NIST Special Publication 800-90B Comments

Comments on the January 2018 (final) version of SP800-90B

Revision history

Revision	Date	Author	Description of Change
1.0	02/21/2018	Joshua E. Hill, PhD Ben Jackson, PhD	Initial release.
1.1	06/25/2018	Joshua E. Hill, PhD	Minor editorial changes. Additional Restart test comment (Comment 3d). Refinement to comment on Output_Entropy (Comment 5c), removal of Comments 5b and 5e, and a new comment on an instance where this function is not necessary (Comment 5f). New graph showing the problem with XORing ring oscillators (Comment 7b). New graph depicting the non-IID assessment distribution for ideal multi-bit sources (Figure 6). New section comparing modeled and statistically assessed noise sources (Section 4). Added "References" section (Section 5).
1.2	07/18/2018	Joshua E. Hill, PhD	Modify comment #1 to reflect our internal process refinement. Comment #15 withdrawn. New comment #16 describes a typo in the compression test's G(z). New comment #17 describes an undefined behavior in the LZ78Y prediction estimate.

1 Introduction

We would first like to congratulate the authors on the publication of the final SP800-90B document. It provides an excellent framework for addressing a very difficult and important challenge within many different security evaluation schemes, and we are sure that it will be extremely valuable for years to come.

We have some comments on the final document; most of our comments are requests for clarification or minor corrections. The two notable exceptions are comments #3 (the Restart Sanity Check) and #8 (the requirement for entropy and noise source entropy assessment invariance across all expected conditions). We view these two issues as jeopardizing the success of the validation program outlined within SP800-90B, because

- the restart sanity check will erroneously fail certain types of correctly working noise sources at a much higher rate than intended (Comment #3), and
- entropy and noise sources are expected to meet requirements that preclude almost all commercially produced noise/entropy sources (Comment #8).

We also outline a series of results that demonstrate that most of the statistical tests specified work as we expected (with the exception of the Restart Sanity Check, as mentioned above), provide examples of the tests assessing data produced from a variety of simulated results, and provide modeled min-entropy results for comparison.

2 Comments

Our comments on the final SP800-90B requirements are:

1. In general, our existing assessment process uses much more data than is requested in SP800-90B; as is clear in the graphs that follow, data sets from a noise source with fixed entropy-relevant parameters have min entropy assessments that conform to some underlying (noise-source dependent) distribution. A single value taken from that distribution doesn't tell the tester a great deal about the underlying distribution, but iterated assessment can. Our current practice is to request 100,000,000 samples, and break this data into 100 1,000,000-sample sets. We independently assess each of the 100 sets, and calculate the median of the assessed values. We then use bootstrapping to establish a confidence interval for the median, and use the lower bootstrap confidence interval bound as the assessed min entropy for the noise source. This practice yields a more consistent and repeatable value than simply using the result of a single assessment. We occasionally encounter products that produce data at such a slow rate that this process isn't feasible, at which point we can easily perform reduced testing (such assessments are still useful, but less meaningful). We have not encountered any vendor who was unable to produce at least several sets of 1,000,000 noise samples. We encourage you to refine this document so that such an assessment strategy is explicitly allowed and encouraged.
2. Section 3.1.3: In the non-IID case, the use of the single-bit-assessment strategy within the multi-bit-assessment strategy (using the term $n \times H_{\text{bitstring}}$ in the last paragraph of this section) limits H_{I} to about 85% of n, as a consequence of the fact that this is the median assessment for statistically idealized single-bit sources. The corresponding limitation for IID sources is 99% of n, but we rarely encounter noise sources that are IID. Further, it isn't clear that a binary IID assessment of a sample from an IID multi-bit sample is appropriate, as IID multi-bit samples need not be bitwise IID. (This also occurs in Section 3.1.5.2. See comment #6.)
3. For the test in Section 3.1.4.3 (the restart sanity check), there is a test construction issue. If this test indicates a failure, then the lab/vendor is prohibited from crediting the noise source with any entropy production, which is (from the vendor's perspective) a catastrophic result. As such, it is vital that this test behave correctly. Our testing indicates that this test fails much more commonly than anticipated (e.g., a theoretical failure rate of nearly 100% for wide data; see Section 3.2 of this comment document for details). The current test construction will lead to a significant proportion of correctly operating sources being erroneously disqualified.
    a. This test isn't correctly constructed, because the value that is found isn't necessarily the maximum count of the noise source's most likely symbol. Instead, it is the maximum count of the column/row-specific most likely symbol, which may be the source's most likely symbol, or it might be any of the other symbols. For example, in the binary case, if one wants to find $P(X = X_{\text{max}})$, then one should account for both the case where the noise source's most common symbol occurred the most in the row/column, and also account for the probability that the most common symbol in a row/column is the noise source's least common symbol. The underlying distribution for the existing test is really the maximum count of any symbol of a multinomial distribution; our testing indicates that the binomial distribution isn't a good approximation of the actual underlying distribution. This suggests three possible approaches:
        i. Corrected Simulated Cutoff Restart Sanity Check: The tester establishes the appropriate cutoff through simulation, using the parameters for the noise source under evaluation. In our testing, the highest cutoff (the "worst case") appears to occur when as many symbols as possible have the same probability as the most probable symbol, and then (if necessary) one final symbol so that the sum of the probabilities is 1 (all

other symbols have probability 0). We found that performing 2,000,000 rounds of simulation of the 1000-sample test (analogous to the per-row/column test) provided stable results.

ii. Corrected Exact Cutoff Restart Sanity Check: The tester establishes the appropriate cutoff through application of exact methods; Levin's "A Representation for Multinomial Cumulative Distribution Functions" and Corrado's "The exact distribution of the maximum, minimum, and the range of Multinomial / Dirichlet and Multivariate Hypergeometric frequencies" both contain (somewhat complicated) procedures for extracting exact cutoffs. Again, one would have to apply the "worst case" probabilities described above.

iii. Corrected Binomial Restart Sanity Check: Change the test so that the binomial distribution is the correct underlying distribution. One way to do this would be to first find the most common symbol within the dataset, and then count the number of occurrences of that particular fixed symbol in each row/column.

We characterize the original approach, and modified approaches (i) and (iii) in Section 3.2 of this comment document. Approach (ii) is expected to perform equivalently to approach (i).

b. This test is also not correctly constructed, because the rows/columns are evidently not independent (any matrix entry which is the most likely symbol contributes to a row and a column count). It's not clear how to fix this issue, but it appears that this doesn't significantly impact the pass rate, so we think that it seems safe to ignore this theoretical problem.

c. This current test specification has some notational problems (this does not affect the results of the testing). The stated equation for the p-value,

$$P(X \geq X_{\text{max}}) = \sum_{j=X_{\text{max}}}^{1000} \binom{1000}{j} p^j (1-p)^{1000-j}$$

is incorrect, as this is not the probability of this event. This fact is clearly acknowledged in the second paragraph of Section 3.1.4.3 by the use of the test statistic cutoff of 0.000005 for a targeted false reject rate of 0.01. The correct calculation (which should then be compared against the cutoff 0.01) can be put in terms of the appropriate Binomial Distribution CDF (BCDF) as follows[1]:

$$P(X \geq X_{\text{max}}) = 1 - \left[1 - \sum_{j=X_{\text{max}}}^{1000} \binom{1000}{j} p^j (1-p)^{1000-j}\right]^{2000}$$
$$= 1 - \left(BCDF(1000, p, X_{max} - 1)\right)^{2000}$$

d. For wide data (data with more than 256 symbols), it is not clear if the data tested for the restart tests is the mapped-down data (as required in Section 3.1.3) or the original wide data. The text of 3.1.4.1 seems to suggest that this should be unmapped data, but in 3.1.4.2, the entropy estimation tests are conducted, which generally presume that the data is at most 8 bits wide.

4. In Section 3.1.5, the entropy source is conceptualized as having a single conditioning function, but it isn't clear how entropy sources that process the raw noise through more than one conditioning stage should be handled. Should all conditioning stages be thought of as a single conditioning function, or is it acceptable to have multiple

[1] This presumes that the test has been reformulated so that the underlying distribution is the Binomial Distribution. If one of the other approaches is taken, then the p-value isn't likely to be easy to calculate, and a comparison with a pre-calculated cutoff is performed instead.

conditioning stages, where separate entropy assessments are generated for each step? In the latter case, must the output of each non-vetted conditioning function be separately statistically assessed (generating separate h' for each stage)?

5. In Section 3.1.5.1.2:
    a. The upper bound for the number of collisions used by Output_Entropy function (U) only applies when $n_{\text{in}} \geq n = \min(n_{\text{out}}, nw)$ (indeed, in the paper that this formula is based on[2], this formula applies only in the case where $n + \log_2 n \ll n_{\text{in}}$). In the case where $n_{\text{in}} < nw$, we think that the text should indicate that nw should be set to n_{in} so that the formula for U makes sense. This action is consistent with the notion of nw presented in Appendix E.
    b. Comment withdrawn. (Modeling suggests the proposal in this comment is overly conservative.)
    c. It doesn't seem proper that the behavior of Output_Entropy varies with the data encoding of output of their noise source; in this formula, we think that n_{in} should likely be replaced by the minimal number of bits required to encode the noise source output being passed into the conditioning function ($\lceil w \log_2 k \rceil$). Below we have the result (after applying the changes suggested in (a) above) of using something like CRC64 (so $n_{\text{out}} = nw = 64$), under the assumption that the vendor feeds in noise source outputs which are one of two symbols, with 1 bit of entropy each noise source output / conditioning input block, either encoded in 1-bit input blocks, or in 64-bit input blocks. Fundamentally, it seems like the entropy produced should be the same in either case. This issue would naturally arise when using any conditioning function that can operate on blocks of arbitrary length.

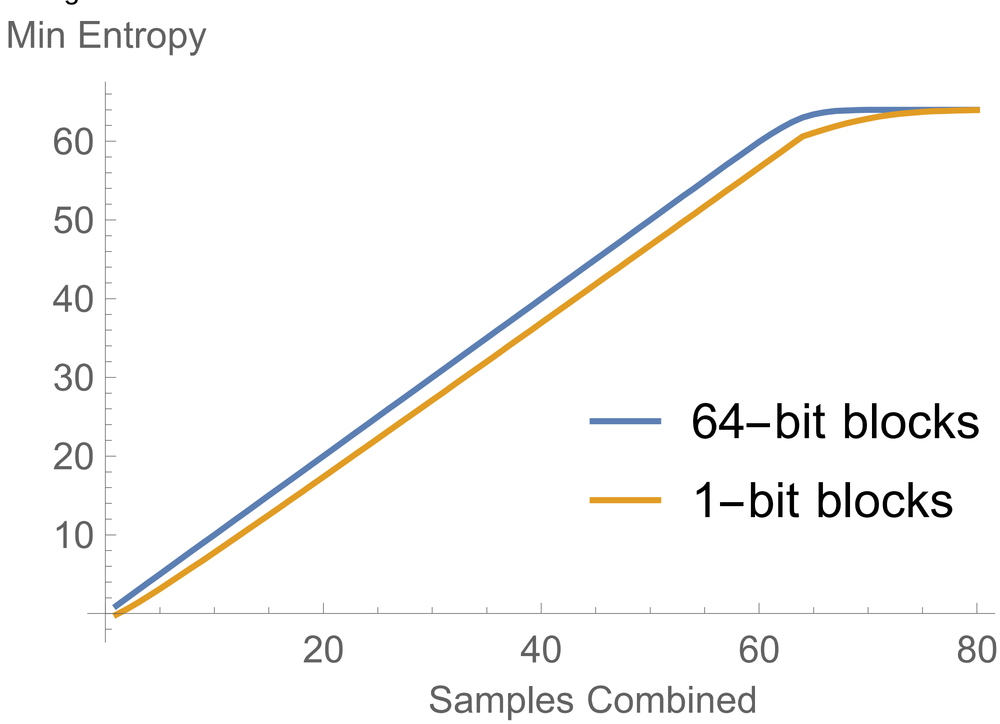

Figure 1

[2] It is also not clear what the meaning of α is in this paper, so it's not clear that the selection of $\alpha = 1$ is appropriate.

d. In this section, SP800-90B claims that “vetted conditioning functions are permitted to claim full entropy”, but it isn’t clear how this claim could be justified; the formula (either before or after the changes we propose) doesn’t appear to yield exactly $h_{\text{out}} = n_{\text{out}}$, and it’s not clear how close to $h_{\text{out}} = n_{\text{out}}$ you have to be in order to describe the entropy source as producing “full entropy”. (This could be resolved in SP800-90C, but this ambiguity immediately impacts the use of the SP800-90A CTR_DRBG without a derivation function, as this construction requires the seed to be full entropy.)

e. Comment withdrawn. (This question is resolved in the last paragraph of 3.1.5.1.2.)

f. In the instance where the conditioning function can be shown to be bijective, there should be some allowance to not apply this formula. (In this instance, $h_{\text{out}} = h_{\text{in}}$) Common examples of such processing include encrypting the raw outputs, and certain styles of LFSR use.

