JEnt (MemOnly): A Hardware-Based Ad Hoc Entropy Source

Joshua E. Hill

Version: 20230613

A Short Introduction to Randomness

Randomness is necessary for cryptographic systems:

- By Kerckhoffs's principle, we assume that adversaries know the design and implementation, thus know how secret values are selected.
- Game theory tells us that in these circumstances, the random selection of parameters yields the least advantage for the attacker.

How Many Bits Would a Bit Compressor Compress if...

- The traditional measure of uncertainty from Information Theory is called entropy.
- Shannon entropy is the most widely adopted notion of entropy.
  - It tells you the smallest average encoded message length for a source.

$$H(X) = -\sum_{i=1}^{n} p_i \lg p_i$$

What Color is Your Entropy?

- We want a worst case, not the average case.
- We get it from a generalization of Shannon Entropy called Rényi entropy.

$$H_\alpha(X) = \frac{1}{1-\alpha} \lg \left(\sum_{i=1}^{n} p_i^\alpha \right)$$

Letting $\alpha \to 1$ gives us Shannon Entropy.
Letting $\alpha = 2$ gives us Collision Entropy.
Letting $\alpha \to \infty$ gives us Min Entropy.

$$H_\infty(X) = -\lg \max_i p_i$$

What a TWIST!

- For independent distributions, these definitions are sufficient.
- In other cases, you can make the definitions conditioned on some internal state or some number of prior outputs.

NIST SP 800-90B and Me!

This document emphasizes central noise source design and analysis goals:

- The noise source must lend itself to analysis.
- A design analysis must produce a specific min entropy lower bound.
  - This bound is supposed to be derived from the developer's understanding of the noise source, and not purely based on some general-purpose statistical analysis.

Let's Get (Non-)Physical

SP 800-90B allows evaluation of Physical or Non-Physical sources.

- Physical Noise Source: "A noise source that exploits physical phenomena ... from dedicated hardware designs ... or physical experiments to produce digitized random data."
- Non-Physical Noise Source: (handwave)
  - A noise source that is not a physical noise source.

Let's Get (Non-)Physical

- Many simple physical noise sources are amenable to use of a stochastic model for generating such a bound.
- Most non-physical noise sources are not, so we need some other way to make a heuristic for this bound.

The CPU Jitter Random Number Generator (JEnt)

The CPU Jitter Random Number Generator (JEnt)

- Non-Physical Entropy Source
- Based on the difficulty that an attacker has in guessing execution timing variations due to memory access and various CPU microarchitectural features.
- Some nice characteristics that enable public discussion
  - Conceptually simple.
  - Open Source
  - Thoroughly documented by the author.
- 18 of the existing 53 ESV (SP 800-90B) certs are for JEnt on various architectures.

What “non-physical” phenomenon is “noisy”?

The JEnt design paper [Müller 2022] describes various possible causes for unpredictable timing:

- Architectural **features that can impact execution time**
  - **CPU instruction pipeline fill levels** have an impact on the execution time of one instruction.
  - **CPU frequency scaling** can alter instruction processing speed, and may depend on workload.
  - **CPU power management** can disable CPU features that impact execution time.
  - **CPU frequency scaling** depending on the workload.
  - **Translation Lookaside Buffer (TLB)**
  - **Scheduling and load balancing**
  - **Interrupt handling**

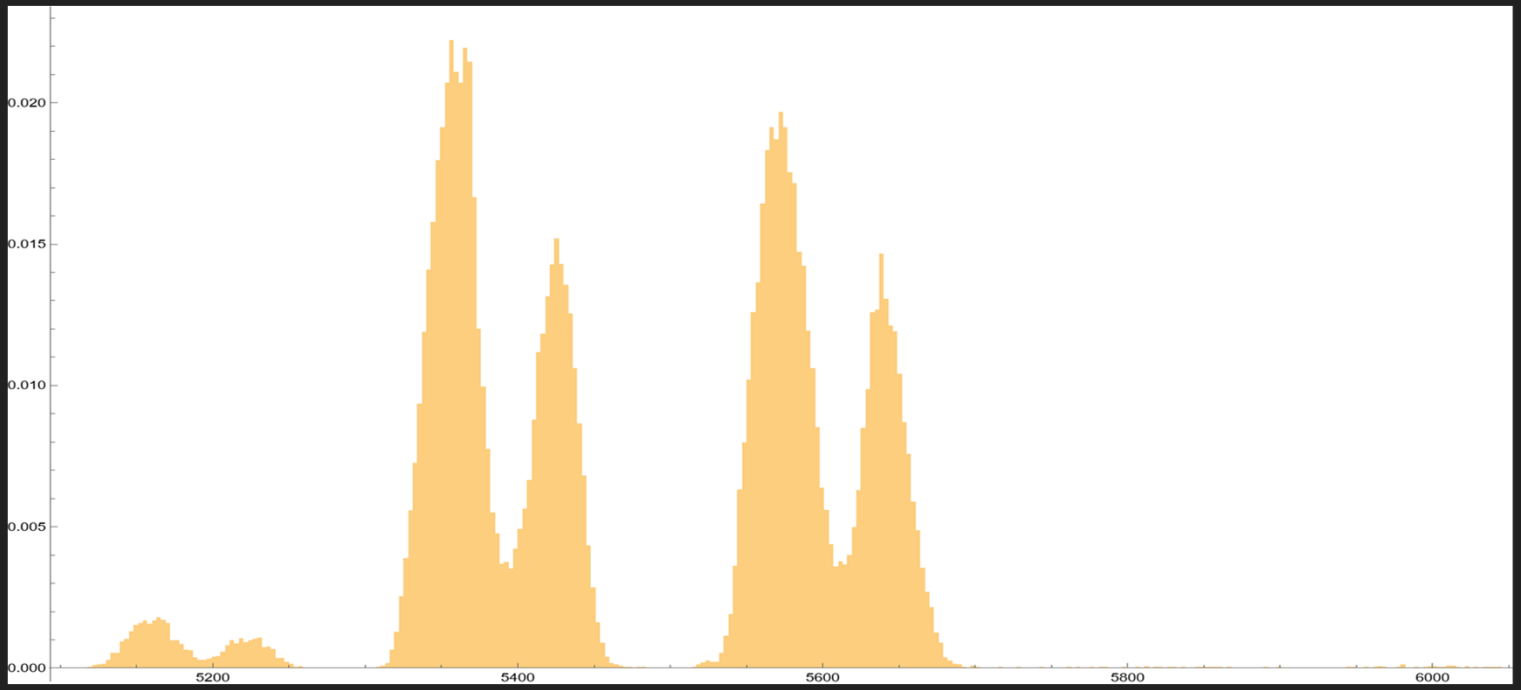
Grist For the Example Mill

Testing here was done using an Intel Xeon 6252 (Cascade Lake):

- L1 (Data cache): 32 KB
- L2 Cache: 1 MB
- L3 Cache: 36MB
- 384 GB (DDR4 Registered 2666 MHz) Memory

What “non-physical” phenomenon is “noisy”?

This is a complex set of effects that varies widely across architectures



A general approach based on these features is difficult to justify

More Noise Means… More… Better?

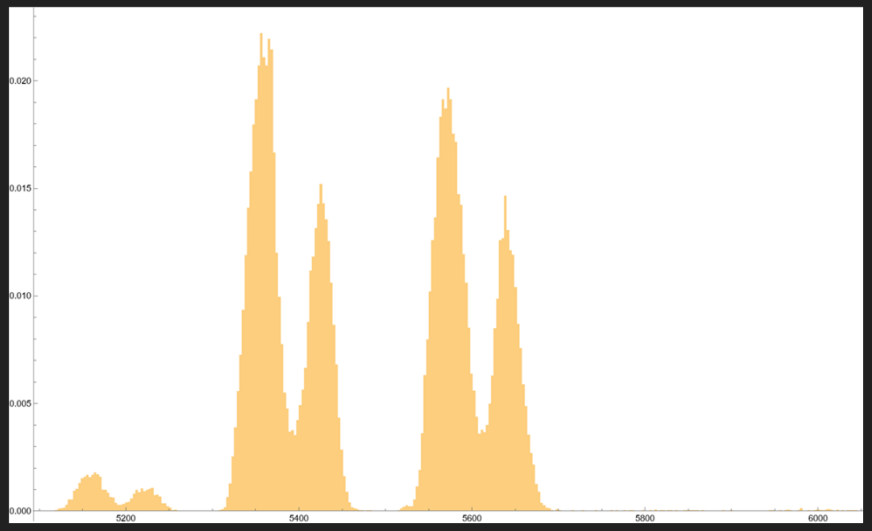
- The data in this source can be difficult to directly test.
- There's a lot going, so the resulting counter values are large.
  - Fast counters yield larger counter values and more variation.
  - More variation suggests there could be more entropy…

But…

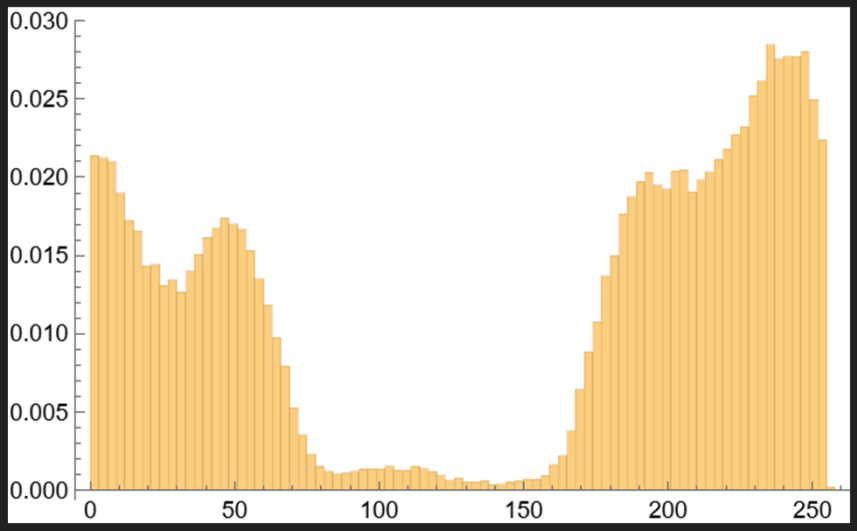
  - Large variation means we can't directly assess the data.
  - Translation further obscures what is going on…

More Noise Means… More… Better?

