly randomly from $S$. For set $S$, we denote by $|S|$ the number of elements in $S$. The same notation is used for a list $\mathcal{L}$. We write variable assignments using $\leftarrow$. If the output is the value of a randomized algorithm, we use $\xleftarrow{\$}$ instead. For a randomized algorithm A, we write output $\leftarrow \mathrm{A}_r(\mathrm{input}_1, \mathrm{input}_2, \cdots, \mathrm{input}_l)$, where $r \in \mathcal{R}$ are the random coins used by A and $\mathcal{R}$ is the set of possible coins. We consider strings $\{0,1\}^n$ to be elements of the Galois Field GF($2^n$). We shorten random variables to r.v. We assume all adversaries are computationally bound. More precisely, we assume adversaries are restricted to non-uniform probabilistic polynomial time (Pass & Shelat, 2010).

**Definition 1** (Statistical Distance $\Delta$). *Let $X, Y$ be r.v.s with range $U$.*

$$\Delta(X, Y) = \frac{1}{2}\Sigma_{u \in U}|P[X = u] - P[Y = u]|$$

.

**Definition 2** ($\varepsilon$-close). *Let $X, Y$ be r.v.s with range $U$.*

$$X \approx_\varepsilon Y \equiv \Delta(X, Y) \leq \varepsilon$$

**Definition 3** (Min-entropy). *Let $X$ be an r.v. with range $U$.*

$$H_\infty(X) = -\log_2(\max_{u \in U} P[X = u])$$

**Definition 4** (k-source). *R.v $X$ is a k-source if $H_\infty(X) \geq k$*

We base our analysis on the following assumption regarding Pulsar timing variations.

**Assumption 1.** *Let $P_X$ be an r.v. representing timing variation in pulsar signals for pulsar $P$ with universe $U$. We assume $P_X$ is a k-source (Definition 4) with non-trivial k.*

We can now precisely define a randomness extractor (Reyzin, 2011) in the cryptographic sense.

**Definition 5** (Randomness-Extractor). *Let seed $U_d$ be uniformly distributed on $\{0,1\}^d$. $\mathcal{E} : \{0,1\}^n \times \{0,1\}^d \mapsto \{0,1\}^m$ is a (k, ε)-extractor if, for all k-sources X on $\{0,1\}^n$ independent of $U_d$,*

$$\mathcal{E}(X, U_d), U_d) \approx_\varepsilon (U_m, U_d)$$

*where $U_m$ is uniformly distributed on $\{0,1\}^m$ independent of X and $U_d$.*

Extractors, as defined above, are also referred to in the literature as **strong** extractors.

**Definition 6** (Universal hash family). *A family $\mathcal{H}$ of hash functions of size $2^d$ from $\{0,1\}^n$ to $\{0,1\}^m$ is called universal if, for every $x, y \in \{0,1\}^n$ with $x \neq y$,*

$$P_{h \in \mathcal{H}}[h(x) = h(y)] \leq 2^{-m}.$$

We denote our Pulsar RNG algorithm as $\mathcal{E}_p$. $\mathcal{E}_p$ relies on a universal hash family. $\mathcal{E}_p$ takes quantified data from a Pulsar entropy source $x_p \xleftarrow{\$} P_X$. It then uses a hash function from a universal hash family $h_p \xleftarrow{\$} \mathcal{H}$ of size $2^d$. In our default implementation, this is the SHAKE-256 hash function from the SHA-3 family of hashes. $\mathcal{E}_p$ then uses $p_x$ as the seed for $h_p$.

$$\mathcal{E}_p(p_x, h) = h_p(p_x)$$

There is a well-known result in cryptography called the Leftover Hash Lemma (Reyzin, 2011), originally proved by Impagliazzo et al. (1989). The Leftover Hash Lemma proves that a universal hash family can be used to construct a strong extractor from a $k$-source.

**Theorem 1** (Leftover hash lemma). *Let X be a k-source with universe U. Fix ε > 0. Let $\mathcal{H}$ be a universal hash family of size $2^d$ with output length $m = k - 2\log_2(\frac{1}{\varepsilon})$. Define*

$$\mathcal{E}(x, h) = h(x)$$

*Then $\mathcal{E}$ is a strong $(k, \varepsilon/2)$ extractor with seed length d and output length m.*

We are now ready to prove our main result, that our Pulsar RNG $\mathcal{E}_p$ is a strong extractor.

**Theorem 2.** *Let $P_X$ be an r.v. representing timing variation in pulsar signals for pulsar P with universe U. Fix ε > 0. Pulsar RNG, $\mathcal{E}_p$ is a strong $(m + 2\log_2(\frac{1}{\varepsilon}))$-extractor with seed length d and output length m.*

*Proof.* The proof follows directly from Assumption 1 and the Leftover Hash Lemma. □