6. In the non-IID case, Section 3.1.5.2 effectively limits h_{out} to about 85% of n_{out}, as a consequence of the fact that this is the median assessment for statistically idealized single-bit sources. (This applies even in the multi-bit base, because one step is to assess the multi-bit symbols as if they were the output of a bit-oriented noise source.) The corresponding limitation for IID sources is 99% of n_{out}, but we rarely encounter noise sources that are IID. Further, it isn’t clear that a binary IID assessment of a sample from an IID multi-bit sample is appropriate, as IID multi-bit samples need not be bitwise IID. (This also occurs in Section 3.1.3; see Comment #2 above.)
7. In Section 3.1.6:
    a. It’s not clear what “multiple copies of the same physical noise source” are, exactly. For example, can we treat multiple ring oscillators with different nominal frequencies as such “multiple copies”? Specifically, how can vendors and labs distinguish between a “copy” of a noise source and an “additional noise source”?
    b. We are uncomfortable with the standard allowing the XOR of “multiple copies” of the “same” physical noise source as being considered a single noise source. In particular, in the provided example of the XOR of the output of multiple ring oscillators, if there are a large enough number of rings, this output is expected to look statistically excellent even if the rings are fully deterministic (see Figure 2).This is a particular problem in this context, as the main assessment strategy here relies on just such a statistical assessment to establish the entropy.

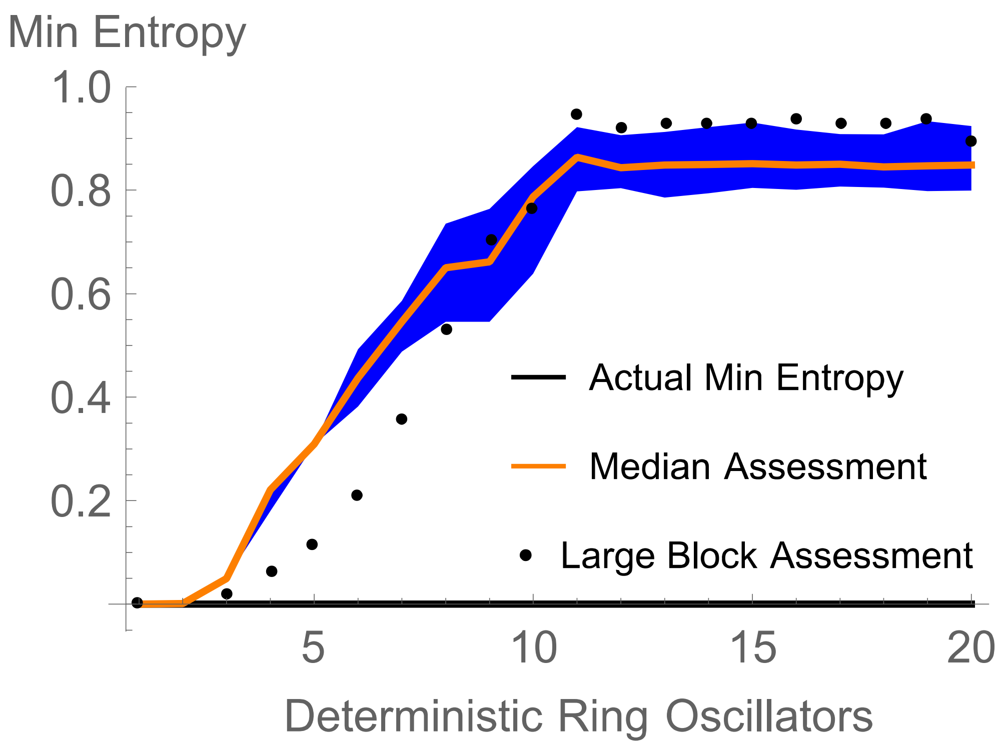

Figure 2

c. The document states that an entropy source can only credit data from a single noise source (the primary noise source). All other noise sources cannot be credited, and they can only be used (at all, without credit) if the conditioning function combining the noise source outputs is one of the vetted conditioning functions. This impacts operating systems particularly, as this section suggests that, for example, network interrupt timing and hard drive timing cannot both be credited. This means that OS-based entropy sources will have to designate a single primary noise source to credit, and can only continue using the other sources if the conditioning is performed with a vetted conditioning function. (For example, this isn't compatible with how Linux's /dev/random LRNG is structured.) We don't have a technical objection to this requirement (it is hard to characterize mutual entropy in such systems!), but it's going to cause substantial headaches for our customers.

8. Both Section 3.2.1 requirement #3 and Section 3.2.2 requirement #2 seem to suggest that all instances of the noise source must behave essentially the same way across all per-part and environmental conditions within its operational range. This isn't true for any noise / entropy source that we've ever encountered; most of the physical sources have substantial part-to-part variation due to manufacturing variations, substantial temperature and voltage sensitivity, and some depend on the frequency of external clocks (e.g., to establish the sampling frequency). Most non-physical noise sources are dependent on the computer's workload, etc. As such, the behavior of almost all noise sources is dependent on some set of entropy-relevant parameters. We suggest that these requirements be changed to require that the vendor produces a list of all such entropy-relevant parameters, require stable behavior of the entropy/noise source for fixed entropy-relevant parameters, and then separately require assessment across the expected range of entropy-relevant parameters (e.g., across a temperature / voltage / process characteristics envelope). The final assessed min entropy value would then be

the smallest assessed value for any entropy-relevant parameter tested. In the absence of such a requirements change, almost no commercially produced noise / entropy sources would be capable of passing these requirements.

9. For Section 4.4.1 (the Repetition Count Test), there is no upper bound for C, which renders this health test ineffective at obtaining any particular security benefit. With the current requirements set, vendors can always claim that $\alpha = 0$ (or arbitrarily close to this value), and then vacuously claim to have this test in place. We recommend that you apply the equivalent requirement imposed by Section 4.5 to this test; requirement (a) from Section 4.5 would impose an upper bound of $C \leq \left\lceil \frac{100}{H} \right\rceil$. The following should be added to accomplish this: "α **shall** be chosen so that $C \leq \left\lceil \frac{100}{H} \right\rceil$."
10. For Section 4.4.2 (the Adaptive Proportion Test):
    a. The description of the cutoff value isn't precise. It should say "Mathematically, C **is the smallest integer that** satisfies the following equation", where the new text is bolded.
    b. Using Excel functions as the sole descriptor of how parameters are calculated seems inappropriate (though, we have no objection to including these for reference). Please describe the CRITBINOM function as the compositional inverse of the CDF for the relevant binomial distribution.

       The cutoff calculation isn't correct (though it is close). By the construction of the test, the first symbol has already been produced, and thus must necessarily have been observed (i.e., there is no possibility of zero of these symbols being observed). The count of the number of these symbols can then be bounded using the binomial distribution, with $W - 1$ (not W) trials. Thus, the formula in footnote 10 should be $C = 2 +$ CRITBINOM($W - 1$,power$\left(2, (-H)\right), 1 - \alpha$). This has a series of small effects on Table 2, which should be as below (updated values are bolded).

Table 1

Binary Data W=1024		Non-Binary Data W=512	
Entropy	Cutoff Value C	Entropy	Cutoff Value C
0.2	941	0.5	**411**
0.4	**841**	1	311
0.6	748	2	**178**
0.8	664	4	**63**
1.0	**590**	8	**14**

    c. There is no upper bound for C, which renders these test requirements ineffective at obtaining any particular security benefit. With the current requirements set, vendors can always claim that $\alpha = 0$ (or arbitrarily close to this value), and then vacuously claim to have this test in place. We recommend that you apply the equivalent requirement imposed by Section 4.5 to this test. Requirement (b) from Section 4.5 would impose an upper bound for C for each entropy value. The following text should be added to accomplish this: "C **shall** be chosen so that, if the entropy source degrades so that it produces only half of the expected entropy, the probability of false accept for this test is less than 50% after examining 50,000 consecutive samples."

Such a bound for C can be calculated as follows: there are $T = \left\lfloor \frac{50000}{W} \right\rfloor$ trials, and we need the eventual probability of non-detection to be less than 50%. Thus, if we call the probability of a single test not finding a failure under these conditions p_{nd}, then we have the cutoff value $p_{\text{nd}}{}^T < 2^{-1}$, so we need

$$p_{\text{nd}} < 2^{-1/T}.$$

The relevant calculation for this maximum cutoff is then based on the per-trial probability of not detecting this low entropy condition, in terms of the probabilities of the k distinct symbols in the degraded noise source, p_i. We denote the family of per-symbol binomial probabilities in terms of the binomial CDF function (BCDF) as $c_i = \text{BCDF}(W-1, p_i, C-2)$, whence

$$p_{\text{nd}} = \sum_{1 \le i \le k} p_i c_i.$$

Let A be an index of a most likely symbol. We can produce trivial bounds for p_{nd} by noting that $c_i \le 1$, so

$$p_{\text{nd}} \le p_A c_A + \sum_{\substack{1 \le i \le k \\ i \ne A}} p_i = 1 - p_A(1 - c_A).$$

We seek a C so that $p_{\text{nd}} \le 1 - p_A(1 - c_A) < 2^{-1/T}$, thus satisfying requirement (b) from Section 4.5. Simplifying, we are left with the inequality

$$c_A < 1 - \frac{1 - 2^{-1/T}}{p_A}.$$

This inequality is satisfied when

$$C \le \text{CRITBINOM}\left(W - 1, 2^{-\frac{H}{2}}, 1 - \frac{1 - 2^{-1/T}}{2^{-\frac{H}{2}}} \right) + 1.$$

A corresponding table similar to SP800-90B's Table 2 would then be as follows:

Table 2

Binary Data W=1024		Non-Binary Data W=512	
Entropy	Max Cutoff Value C	Entropy	Max Cutoff Value C
0.2	972	0.5	450
0.4	914	1	386
0.6	858	2	281
0.8	804	4	148
1.0	754	8	40

11. For Section 5.2.1
    a. In step #1, $e_{i,j}$ should instead be $e_{i,j} = p_i\, p_j \left\lfloor \frac{L}{2} \right\rfloor$. (The existing statement neglects the floor operation).
    b. In step #2, the procedure isn't completely specified. The document should indicate how symbols with equal $e_{i,j}$ should be sorted so that the estimator is fully specified (otherwise, a range of outputs is possible, depending on how symbols with equal expected values are sorted). One possibility (which is what we

implemented in our tool) is to sort the symbols primarily on the expected value, and secondarily (lexicographically) sort on the tuple value.