If we truncate to the lowest byte, we can run the SP 800-90B estimators…



Becomes



Getting to Know You SP 800-90B Entropy Estimator

- These estimators are conceptually simple.
- These estimators can only evaluate up to 8 bits of data per raw symbol.
- Many estimators operate under an IID assumption.
- Many estimators essentially extract a single extracted parameter.
- All of the estimators work better when supplied a fixed distribution.

Conclusion: Not magic boxes that output truth.

KeyPair CONSULTING

Problem Statement

- We need a simple source of variation that we think is non-deterministic.
- We would like the resulting data to reflect only this chosen source.
- We would like to be able to minimize or avoid translation or mapping.
- We want to be able to argue that the statistical testing is meaningful.

Proposed JEnt Changes

What “non-physical” phenomenon is “noisy”?

- CPU Clock trees produce domains that globally asynchronous, but locally coherent.
- Drift between asynchronous clocks can induce uncertainty.
- Different memory modules can run at different frequencies, so memory is one such domain.
- We attempt to construct a justifiable technical argument based on an approach only crediting the memory update timing.

Focus On What You Want

- The baseline JEnt raw data is influenced by both memory access timing variation and (large scale) conditioning.
- The large conditioning operation creates large timing variations.
- Memory timing also varies, but this variation is at a more modest scale.
- Prior investigation has identified variations in memory timing as a source of non-determinism.
  - (In some architectures)
- We change the primary noise source to reflect only the memory update timing.
- Other timing variation is still used as an additional noise source.

KeyPair CONSULTING

Memory Access Variation

```
static inline void jent_get_nstime(volatile uint64_t *out)
{
        unsigned int dummy;
        *out =  __rdtscp(&dummy);
        _mm_lfence();
}
```

Memory Access Variation

```
static uint64_t jent_memaccess(struct rand_data *ec)
{
... initialize
        jent_get_nstime_(mostly)(ec, &starttime);
… optionally loop
        {
                tmpval = ec->mem +
                 (xoshiro256starstar(ec->prngState.u) & MASK;
                *tmpval = (unsigned char)((*tmpval + 1) & 0xff);
        }

        jent_get_nstime_(mostly) (ec, &endtime);
… return the delta
}
```

KeyPair CONSULTING

Take Only What You Want

- Even simple sources have distinct behaviors (thus timing sub-distributions).
- These behaviors reflect how successful the caching system has been.
- We are interested timing updates that result in actual RAM I/O.
- Because the source is simple, we can classify the sub-distribution by timing alone.
- We filter the output of the primary noise source so that it contains only the identified sub-distribution.
- Other data is still used as supplemental data in the conditioner.

BUT WAIT, THERE'S MORE!

In summary:

- The primary noise source now reflects only memory timing (a single sort of event that we think may be non-deterministic… on some architectures).
- We filter the results so that we only output data from a single sub-distribution.
  - We can configure the library so that the desired sub-distribution occurs suitably often.
- Data from the primary noise source now requires less (or no) translation.

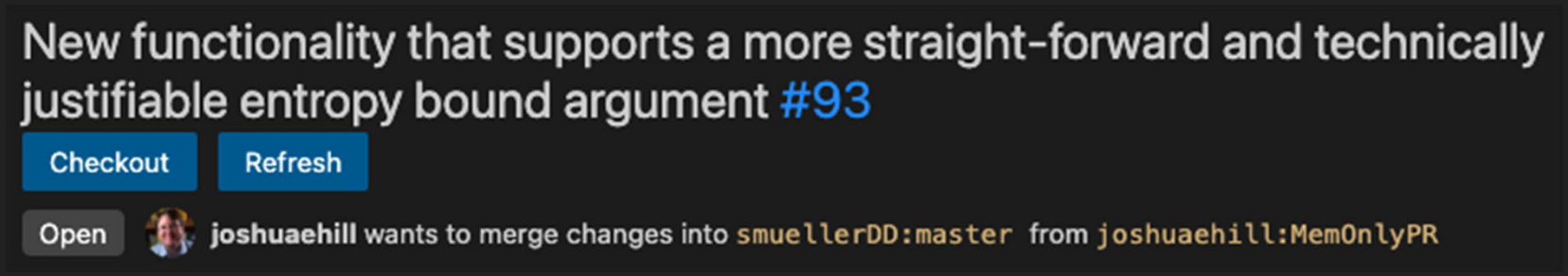
Our estimators now have a fighting chance…

(In related news, health testing is similarly more powerful.)

JEnt Codebase Flavors

We have a GitHub branch with this functionality:

[JEnt (MemOnly)]: MemOnly branch with an associated pull request



Assessment Strategies

KeyPair CONSULTING

Assessment Strategy #1
The Single Sub-Distribution Empirical Analysis Approach

- Choose parameters so that RAM I/O is likely.
- Select the desired sub-distribution.
- The resulting data now reflects the variation of a single operation, and is suitably narrow for direct analysis.
  - This simplifyies the system to the point where the 90B estimators are more likely to produce reasonable results.
- Test raw data output using the SP 800-90B estimators.

(The “Do the needful” assessment approach.)

Assessment Strategy #2
The Essentially IID Analysis Approach

This starts the same way:

- Choose parameters so that RAM I/O is likely.
- Select the desired sub-distribution.

At this point, engage the statistical fanciness…

Assessment Strategy #2
The Essentially IID Analysis Approach

- Non-IID sources have statistical memory. Internal state that induces relationships between the current output and some number of past outputs.
- The statistical memory "depth" is the number of symbols for which that state induces a significant interrelationship.
- If the memory depth is finite, we can decimate (throw away) enough data so that the remaining data acts like IID data.
    - "Thrown away" data can still be integrated into the conditioner as "supplemental data" and not credited as containing entropy.

How do we know when we've thrown away enough data?

Assessment Strategy #2
The Essentially IID Analysis Approach

Essentially, run the SP 800-90B (Section 5) IID tests… a lot…

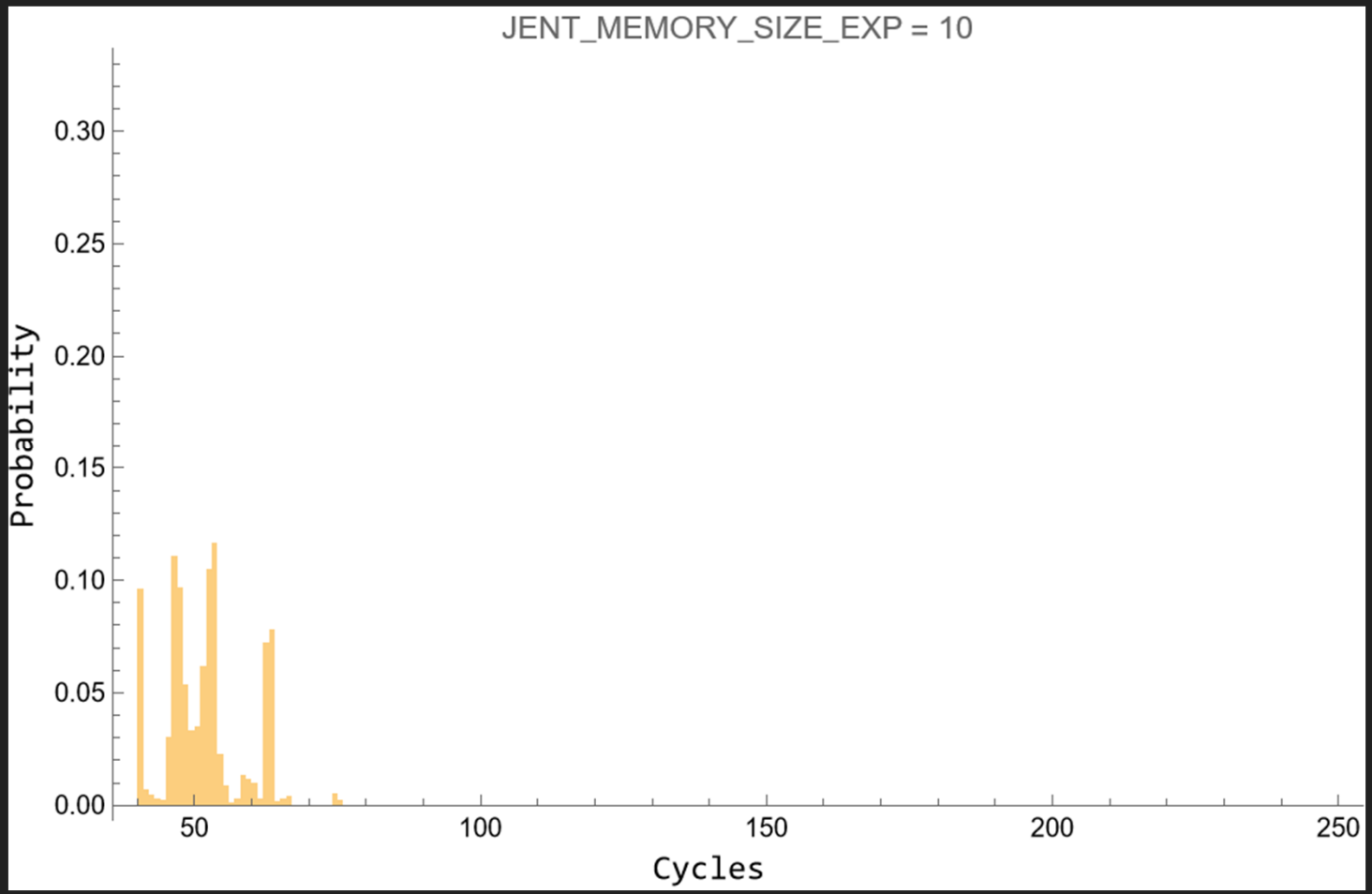
- Take many samples of data.
- Run each of the 22 tests on each of the data samples.
- Check to see if each of the 22 IID tests is passing “sufficiently often”.