12. For Section 5.2.2, step 2, this again isn't fully specified for the same reason as in comment #11b. (What sort is correct when the expected values are equal?)
13. For Section 6.3.3, it's regrettable that the multi-bit Markov estimator (which was present in the last draft) was removed. This estimator seemed to provide meaningful insight to a variety of systems and was reasonably well behaved so long as adequate data was provided.
14. For Section 6.3.5 (the t-tuple estimate), what if there is no such t? The estimator is inconclusive in this instance, and the estimator specification should indicate what to do when this condition occurs.
15. Comment withdrawn. In the MultiMMC Prediction Estimate, maxEntries is a per-length bound on the number of counters, not a global value across all word lengths (as is the case with the LZ78Y Prediction Estimate). In the MultiMMC Prediction Estimate, if a new postfix comes after a known prefix, the corresponding counter is not created when the number of counters is already maxEntries, whereas in the LZ78Y Prediction Estimate, encountering a known prefix always results in incrementing some value in the dictionary for the observed postfix (even after maxDictionarySize prefixes are encountered). These distinguish the two prediction estimates.
16. For section 6.3.4, there is a typo in the function definition of $G(z)$. The Hagerty-Draper paper's description [HD] of this sum (equation 4.35) makes it clear that the sum should be taken over all the symbols in the test group (that is, the symbols after those used to build the dictionary); the outer sum should have the same number of terms as the testing group, $\lfloor L/b \rfloor - d$. As such, the upper bound for this sum should be $\lfloor L/b \rfloor$.
17. For section 6.3.10
    a. Algorithm step 3.a.i.2, when initializing the prefix $\left(s_{i-j-1}, \dots, s_{i-2}\right)$, one should initialize all postfix values to 0, not just the current postfix (s_{i-1}), as if the same prefix reoccurs at a later index, say at $\left(s_{i'-j-1}, \dots, s_{i'-2}\right) = \left(s_{i-j-1}, \dots, s_{i-2}\right)$, the test for this prefix will succeed (in step 3.a.i/3.a.ii), and the corresponding postfix entries will be incremented in step 3.a.ii, without $D\left[s_{i'-j-1}, \dots, s_{i'-2}\right]\left[s_{i'-1}\right]$ necessarily having been initialized.
    b. Algorithm step 4 uses C, but C is not defined earlier in the algorithm. Between steps 3c and 4, insert, "Let C be the number of ones in the array 'correct.'"

3 Results with Uniform Data

Here is a summary of the results of the testing that we've performed within in our acceptance testing for the statistical tests described in the SP800-90B-final document.

For all these tests, we use data output from the Intel RDRAND instruction, which uses an AES-128 CTR_DRBG (so it should be fairly statistically ideal, and any problems we see are likely due to test construction issues).

3.1 Estimating Min Entropy

3.1.1 Non-IID Overall Assessments

The first set of graphs are histograms of the assessed entropy under the non-IID track for data of various bit lengths. Each of these assessments was done on data sets of 1,000,000 samples.

For the binary case, we performed 1,000,000 distinct assessments (each using a distinct data set). For all the other cases, we performed 30,000 distinct assessments per data width (each using a distinct data set). Recall, also, that the binary-case involves more estimators, so the binary results have a somewhat different meaning than the other results.

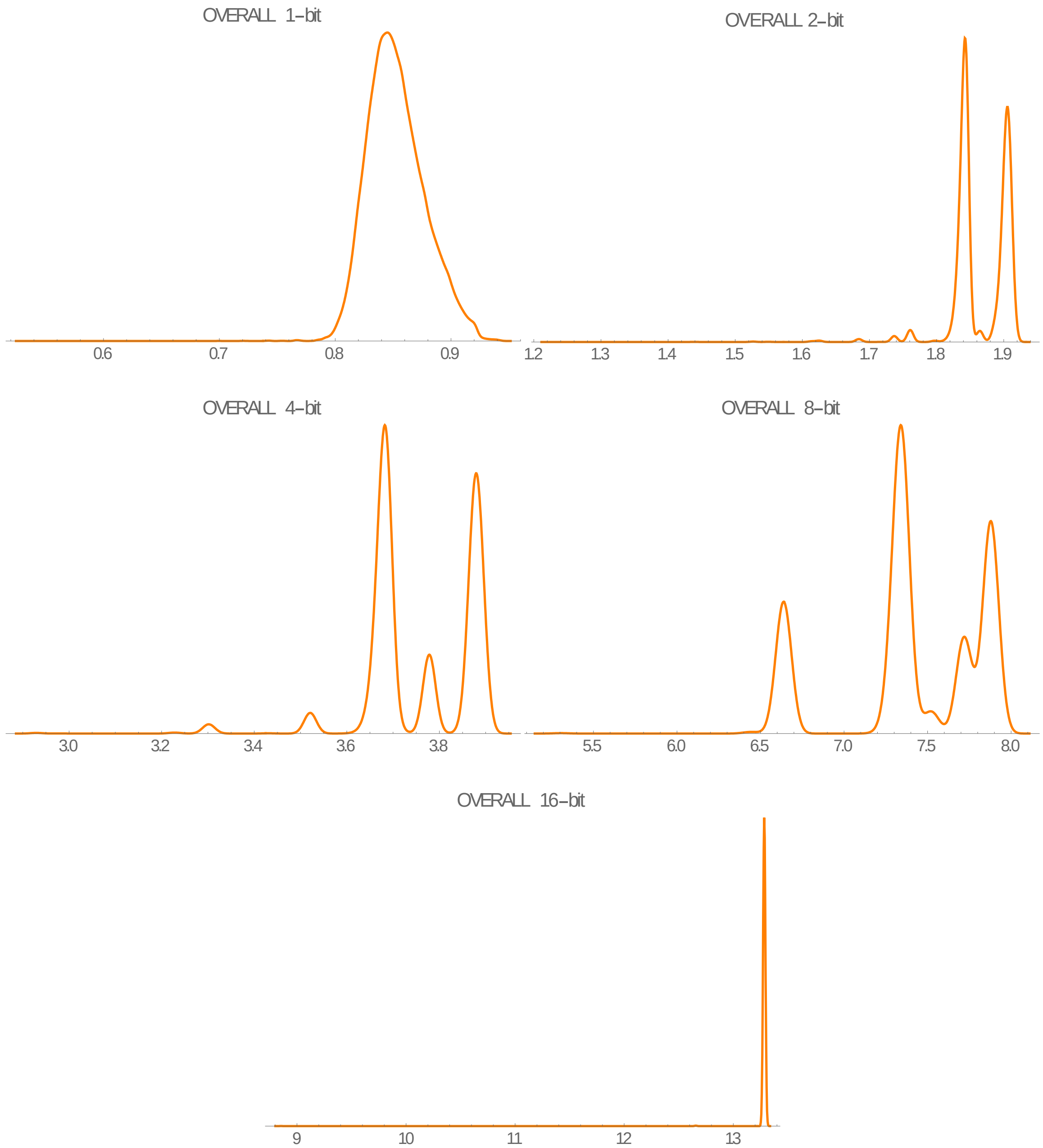

Figure 3

It would appear that this non-IID assessment process works reasonably well up to 8-bit symbols, and not very well for 16-bit symbols. (This last finding is clearer when reviewing the results of the individual estimators).

Figure 5 shows the median of the assessment divided by the bit length. (Recall that the non-IID assessment for 1-bit symbols involves more estimators, so the results are not strictly comparable with the other values.)

We can also supply per-estimator information if you desire, but the “top level” numbers seemed most interesting in this case.

3.1.2 IID Overall Min Entropy Assessment

The IID entropy assessment is limited to the result of the Most Common Value estimate.

For the binary case, we performed 1,000,000 distinct assessments (each using a distinct data set). For all the other cases, we performed 30,000 distinct assessments per data width (each using a distinct data set).

We provide the histograms for the IID assessment track below.

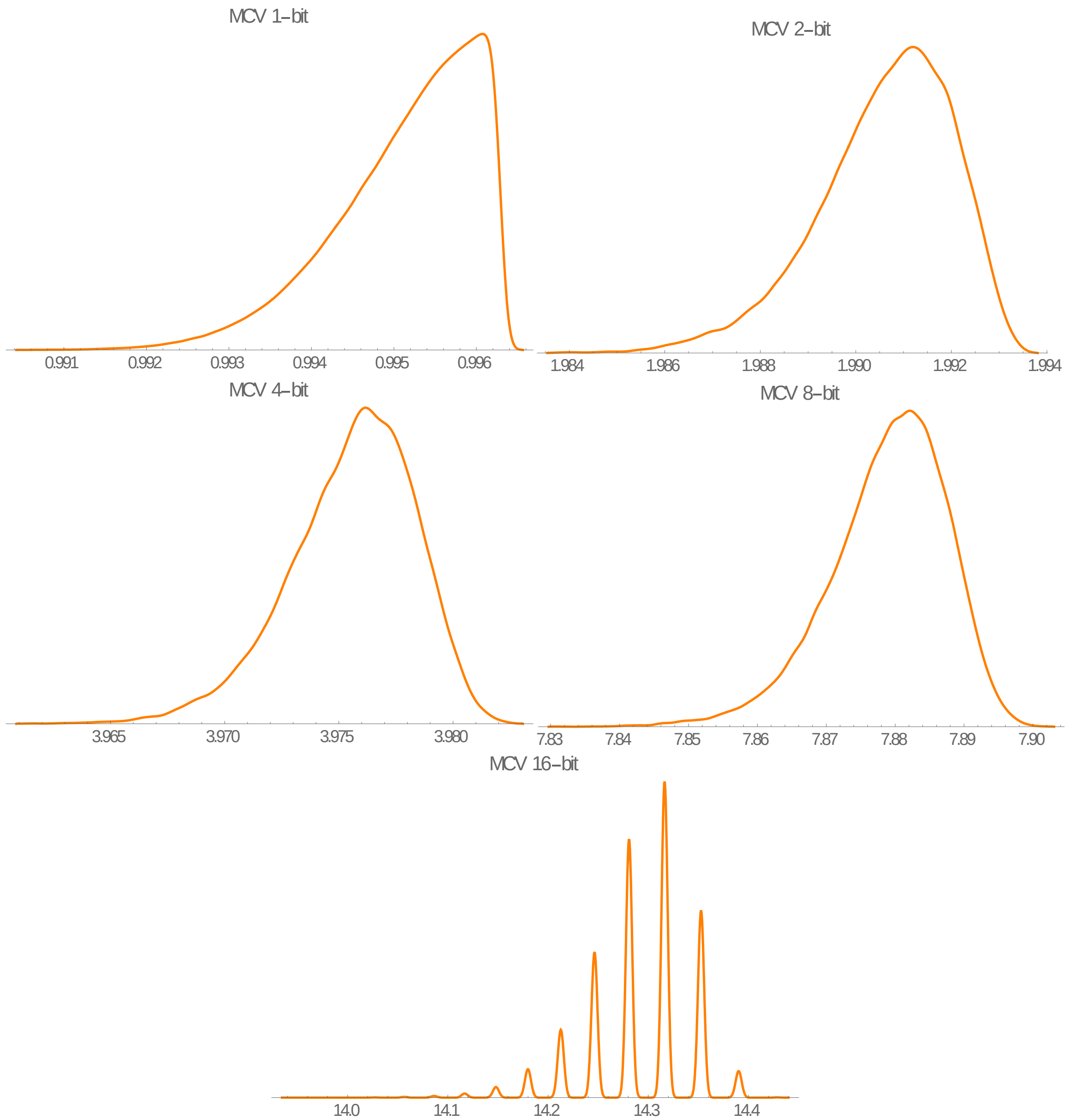

Figure 4

In general, the median of the distribution of assessments is reasonably close to full entropy in each assessment, other than in the very-wide symbol case, where inadequate data makes the estimator perform oddly. Figure 5 shows the median of the assessment divided by the bit length.

3.1.3 Interpretation of Min Entropy Estimates

The non-IID track involves more estimators than the IID track, and so can detect more types of defects in the data produced by the noise source. As a consequence of using more estimators, the overall assessment (which is the minimum of any particular estimator's assessment) is expected to decrease. As such, we expect there to be a reduction in the assessed entropy from the IID case to the non-IID case.

Further, increasing the number of symbols present in the data (k) while keeping the number of samples examined constant is expected to similarly decrease the assessment for most of the estimators.

As such, the data presented in Figure 5 is consistent with the behavior of any similar assessment process. Both assessment tracks seem to perform as expected.