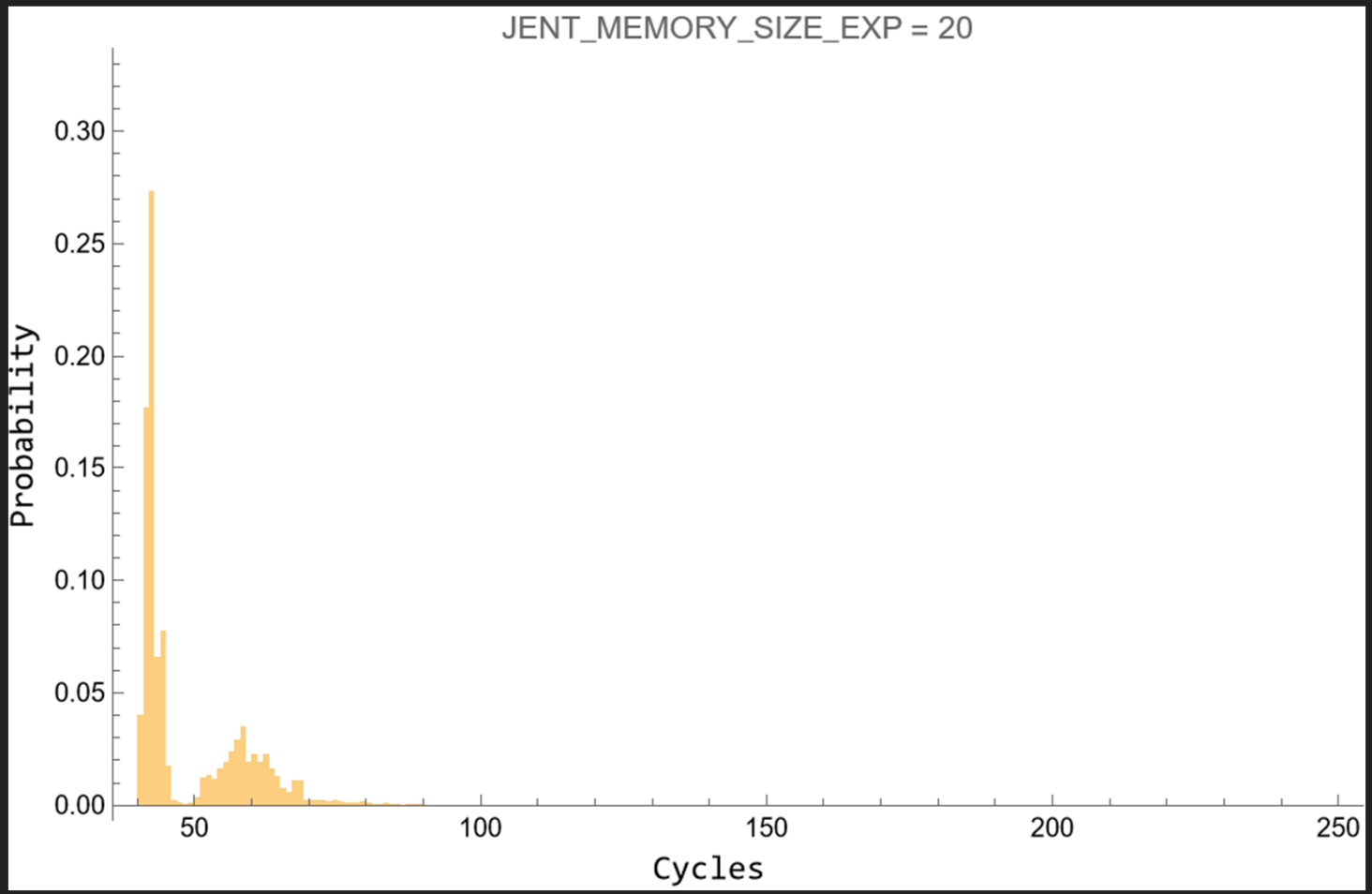
Do this for each decimation level until it works.

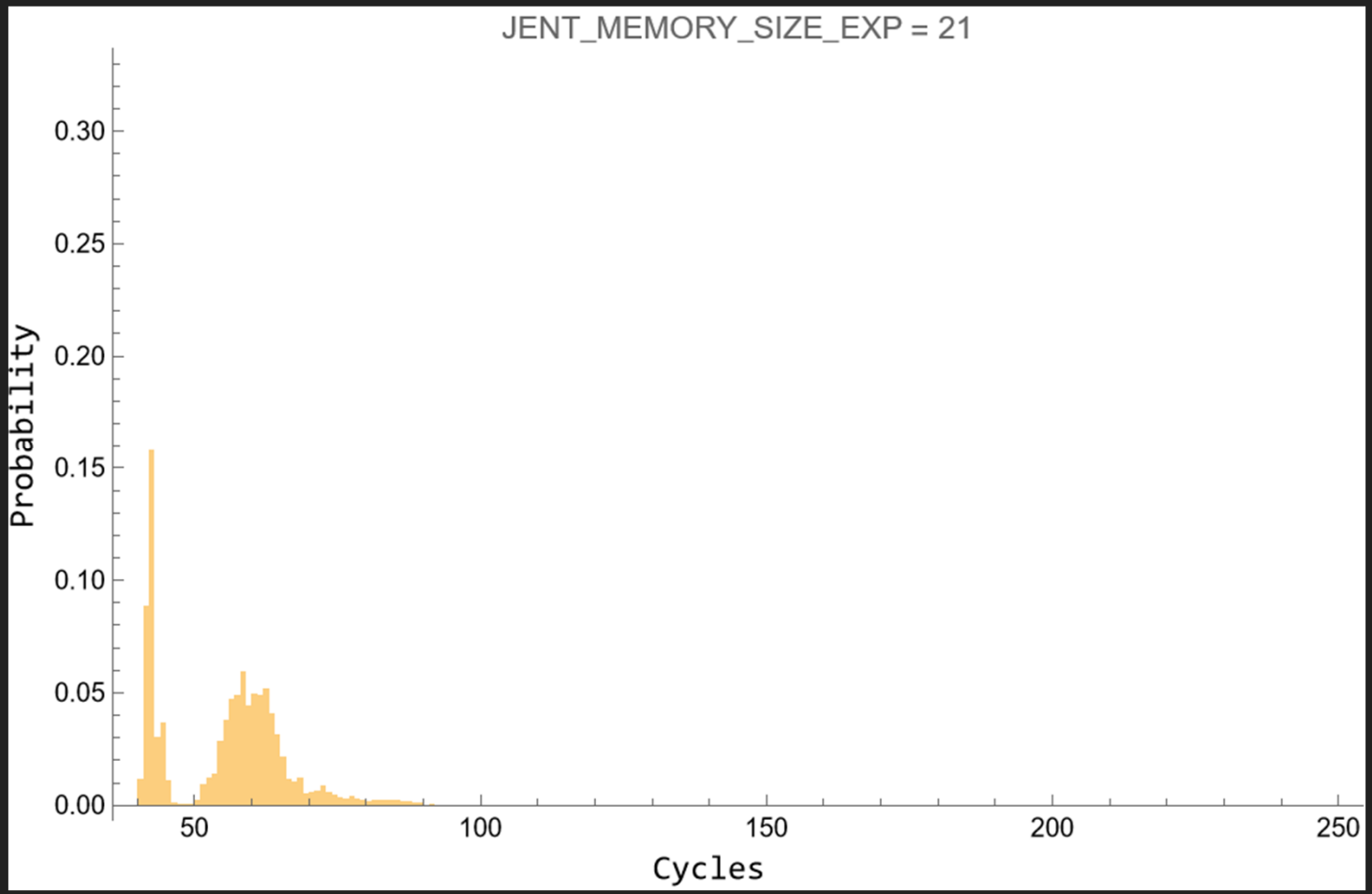
Assessment Strategy #2
The Essentially IID Analysis Approach

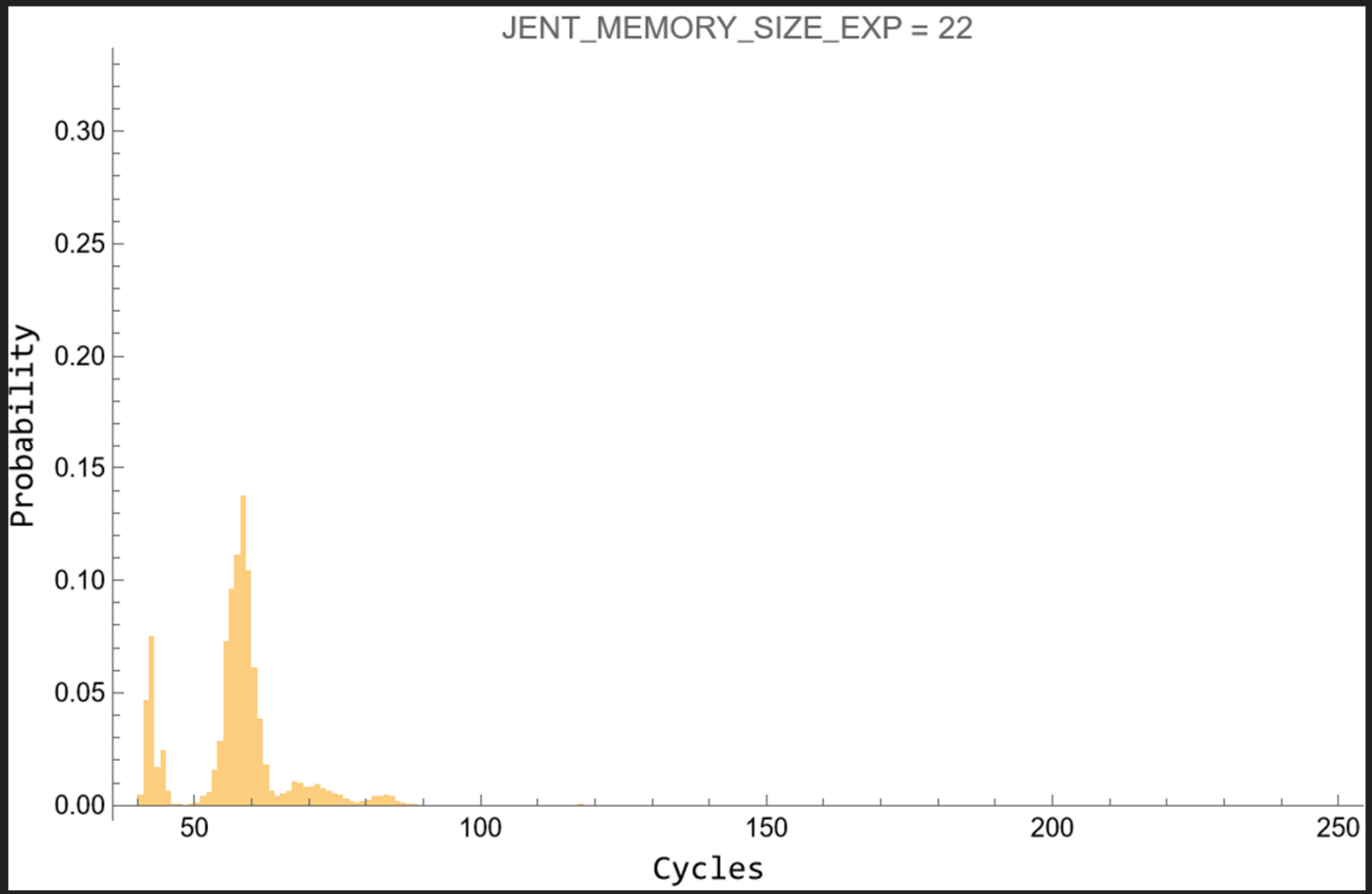
- Once you are decimating sufficiently, you can estimate $H_{submitter}$ in the obvious way.
- It is probably best (and likely required) that you still don't make an overall IID claim in the SP 800-90B assessment.
  - There is no general-purpose design-oriented reason this ought to be an IID source!