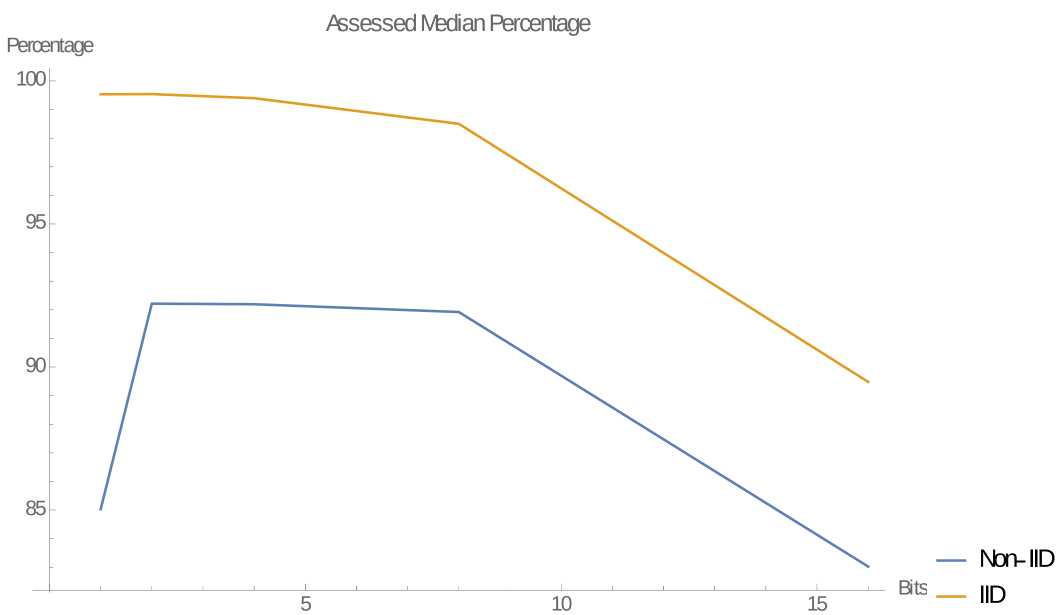

Figure 5. Assessment Percentage (IID and Non-IID)

An alternate presentation of the multi-bit IID assessments is in Figure 6; in Figure 6, the blue regions depict the observed range of assessments.

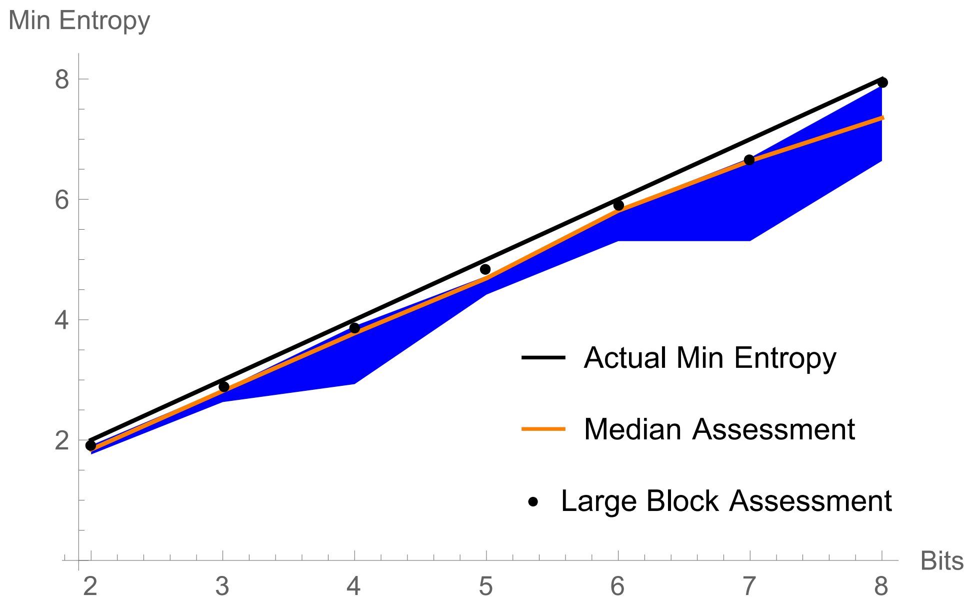

Figure 6

3.2 Restart Sanity Check

We conducted each test under one of three entropy hypotheses:

1. full entropy (the “FullEnt” series), which is the correct assumption for this data,
2. the median IID assessment (the “IIDMedianEnt” series), reflecting what we’d expect for a near-ideal source under the IID assessment strategy, or
3. the median non-IID assessment (the “MedianEnt” series), which is what we’d expect for a near-ideal source under the non-IID assessment strategy.

For each variant of the restart sanity check that we examined, we conducted 100,000 restart sanity checks per data width / entropy assumption tuple.

3.2.1 Original Restart Sanity Check

As mentioned above, the stated probability equality (which is described as equivalent to the calculation of the p-value) is invalid, so we don’t expect the distribution of the p-values calculated in this way to be uniformly distributed. If this test were completely reasonable, we would expect the corrected p-values[3] to be uniformly distributed in the interval [0,1], but this was not the result that we observed. None of these parameters produce the desired uniform distribution of p-values, which suggests that the underlying test construction is flawed.

Having said that, the proportion of failures is still reasonable for some conditions, as seen in the following graph.

[3] The p-values calculated using the formula provided in Section 3.1.4.3 can be corrected using the function $p_{\text{value}} = 1 - \left(1 - P(X \geq X_{\text{max}})\right)^{2000}$.

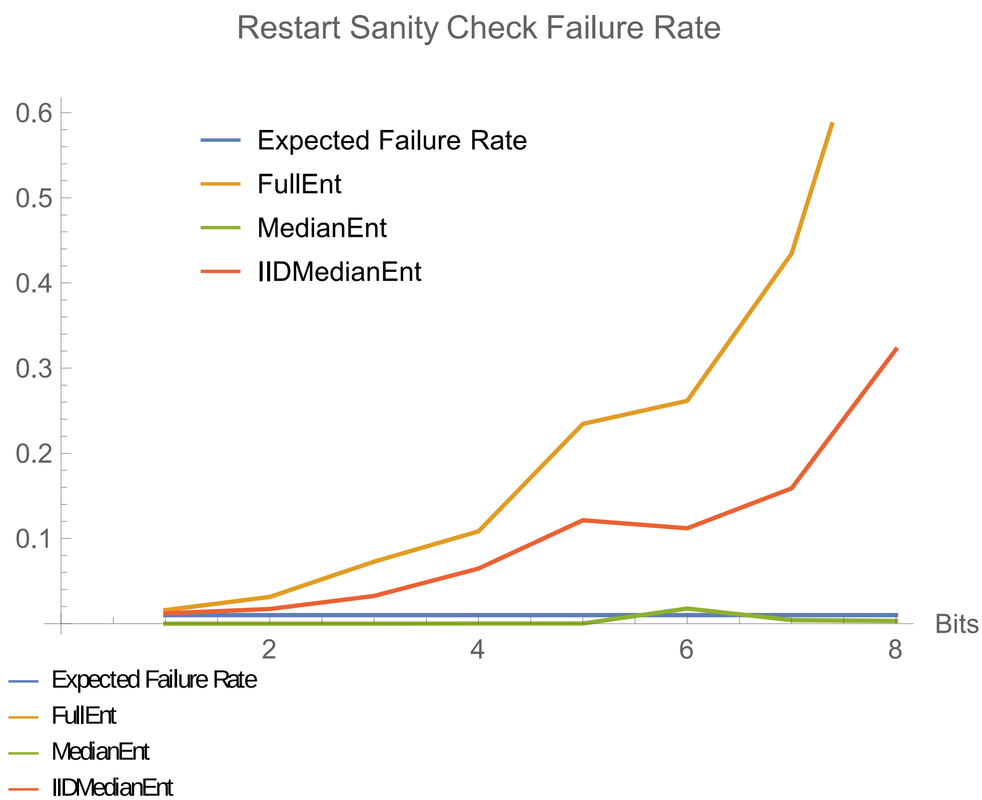

Figure 7

The most technically meaningful failure rate here is the "FullEnt" track, as that reflects the appropriate assumption for the tested data. This assumption also best shows the problems in this test construction. We anticipate somewhat better behavior in actual testing, as the end entropy assessments for the IID and non-IID tracks produce reduced entropy estimates.

For at least the non-IID assessment strategy, these tests appear to be reasonable to apply on data up to 8 bits wide. The non-IID assessment strategy would be expected to commonly fail this test for larger multi-bit samples (1.2% failure rate for 1-bit data, 1.7% failure rate for 2-bit data, 6.5% failure rate for 4-bit data, 32.4% failure rate for 8 bit data, 25.1% failure rate for 16-bit data).

3.2.2 Corrected Simulated Cutoff Restart Sanity Check

For this evaluation, the original test was used (so each test produces a maximum of the count of the per-row/column most likely symbols), but with fixed cutoffs. The cutoffs used for this test were found through simulation of 2,000,000 rounds of single 1000-sample tests (equivalent to a single row or column test) with a targeted per-test $\alpha = 0.000005$. The cutoffs used are described in the following table (along with the binomial / original cutoffs for reference):

Table 3

Min Entropy	Series	Simulated Cutoff	Binomial / Original Cutoffs
0.852803	MedianEnt	625	623
0.995319	IIDMedianEnt	572	571
1	FullEnt	572	570
1.86565	MedianEnt	343	338
1.9908	IIDMedianEnt	317	314
2	FullEnt	318	312
3.75695	MedianEnt	118	113
3.97586	IIDMedianEnt	105	100
4	FullEnt	104	99
7.41556	MedianEnt	23	19
7.87995	IIDMedianEnt	20	16
8	FullEnt	19	15
13.2812	MedianEnt	6	3
14.3163	IIDMedianEnt	5	3
16	FullEnt	4	3

We see above that the simulated cutoffs consistently produce a slightly higher bound than the purely binomial case (these binomial bounds were incorrectly applied to the original version of this check, and are also applied to the corrected binomial check).

The corrected simulated cutoff sanity checks display the expected rates with the same test construction as originally specified (but with updated cutoff values).

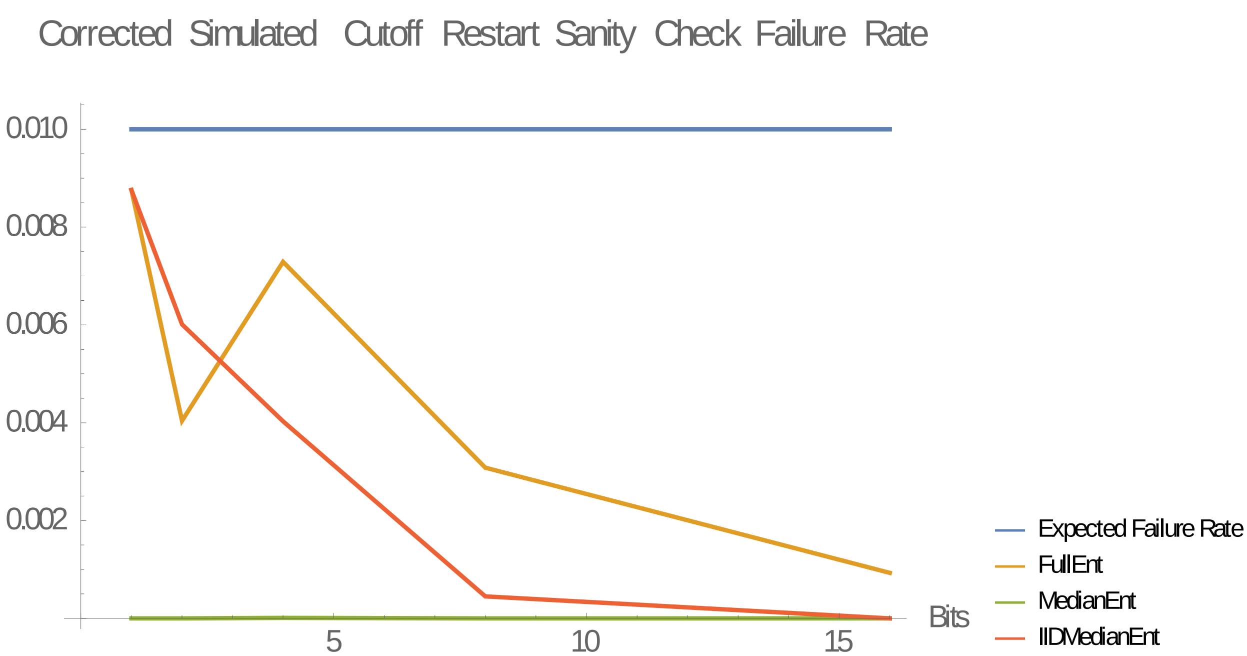

Figure 8

3.2.3 Corrected Binomial Restart Sanity Check

In the corrected binomial version of the Restart Sanity Check, the symbol that is most common for the full 1,000,000 sample data set is established, and then only this most common symbol is counted for each of the column / row counts. This removes the impact of the number of symbols on the test (we now expect a binomial distribution), but leaves the non-independence defect of the row / column count statistics. As a result, the distribution of the resulting p-values still isn't particularly uniform looking, but the proportion of tests passing is well behaved for all symbol widths tested.

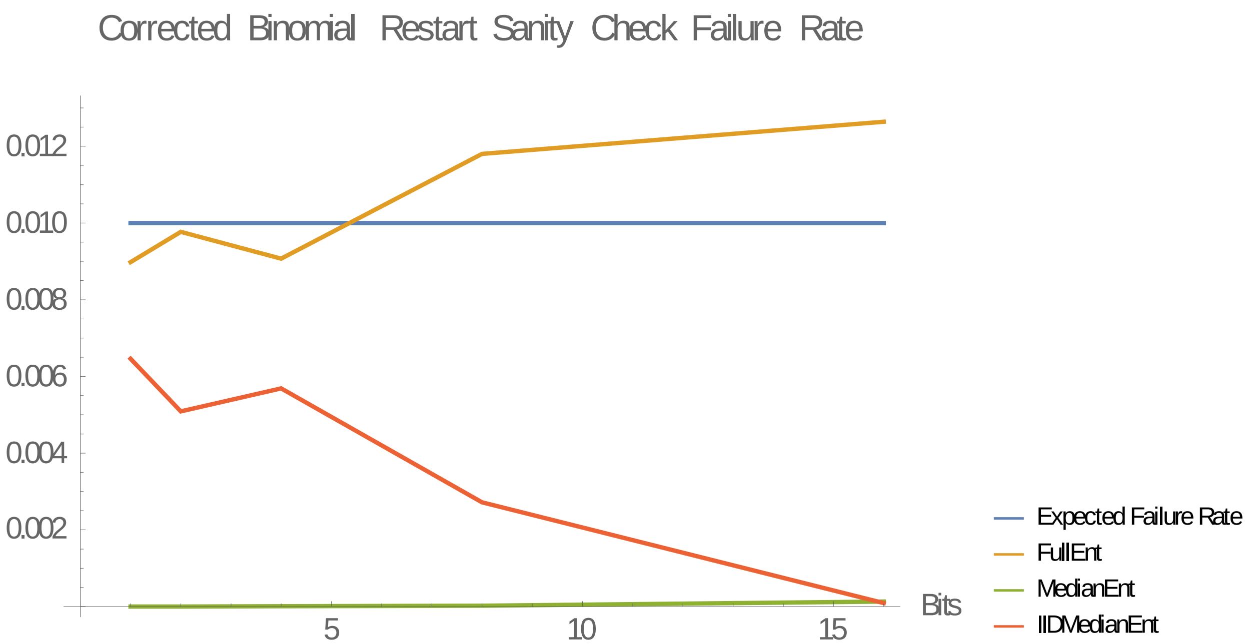

Figure 9

3.2.4 Comments on the Restart Sanity Check

The original test specified is flawed, and will lead to a higher than desired failure rate for all data. Certain types of noise sources would fail at only slightly elevated rates, but due to the catastrophic result of a failure, it is vital to get this test "right".

The two correction proposals that we offer both resolve the main issues with the restart sanity check, but they accomplish these in different ways.

The corrected simulated cutoff version attempts to find reasonable bounds on a per-evaluation basis. A variant of this would be to find the exact cutoff, under the "worst case" assumption of the symbol probabilities described above, also on a per-evaluation basis. This requires that the lab/test tool performs some modest simulation or statistical calculations prior to conducting the restart sanity check.

The corrected binomial sanity check is easier to model statistically, but the power of the resulting statistical testing seems reduced.

3.3 Tests of the IID Assumption

We tested the permutation test, chi-square tests, and length of the longest repeated substring tests independently. They all failed roughly as commonly as expected, and most appear to be reasonably constructed.

3.3.1 Permutation Tests

The construction of the permutation tests doesn't allow for calculation of a p-value directly, but we can examine the percentile of the permutation test reference data result within the full permutation test result data set. This data reflects a shortcut procedure (described on github by the user "zipnemud" here), wherein each permutation test is short circuited once the number of values above and below the reference value is suitably high to guarantee a pass of the permutation tests).[4]

These results reflect 5402 permutation tests on each data width.

In this testing, the "Length of Directional Runs" test and "Length of Runs Based on Median" permutation tests, the resulting percentile distributions departed significantly from the expected uniform distribution for all data widths we tested. Below, we show some representative histograms (the others have the similar "spikey" style distributions).

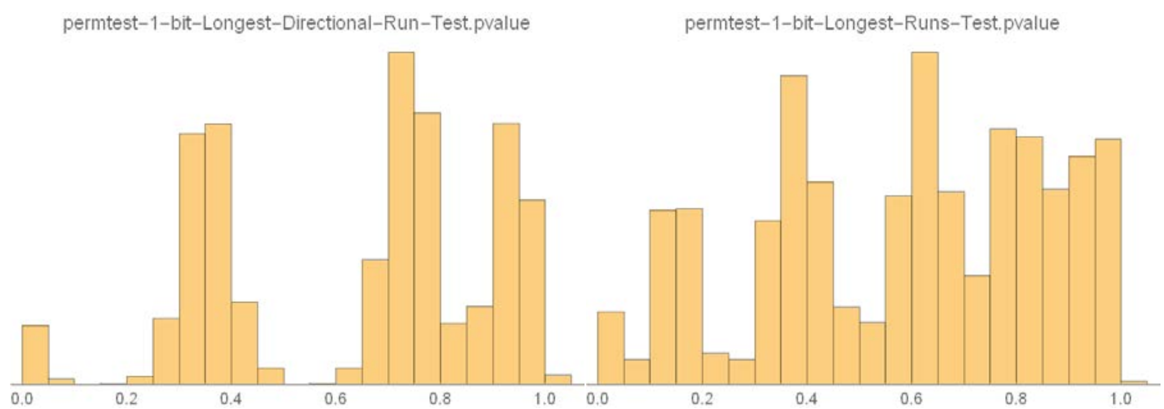

Figure 10

Despite these irregularities, the permutation tests had failure rates that were near the expected rates.

In the figure below, we show the observed permutation test failure rates for various symbol sizes, along with the expected failure rate for all the tests (under the hypothesis that each test is independent, and each test has a failure rate of 1/1000), and a marked "Elevated Failure Rate" (under the hypothesis that each test is independent, and each test has a failure rate of 2/1000).

[4] We mention this, because this short-circuiting will have some result on the distribution of percentiles that we present here (but no impact on the proportion of permutation tests that pass!).

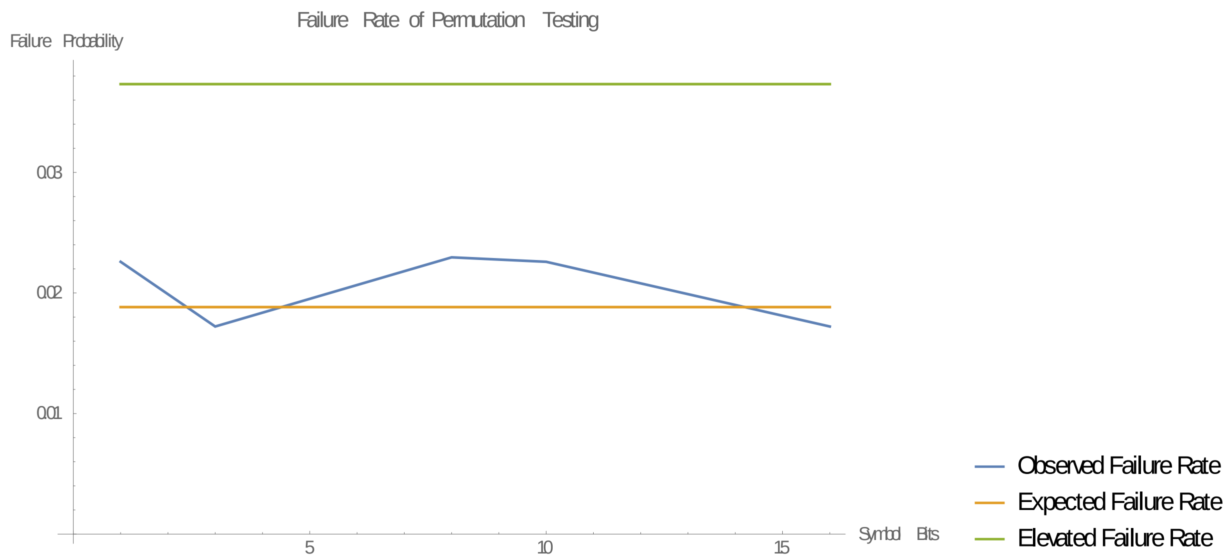

Figure 11

3.3.2 Chi-Square Tests

We conducted over 111,000 tests for each data width, and most of the new Chi-Square tests (both for Independence and Goodness-of-Fit) performed quite well. The exception was the Chi-Square Independence test on wide data which, for datasets of 1,000,000 samples, simply didn't have sufficient data to be well behaved. All the other tests performed quite well (the distribution of the p-values from these tests are fairly uniform!). The histogram for the one problematic test (performed on 10-bit-wide data samples) is shown below.

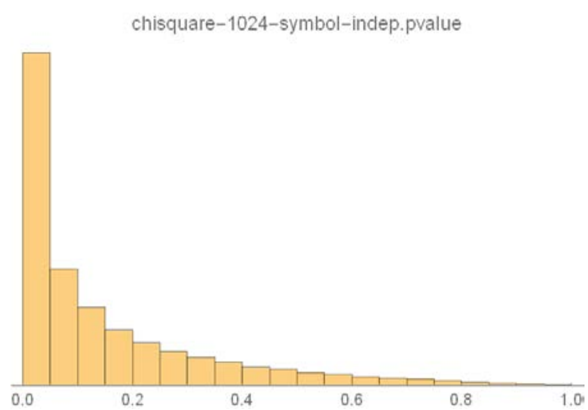

Figure 12

This suggests that for this data set size (1,000,000 sample data sets), the data should be on the order of 8 bits or less (surely less than 10 bits).

3.3.3 Length of the Longest Repeated Substring Test

The LRS test also produces p-values, so we can also assess the distribution of the resulting p-values.

These results reflect over 154,000 LRS tests on each data width.