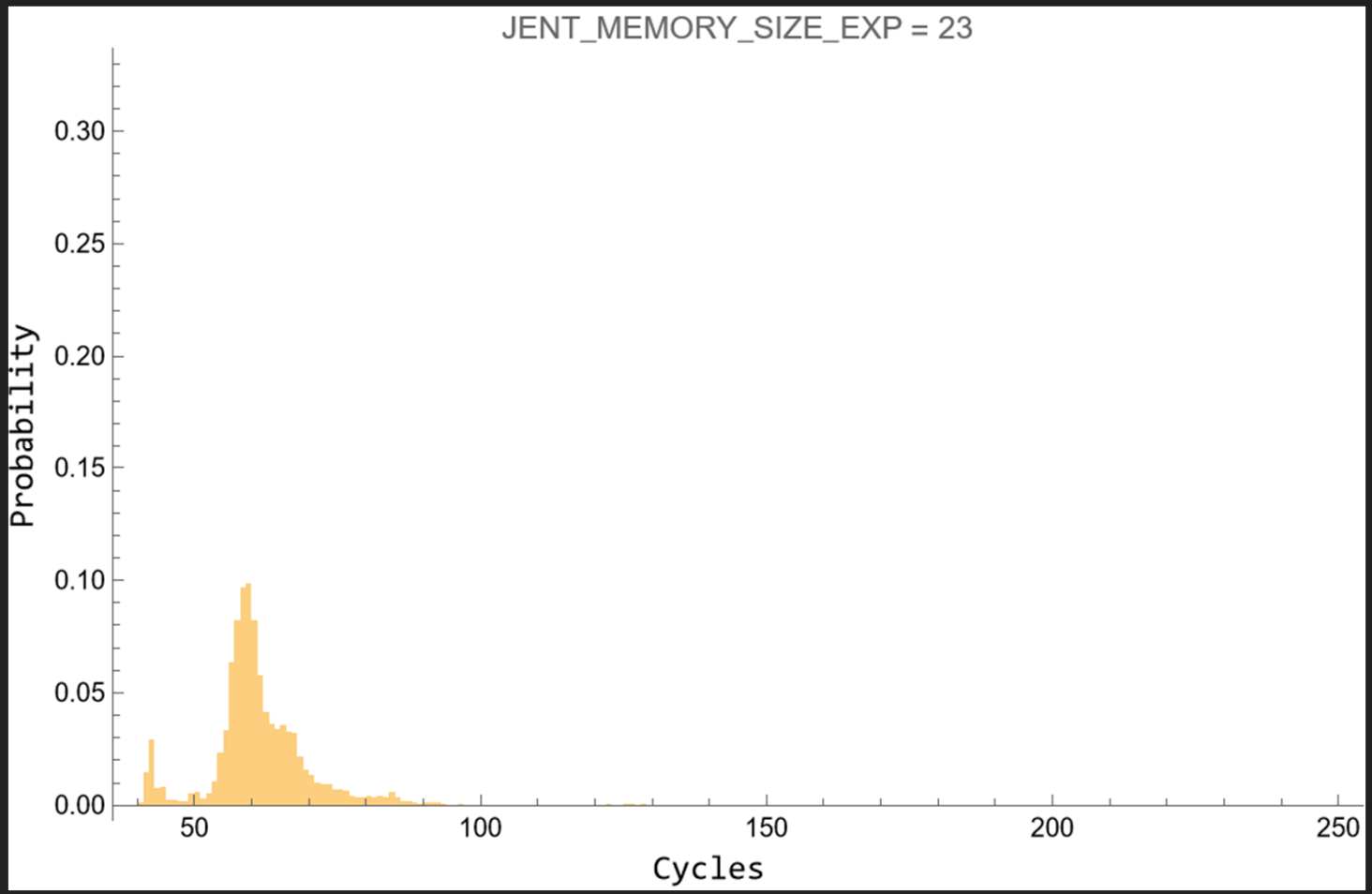
A Worked Example

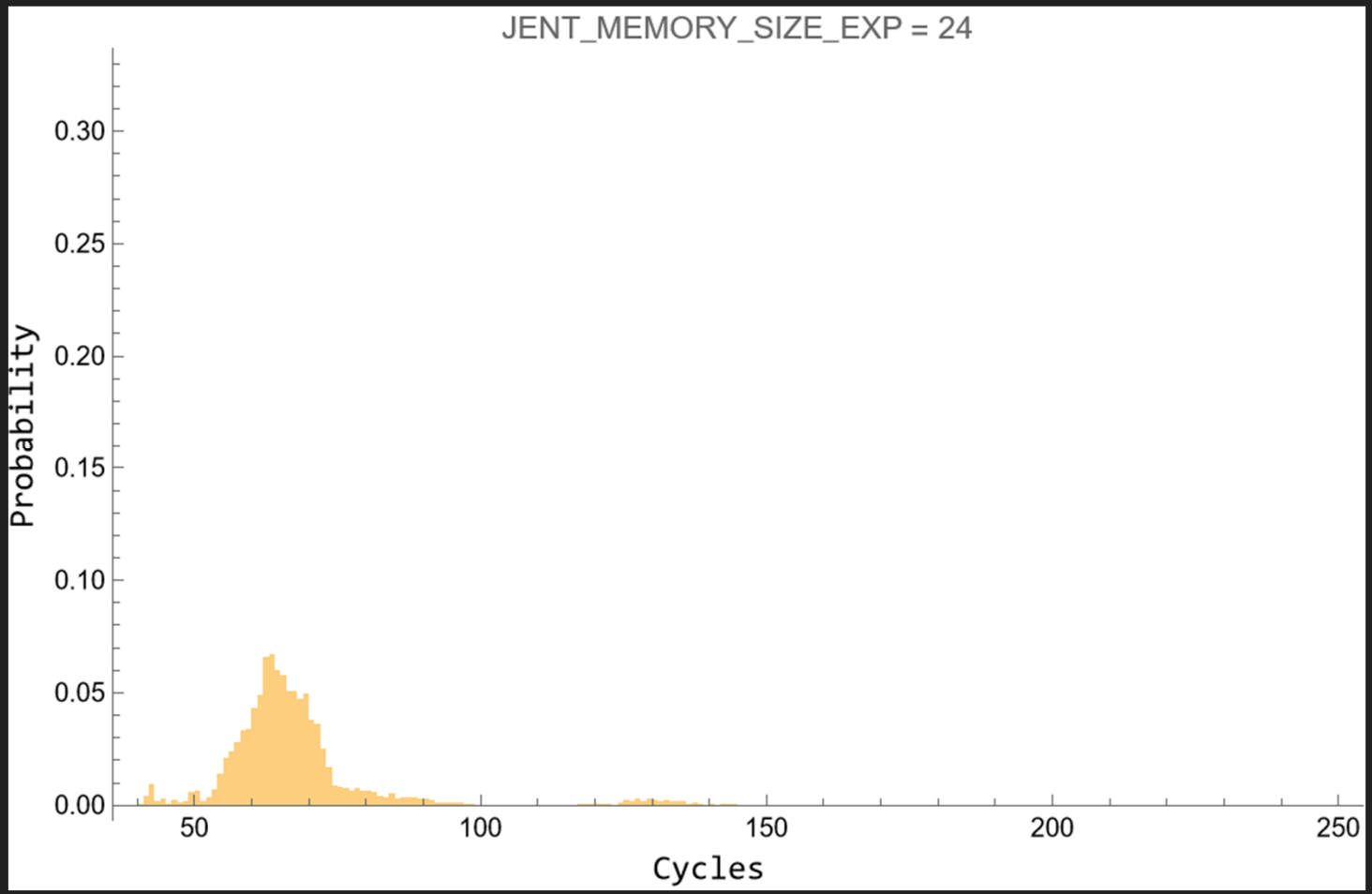

JENT_MEMORY_SIZE_EXP = 10
Probability
0.30
0.25
0.20
0.15
0.10
0.05
0.00
50
100
150
200
250
Cycles

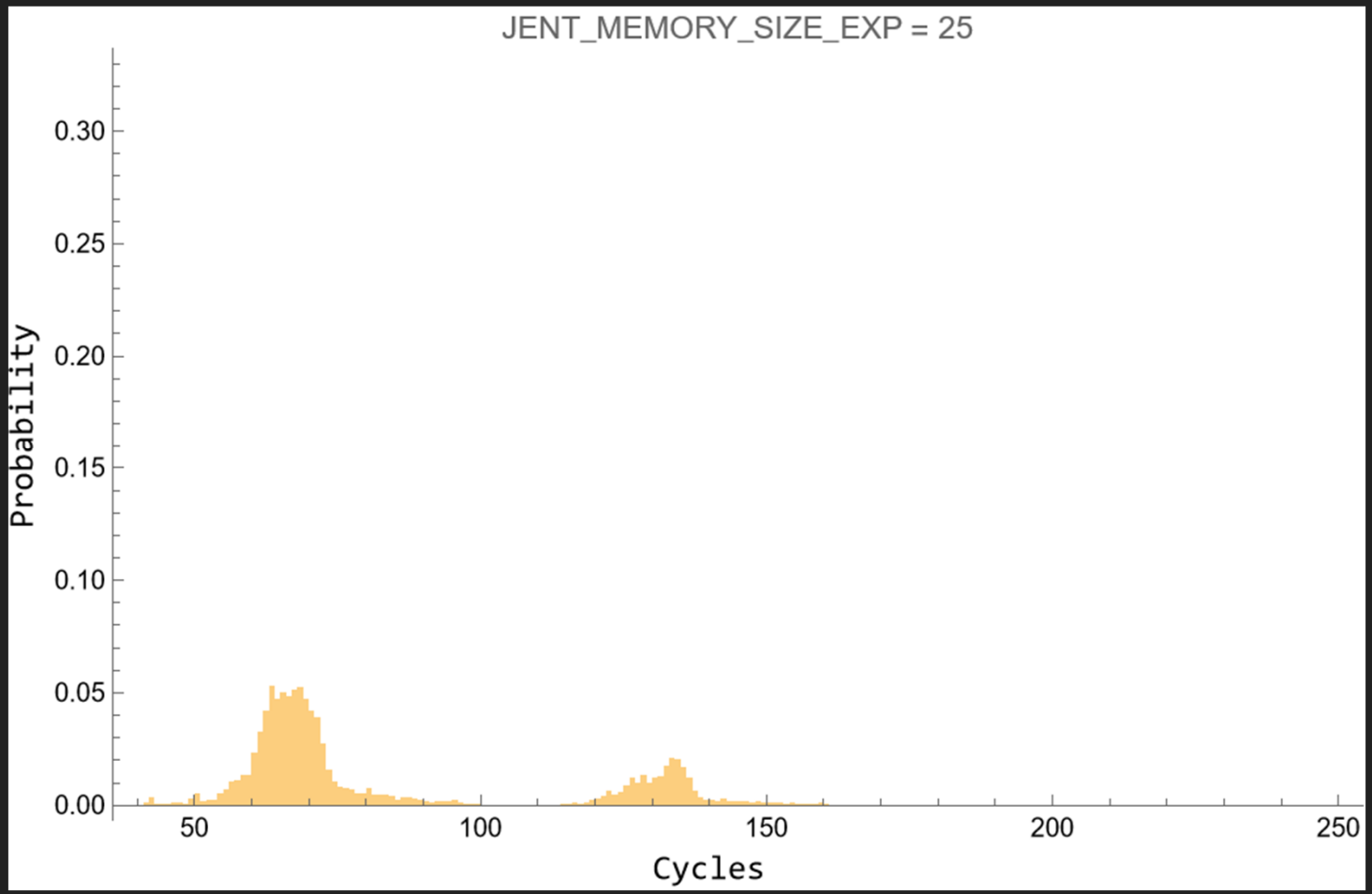

JENT_MEMORY_SIZE_EXP = 20
Probability
0.30
0.25
0.20
0.15
0.10
0.05
0.00
50
100
150
200
250
Cycles