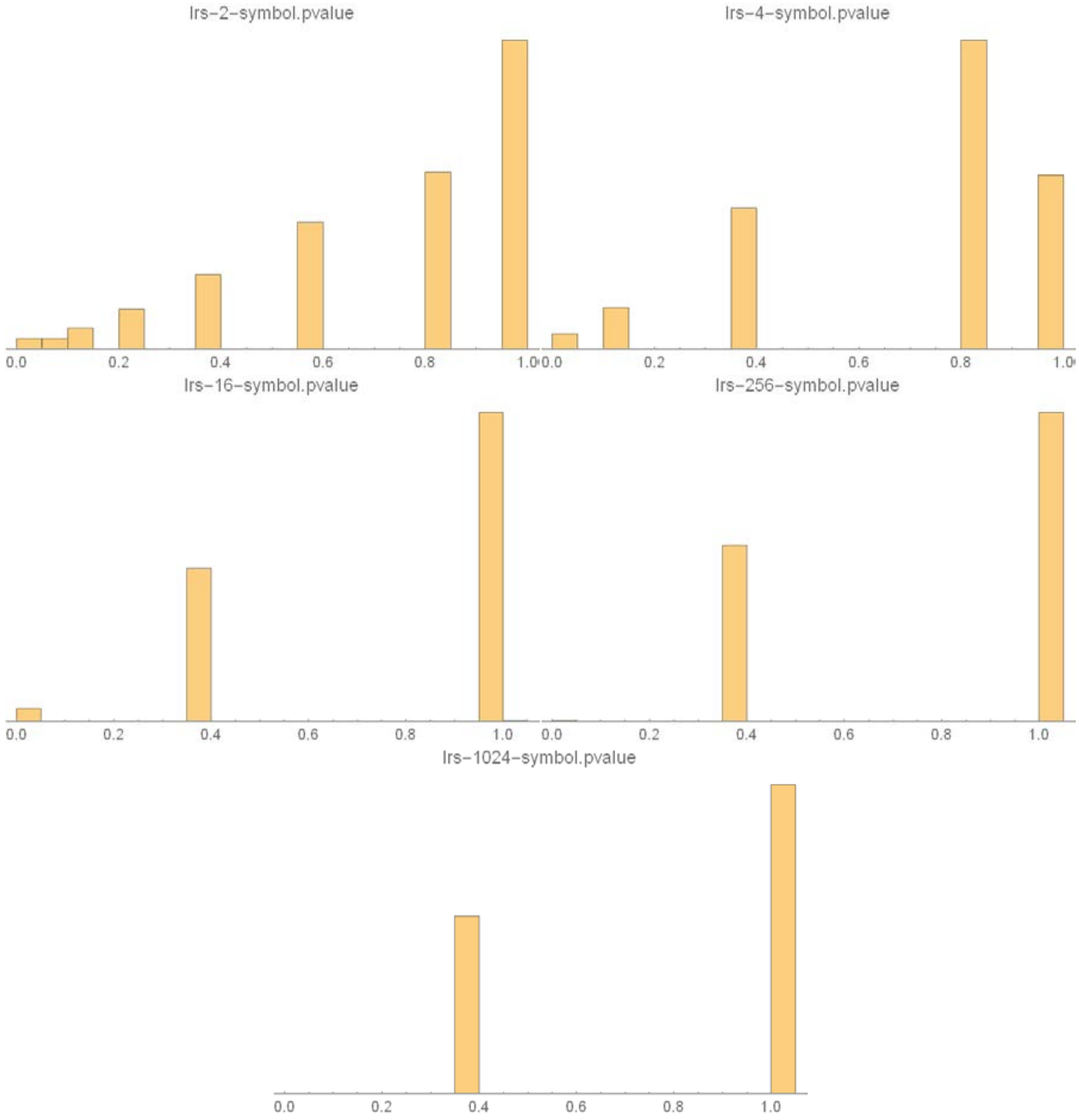

Figure 13

This distribution is clearly non-uniform, so something is a bit amiss, but the pass rates are reasonable for all the tested data widths.

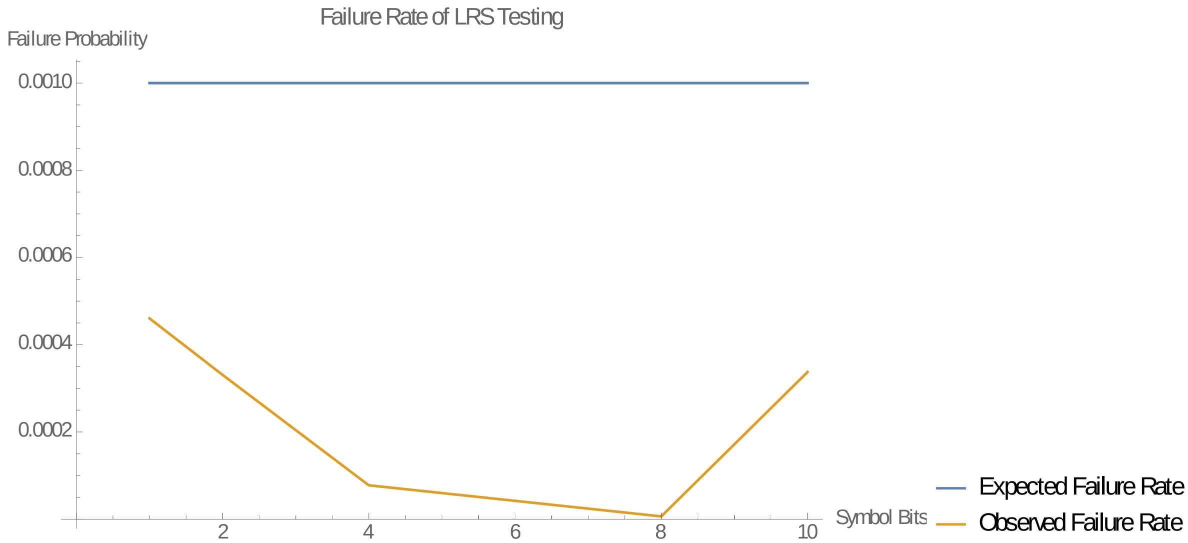

Figure 14

As these test failure probabilities are well below the expected test failure rate, this test seems to perform reasonably for all data lengths tested.

4 Modeled vs Statistically Assessed Min Entropy

In this section, we simulate and model various styles of sources. The output of the simulated sources is statistically analyzed. This work is a larger-scale version of DJ Johnston's 2017 work using NIST's reference python implementation (which is based on the draft 2016 document). [J 2017]

All results are with respect to the non-IID tests. For each parameter setting, the results depicted reflect 100 tests of 1 million samples each, and a single test of 100 million samples (the "large block assessment"). Blue regions show the range of assessments. Green regions reflect modeling range.

For all of these tests, we construct the referenced source based on data that ultimately comes from the Intel RDRAND instruction, which uses an AES-128 CTR_DRBG (so it should be fairly statistically ideal).

4.1 Simple Noise Sources

We start by examining a simple biased bit source. As we vary the probability of producing a '0' symbol, the resulting assessed entropy is depicted in Figure 15.

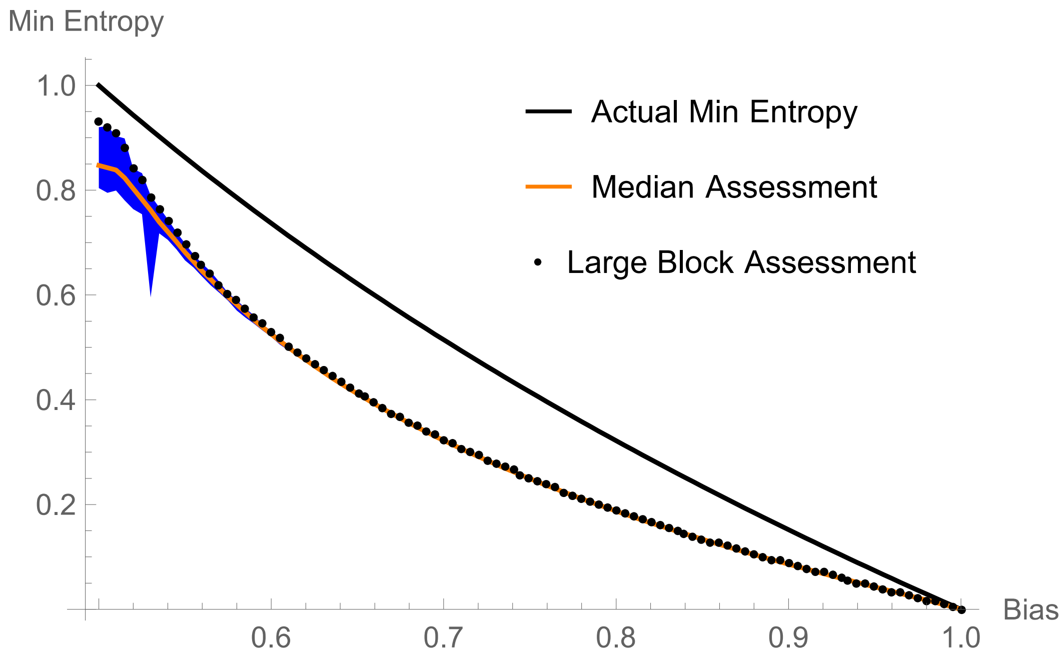

Figure 15

When we instead produce correlated / anti-correlated bits so that

$$\Pr(X_j = a | X_{j-1} = a) = \frac{(c+1)}{2},$$

we then find the results in Figure 16.

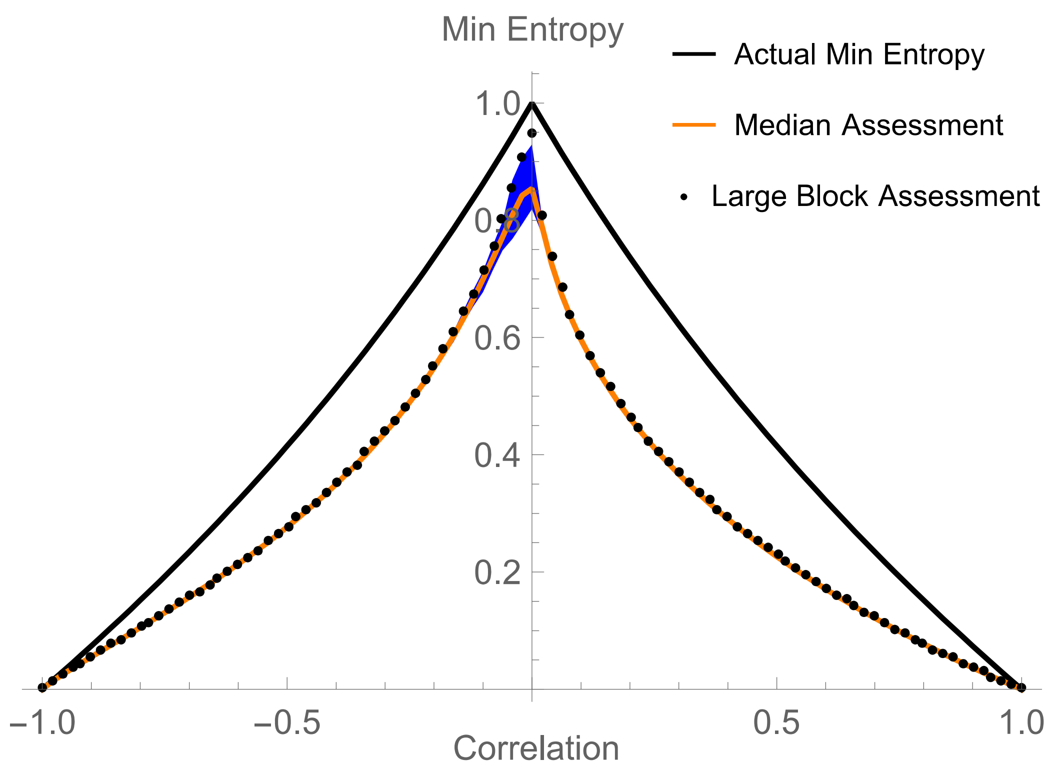

Figure 16

If we take a Gaussian noise source sampled by an 8-bit ADC, we find a somewhat more complicated result, depicted in Figure 17.