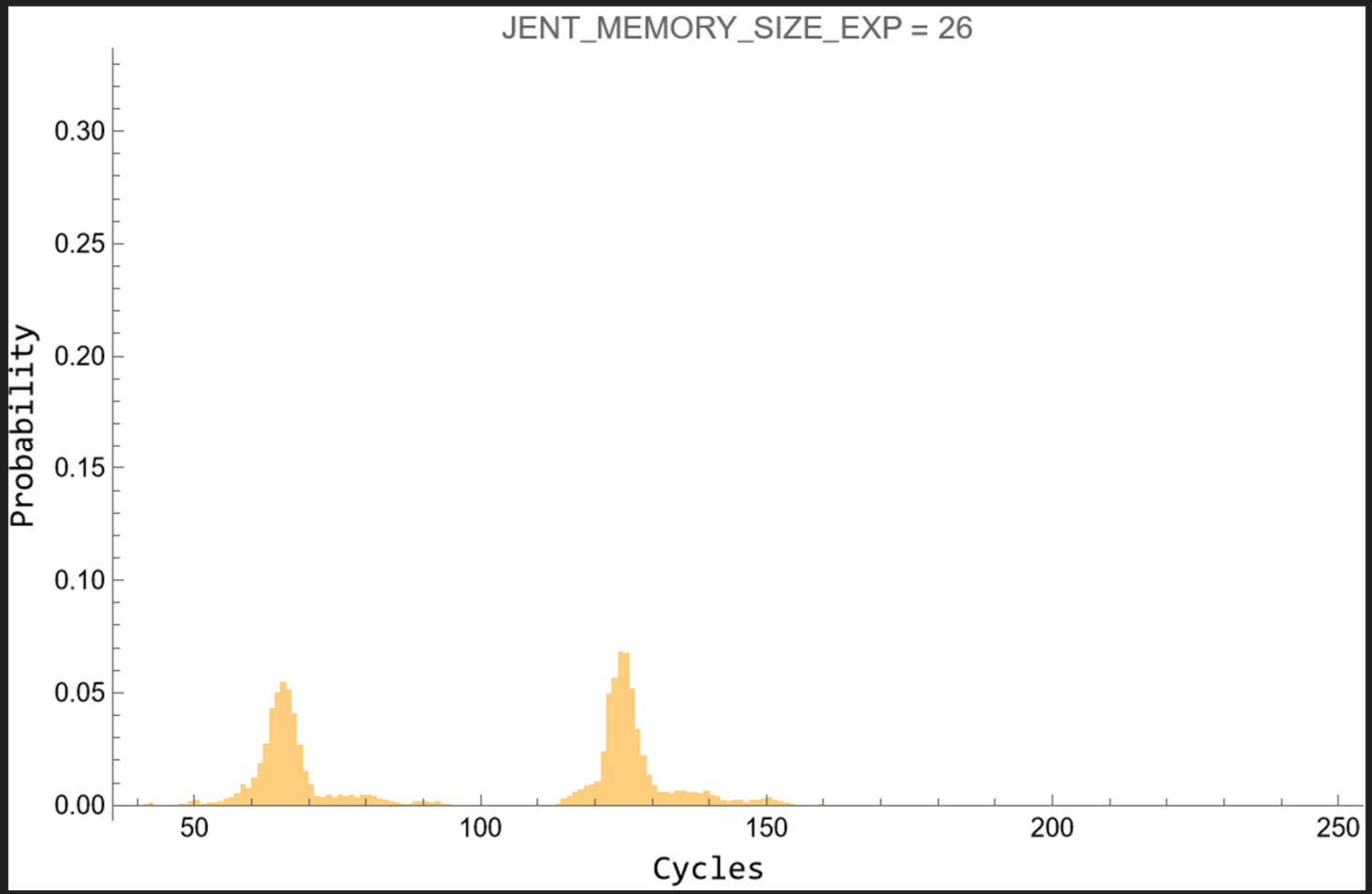

JENT_MEMORY_SIZE_EXP = 21
Probability
0.30
0.25
0.20
0.15
0.10
0.05
0.00
50
100
150
200
250
Cycles

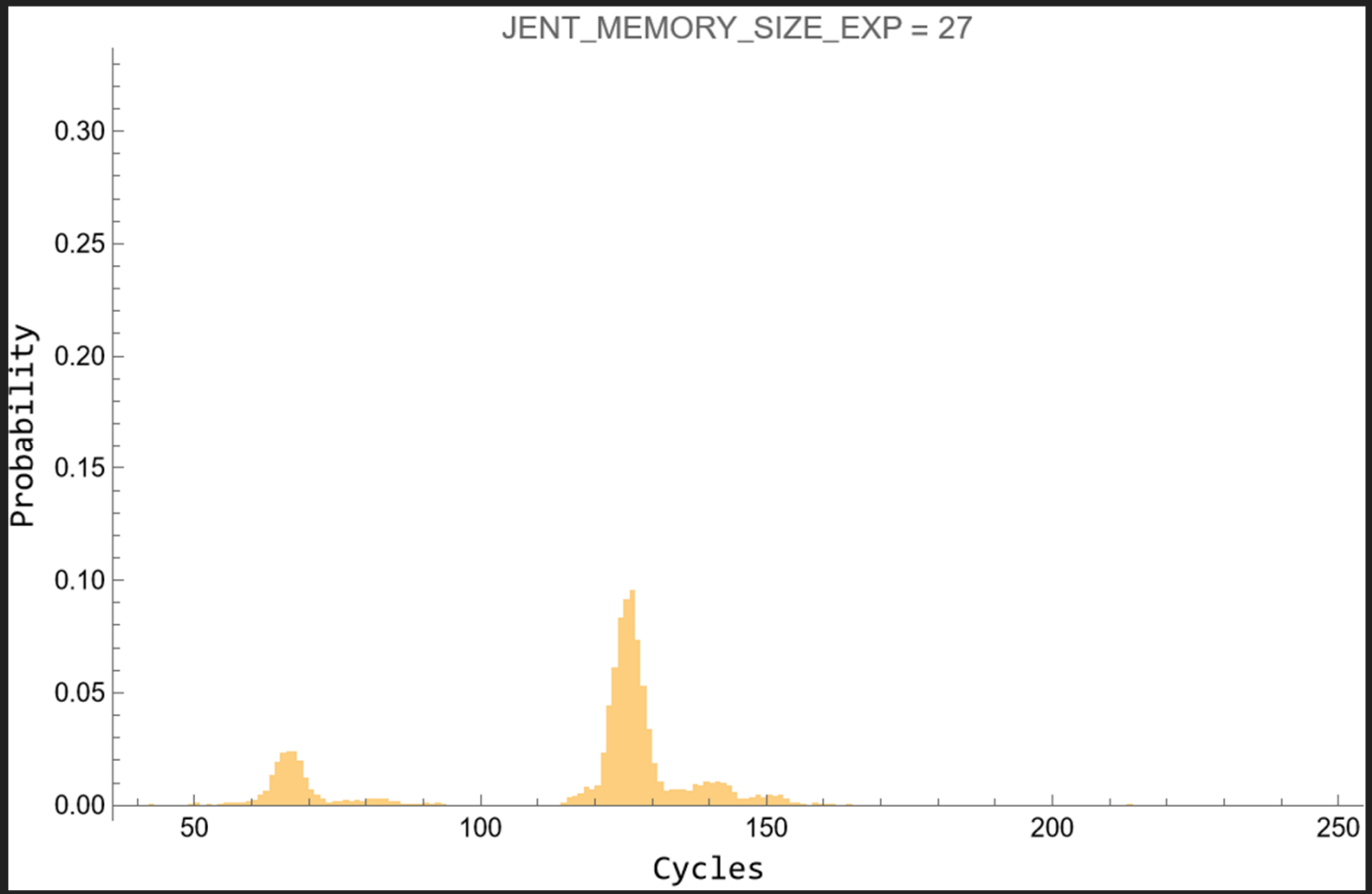

JENT_MEMORY_SIZE_EXP = 22
Probability
0.30
0.25
0.20
0.15
0.10
0.05
0.00
50
100
150
200
250
Cycles

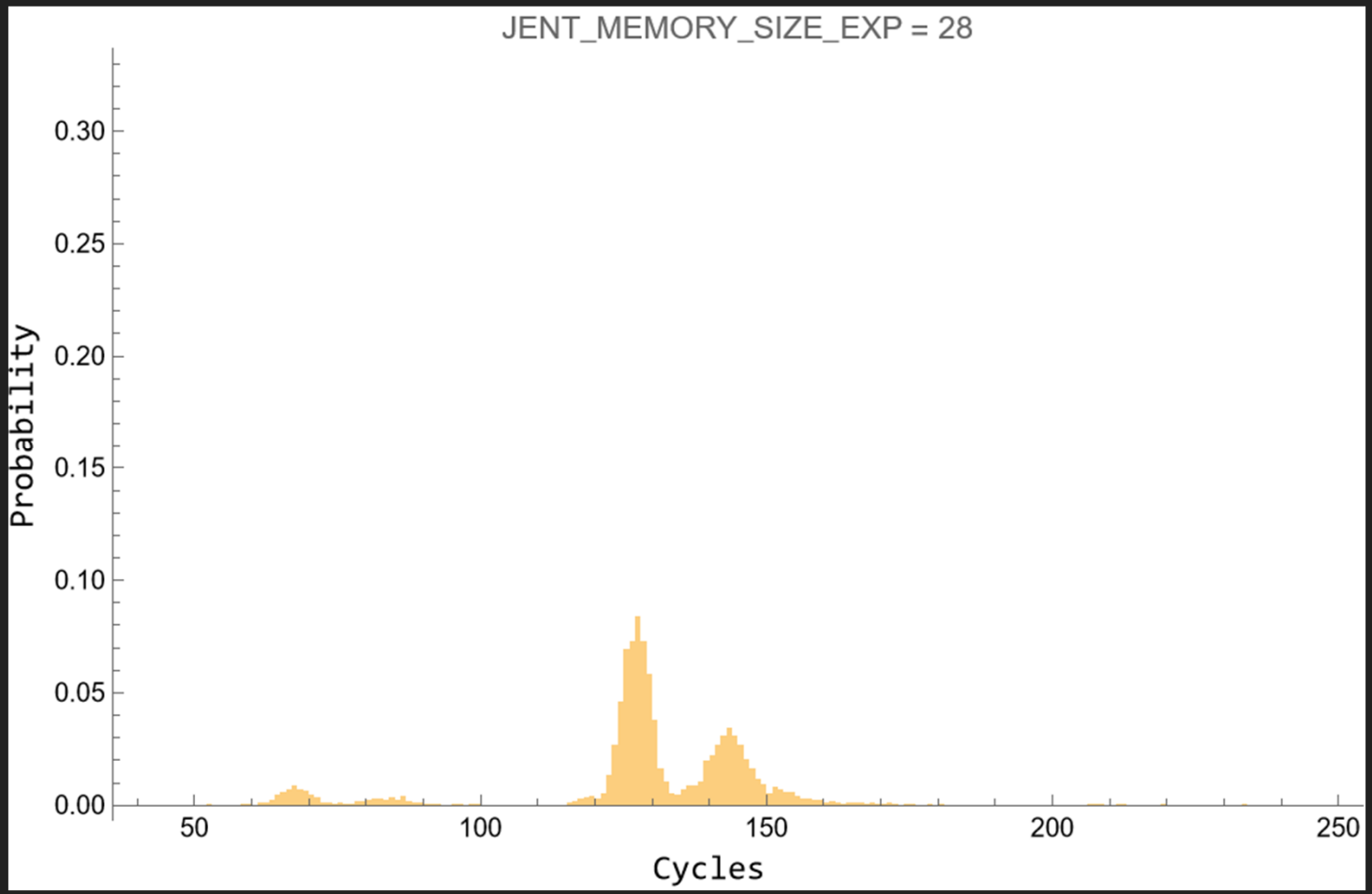

JENT_MEMORY_SIZE_EXP = 23
Probability
0.30
0.25
0.20
0.15
0.10
0.05
0.00
50
100
150
200
250
Cycles

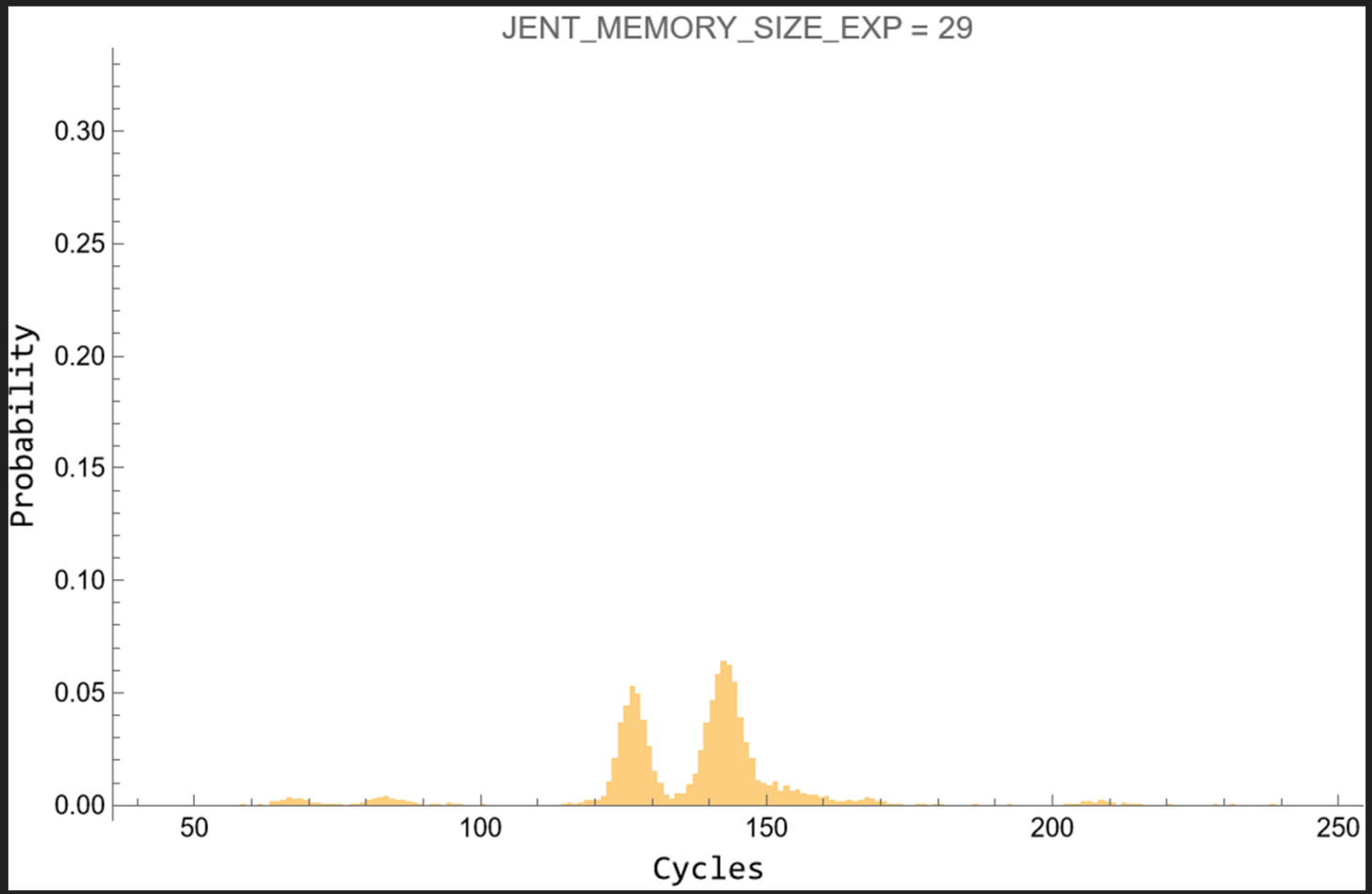

JENT_MEMORY_SIZE_EXP = 24
Probability
0.30
0.25
0.20
0.15
0.10
0.05
0.00
50
100
150
200
250
Cycles

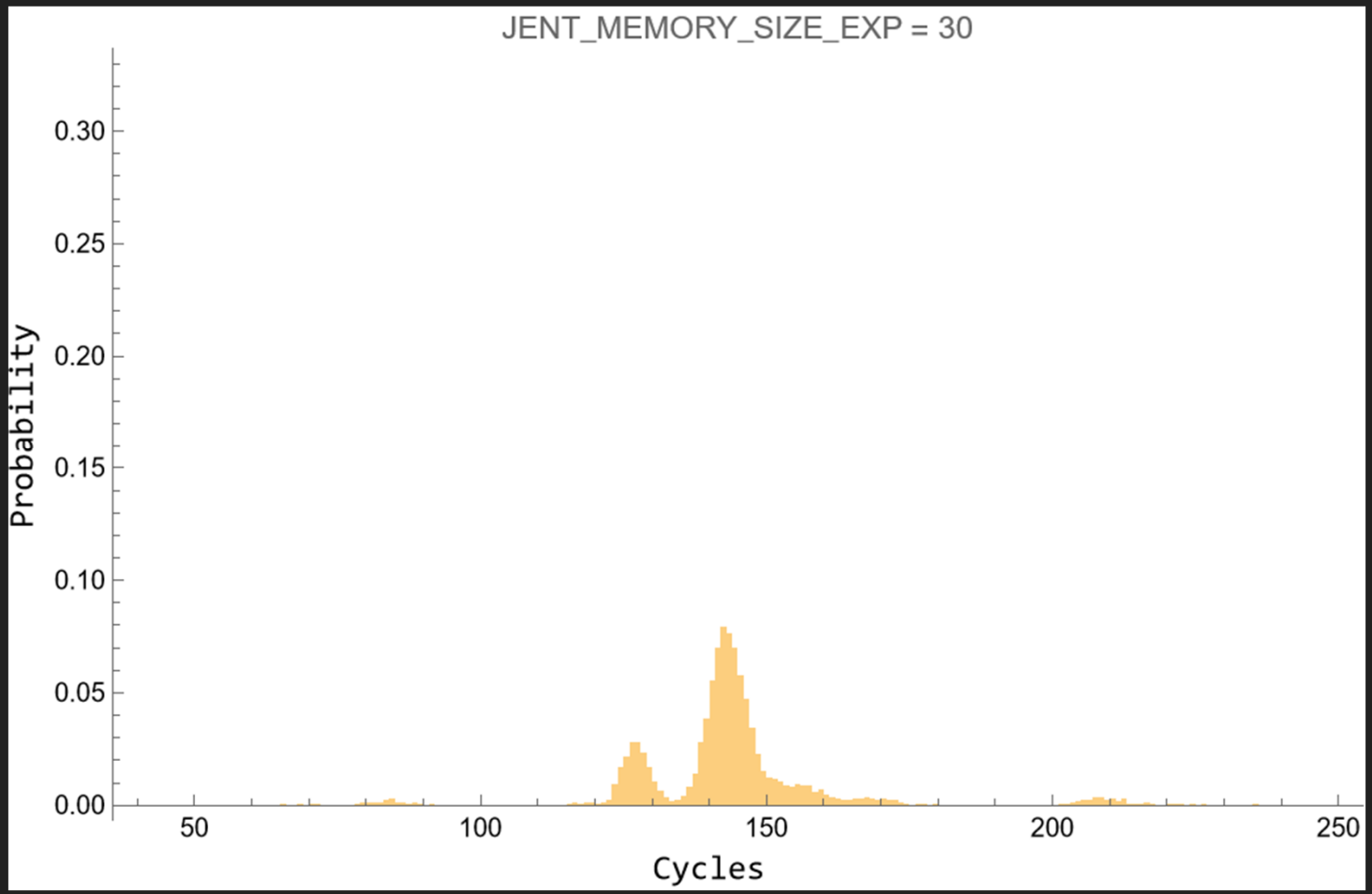

JENT_MEMORY_SIZE_EXP = 25
Probability
0.30
0.25
0.20
0.15
0.10
0.05
0.00
50
100
150
200
250
Cycles