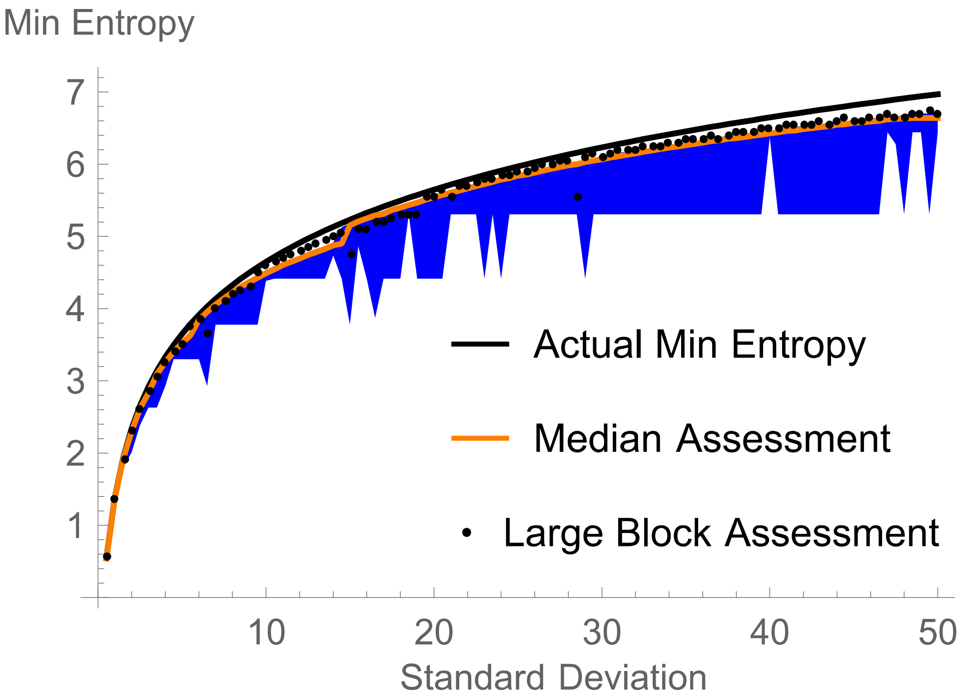

Figure 17

For each of the sources in this section, these are all actually either IID sources, or models where the dependency can be easily teased out by the 90B statistical tests. As such, we would be surprised if the tools overestimate the entropy in these cases.

In general, the statistical assessments seem to be well behaved and generally track the actual min entropy in a pleasing way.

4.2 Perturbed Simple Noise Sources

In these cases, a simple noise source (which we saw is assessed reasonably well) is processed or combined with some additional signal.

We first examine the assessments of a fixed Gaussian source that has some periodic signal added to the random process. We would expect to encounter this when the underlying noise source has electronic design or implementation problems (e.g., insufficient grounding, insufficient power source, etc.) The results of such perturbation is depicted in Figure 18.

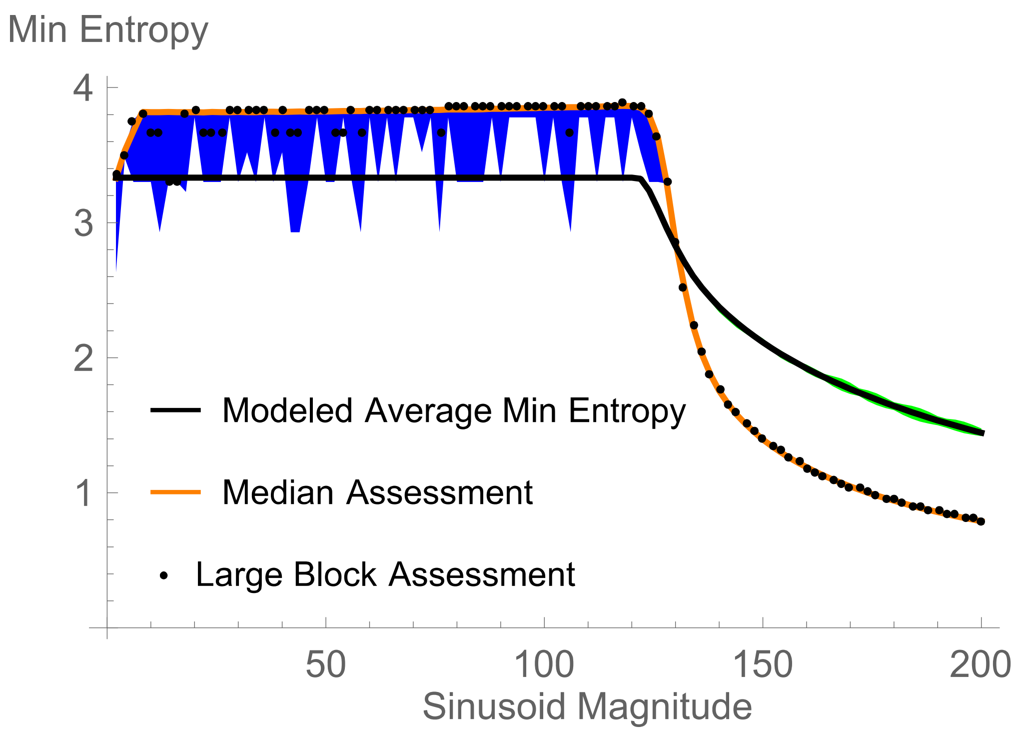

Figure 18

If we take the output of a Gaussian source (of varying standard deviation), and process the data through a simple LFSR, we find the results in Figure 19. This is directly comparable to Figure 17.

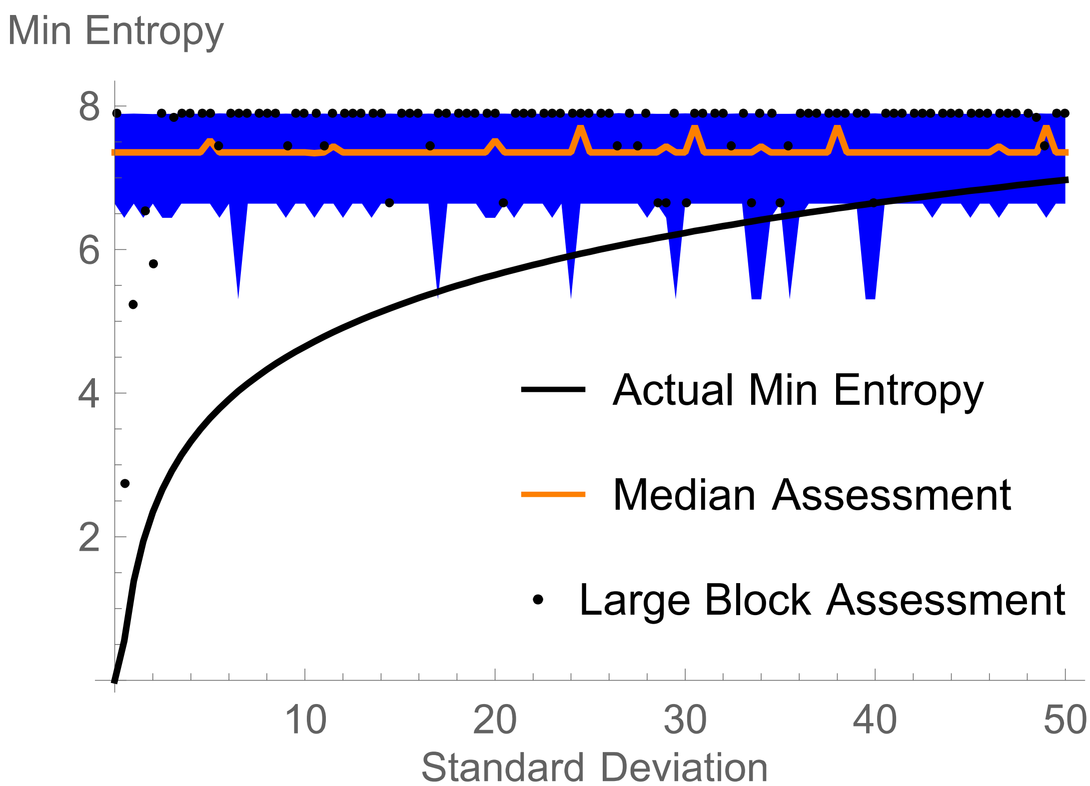

Figure 19

The graphs in this section show us that the addition of small, wholly deterministic variations induce substantial overestimates of entropy. As such, it is **vital** to test only raw data, and to filter out any extraneous signals that are not due to the underlying unpredictable process.

The analysis of the LFSR-conditioned data shows that any conditioning, even if conceptually simple, makes establishing a lower bound for the min entropy via statistical testing impossible.

4.3 More Complicated Noise Sources

In this section, we review a few practical systems that are associated with very commonly fielded noise sources. In these graphs, the green region depicts the range of modeled min entropy.

We first examine noise sources that are reasonably well modeled using the SUMS (Step Update Metastable Source) model. This includes the noise source underlying the Intel RdSeed and RdRand source. We use the model as described by [HKM 2012].

Here, we fix the right step size to 0.1, and vary the left step size, which is consistent with the approach used by Johnston. [J 2017]

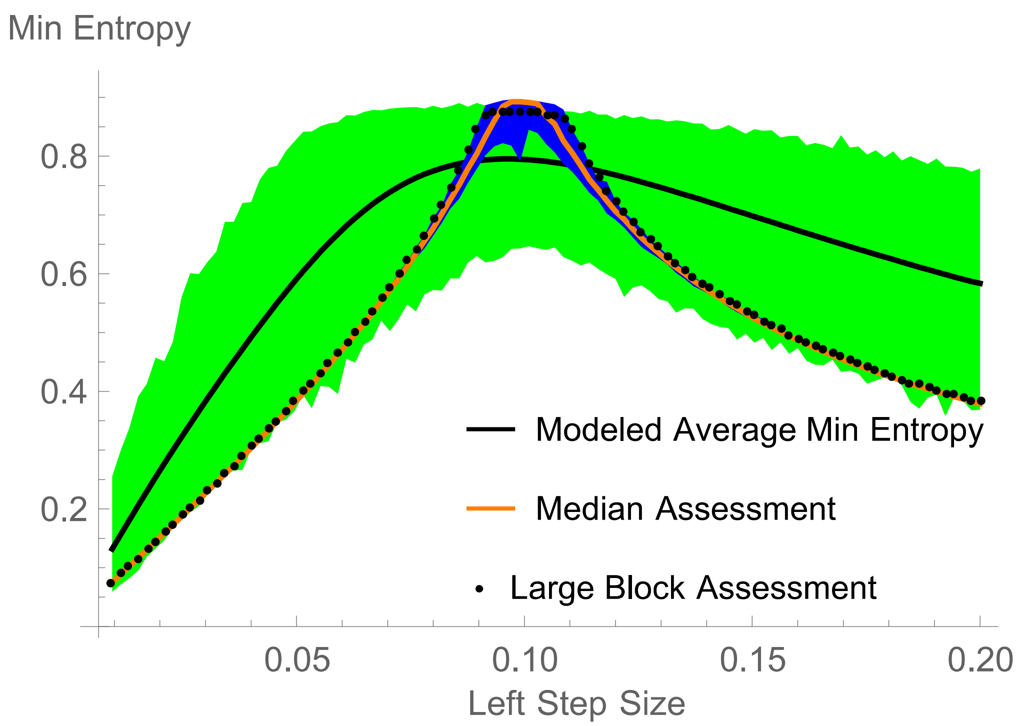

Figure 20

We then examine the results of modeling and statistically assessing the output of a single ring oscillator, which is periodically sampled. Here, we simulate and model a ring oscillator whose nominal frequency is 1 GHz (with a fixed per-data-parameter period distributed normally about 1 ns, with standard deviation of approximately 0.04% of this value), sampled at 1 MHz (these values allow for calculation of the per-sample-period accumulated jitter, based on the per-oscillator-period jitter). Figure 21 shows the modeled and statistically assessed min entropy, as we vary jitter. The accumulated per-sample-period jitter is depicted, presented as a percentage of the ring oscillator period. For this figure, we assume that an attacker cannot predict any portion of this jitter.