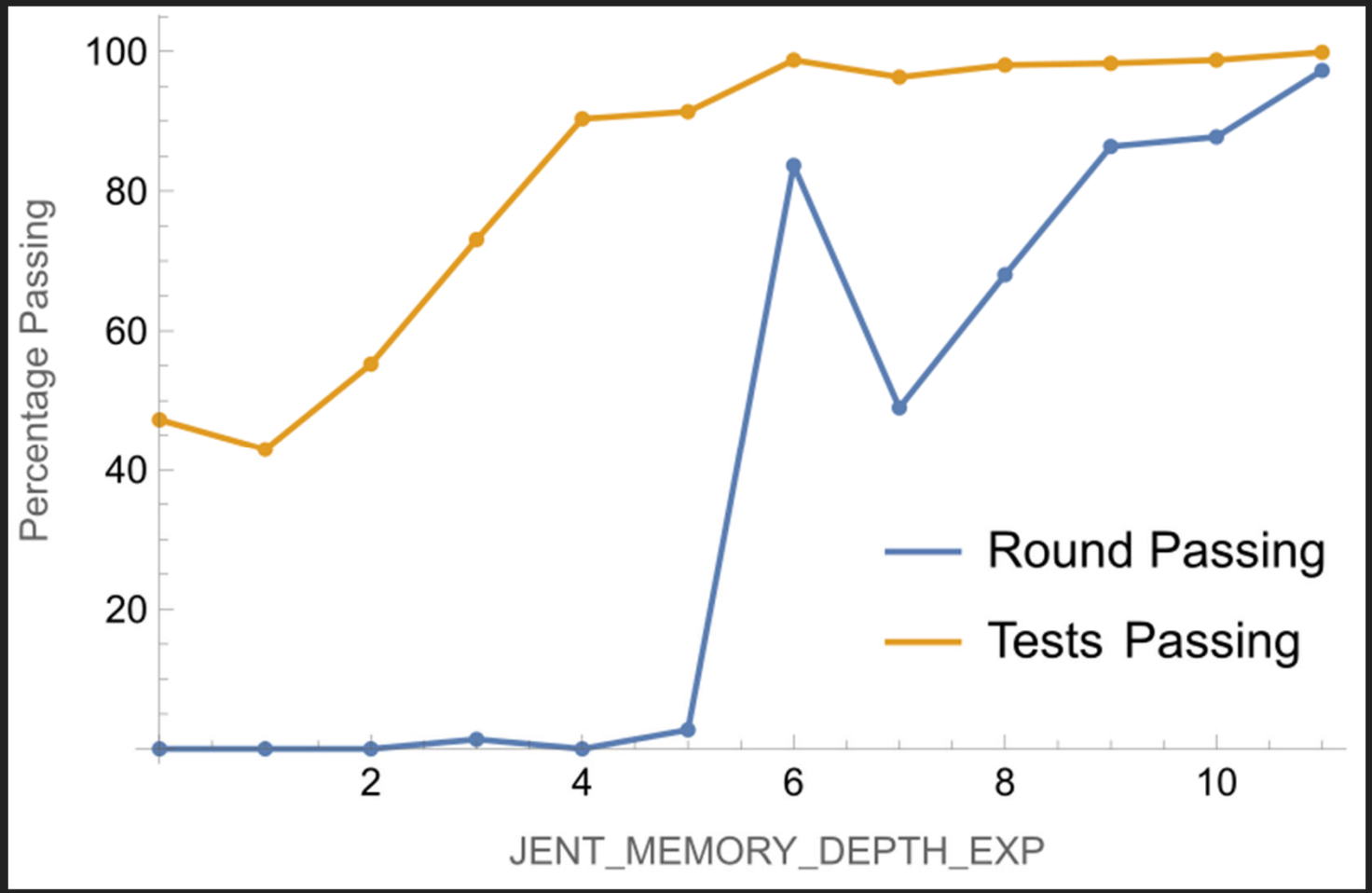

JENT_MEMORY_SIZE_EXP = 26
Probability
0.30
0.25
0.20
0.15
0.10
0.05
0.00
50
100
150
200
250
Cycles

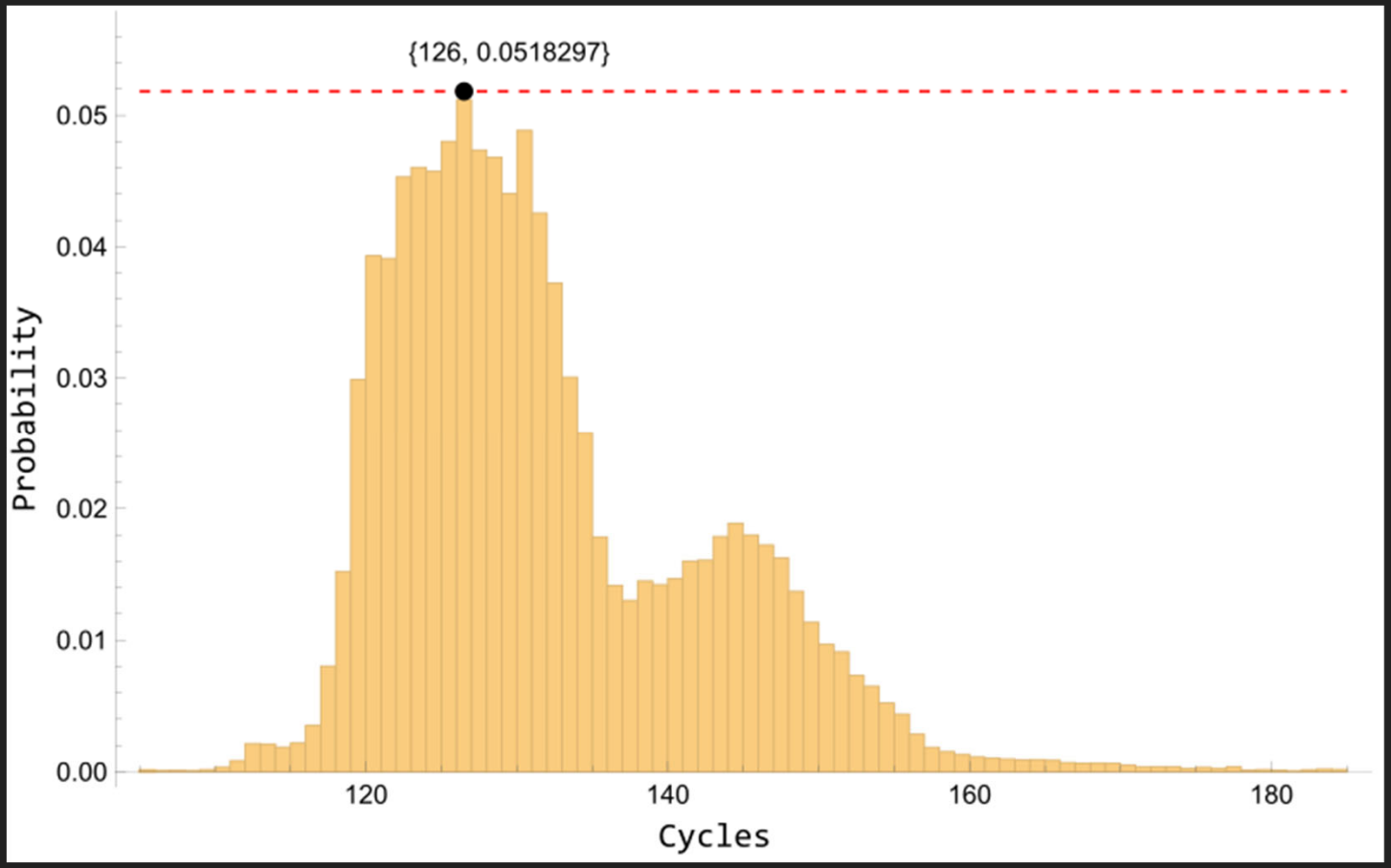

JENT_MEMORY_SIZE_EXP = 27
Probability
0.30
0.25
0.20
0.15
0.10
0.05
0.00
50
100
150
200
250
Cycles


JENT_MEMORY_SIZE_EXP = 28
Probability
0.30
0.25
0.20
0.15
0.10
0.05
0.00
50
100
150
200
250
Cycles


JENT_MEMORY_SIZE_EXP = 29
Probability
0.30
0.25
0.20
0.15
0.10
0.05
0.00
50
100
150
200
250
Cycles


JENT_MEMORY_SIZE_EXP = 30
Probability
0.30
0.25
0.20
0.15
0.10
0.05
0.00
50
100
150
200
250
Cycles


100
80
60
40
20
Percentage Passing
2
4
6
8
10
JENT_MEMORY_DEPTH_EXP
Round Passing
Tests Passing



H_submitter = - log_2 (0.0518297) = 4.27

References

- [90B] Meltem Sönmez Turan, Elaine Barker, John Kelsey, Kerry A. McKay, Mary L. Baish and Mike Boyle. Recommendation for the Entropy Sources Used for Random Bit Generation. January 2018.
- [JEnt (MemOnly)] Jitterentropy library with MemOnly updates. https://github.com/joshuaehill/jitterentropy-library/tree/MemOnly
- [JEnt (Original)] Jitterentropy library. https://github.com/smuellerDD/jitterentropy-library
- Jitter RNG SP800-90B Entropy Analysis Tool. https://github.com/joshuaehill/jitterentropy-library/blob/MemOnly/tests/raw-entropy/README.md
- [Müller 2022] Stephan Müller, CPU Time Jitter Based Non-Physical True Random Number Generator, July 1, 2022.

Thank you!