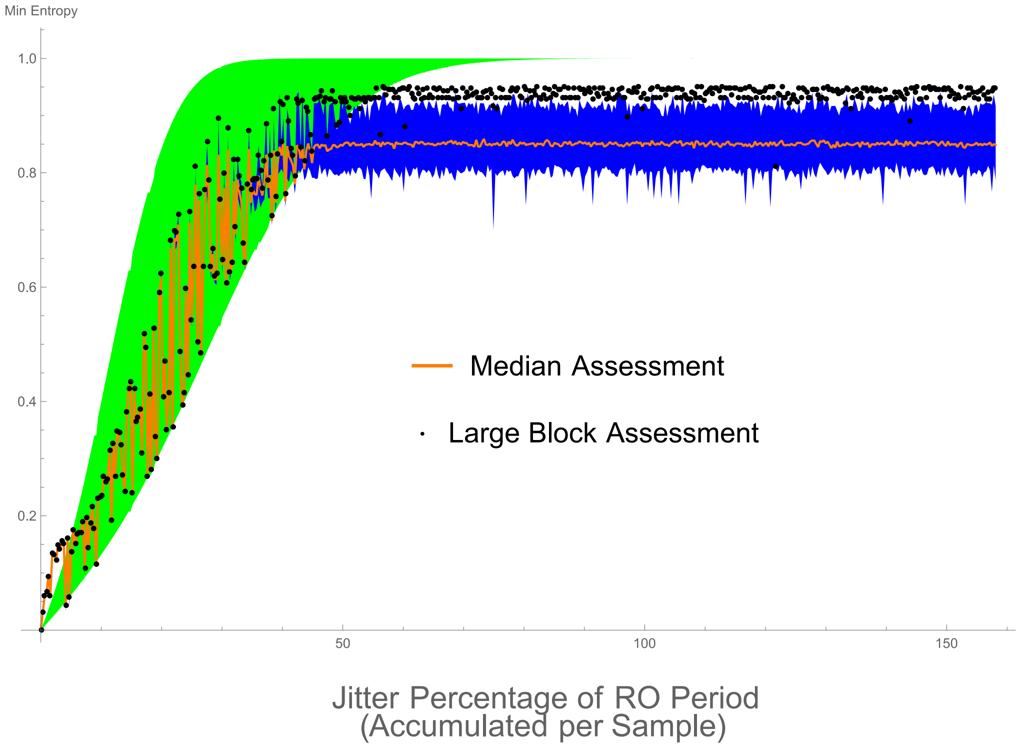

Figure 21

For both of these noise sources, the underlying models are now somewhat complicated, and can return a range of entropy values for each parameter. The statistical testing results generally lie within the expected modeled ranges, but **the lower end of the modeled range is the value that ought to be used for entropy assessment**; this is lower than the value produced by the SP800-90B tests, which suggests that with practical sources, the vendor's assessment of the entropy ($H_{\text{submitter}}$) is of vital importance.

4.4 Practical Considerations for Non-Ideal Noise Sources

If we try to account for variation that is present, but predictable (as in [BLMT 2011]), then we must try to tease out which parts of the variation are due to local Gaussian noise (and are thus un-guessable by any reasonable attacker) and which parts of the variation are due to switching noise, power noise, and any other noise that is fundamentally predicable by any attacker with a sufficiently detailed understanding of the particular noise source design and implementation.

If we take the results of [BLMT 2011] and credit 30% of the standard deviation as being unpredictable (and assume that the attacker can guess the remaining component), then we have a more substantial problem. No statistical test on the output of such a design can distinguish between the predictable variation and unpredictable variation, so under this assumption set, it isn't reasonable to rely on the results of statistical testing to establish a lower bound for min entropy production. This situation is depicted in Figure 22.

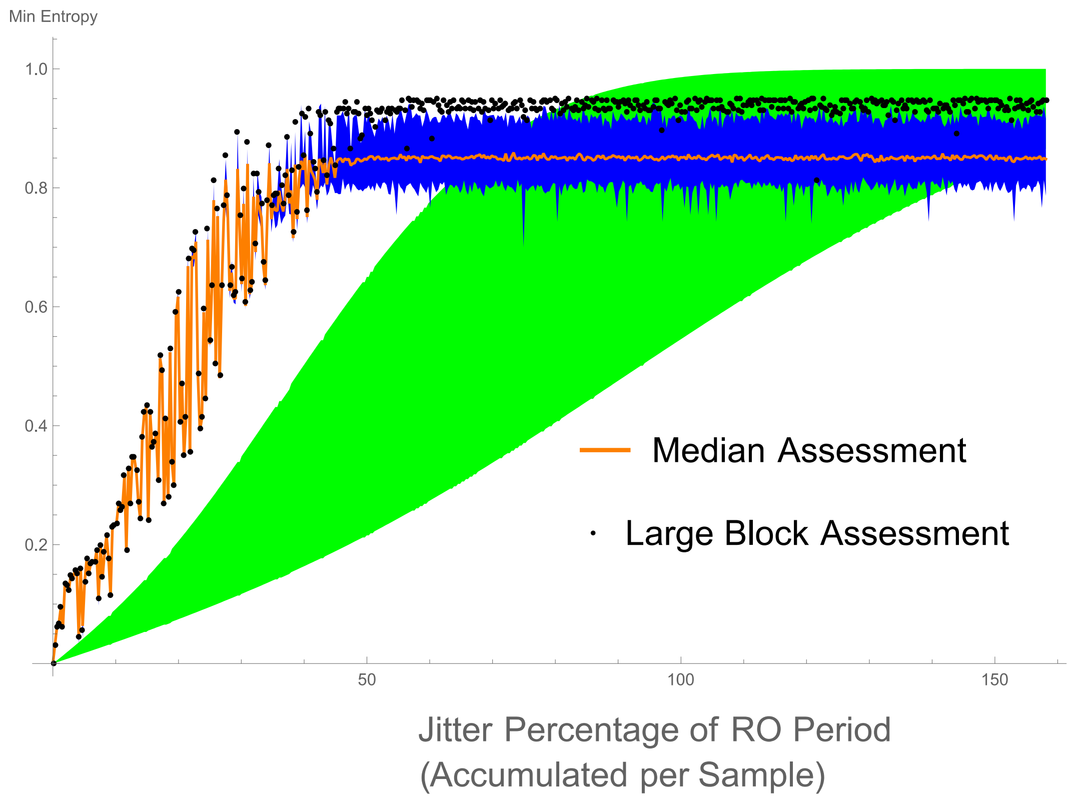

Figure 22

Here we see that a particular statistical assessment corresponds to a range of possible jitter percentages, each having distinct min modeled min entropy rates. In this circumstance, if the jitter percentage and proportion of observed jitter attributable to local Gaussian noise can be determined, then the lower bound of the modeled region should be used as the vendor's $H_{\text{submitter}}$ estimate.

The relationship between overall jitter percentage and the median of the statistically assessed entropy (across many tests) is reasonably stable in simulated oscillators, so one could deduce a lower bound for the per-sample jitter percentage from the statistical testing results. Using this relationship (so long as one can estimate the percentage of observed jitter that is due to local Gaussian noise), one could also back into an $H_{\text{submitter}}$ estimate using a combination of the modeled and statistical results, as follows:

1. Run statistical testing on a large sample of output from the ring oscillator, and use these results to establish a lower bound for the overall per-sample jitter percentage.
2. Use the estimated lower bound for the overall per-sample jitter percentage, σ, and the expected proportion of this jitter standard deviation due to local Gaussian noise, g, to estimate the per-sample jitter percentage that is due to local Gaussian noise, $g\sigma$, and then use this parameter within a ring oscillator model, $H_{\text{model\_min}}(g\sigma)$. This model produces a lower min entropy bound appropriate for use as $H_{\text{submitter}}$.

We depict an approach to arriving at min-entropy bounds in Figure 23; in this graph, the cyan region depicts the ideal modeled min entropy range, the red curve is the statistical assessment lower bound, the blue curve is the statistical assessment upper bound, and the green region depicts the modeled min entropy range where only 30% of the observed jitter is local Gaussian jitter (and is thus unpredictable to an attacker). In this diagram, we depict the case where the statistical tests indicate a result of 0.7 bits of min entropy per bit.

For the corrected model / statistical lower bound, we first follow this statistical result value horizontally until it intersects with the statistical assessment upper bound (they meet at a jitter value of approximately 22.4%), and then vertically down, to the reduced jitter modeled lower bound (approximately 0.0844 bits of min entropy).

For the corrected model / statistical upper bound, we follow this statistical result value horizontally until it intersects with the statistical assessment lower bound (they meet at a jitter value of approximately 37.7%), and then vertically down, to the reduced jitter modeled lower bound (approximately 0.152 bits of min entropy).

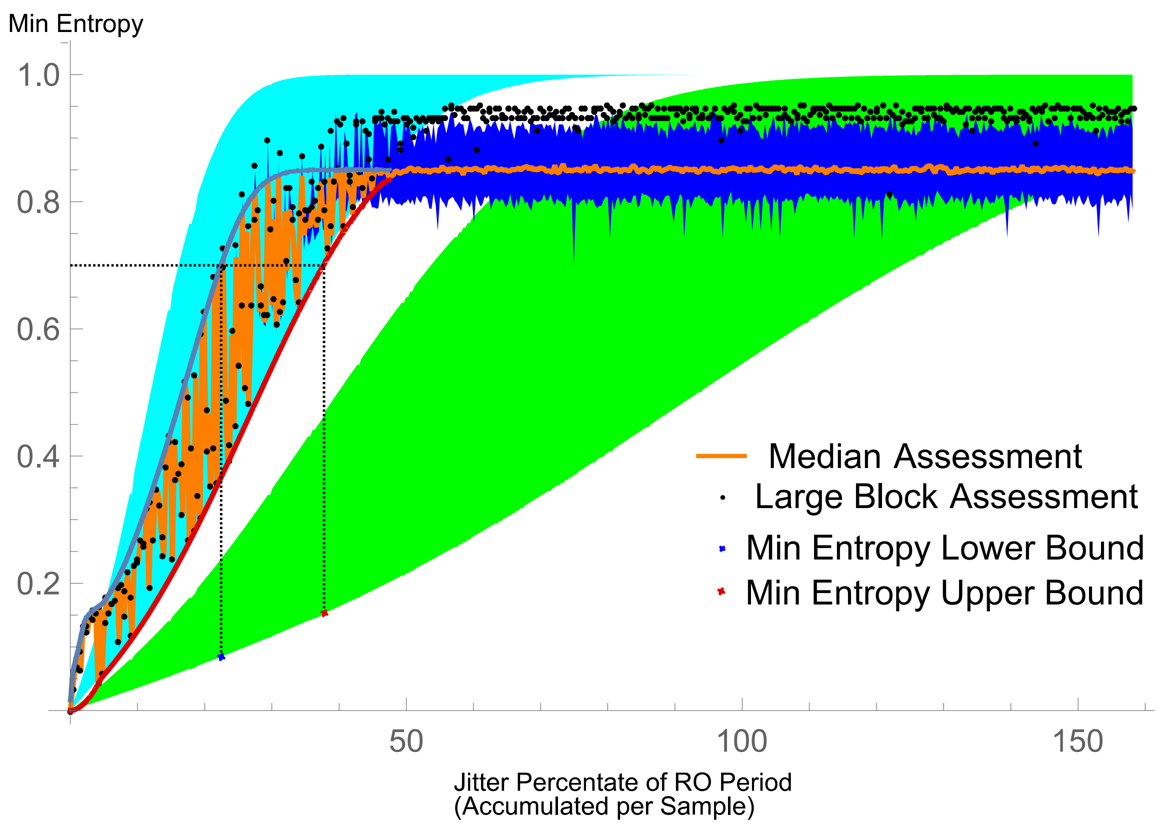

Figure 23

4.5 Overall Observations

The statistical testing results for a particular source form a distribution; for more complicated sources, this distribution tends to be wider. Single results aren't very meaningful, as they don't provide insight into the underlying statistical result distribution.

The underlying source of uncertainty needs to be well understood and (to the degree possible) directly sampled. When the output of the noise source is influenced by processes that are predictable (for a sufficiently informed attacker), this influence should either be filtered out prior to statistical analysis, or some sort of model-based correction should be applied. If perturbed data is directly analyzed, the resulting min-entropy assessment is likely to be artificially high.

The statistical tests seem to do a good job of assessing simple noise sources, but have more trouble at providing a lower bound for more complicated noise sources. For complicated noise sources, the statistical testing results generally reflect values within the modeled min entropy envelope, but these statistical testing results often don't conservatively estimate the noise source min entropy. This suggests that the assessment of non-trivial non-IID sources should be further reduced below the value produced through statistical assessment.

5 References

[BLMT 2011] Baudet, Lubicz, Micolod, and Tassiaux. *On the security of oscillator-based random number generators*. Journal of Cryptology, April 2011, Volume 24, Issue 2.

[BBFV 2010] Bochard, Bernard, Fischer, and Valtchanov. *True-Randomness and Pseudo-Randomness in Ring Oscillator-Based True Random Number Generators*. International Journal of Reconfigurable Computing, Vol. 2010.

[HKM 2012] Hamburg, Kocher, and Marson. *Analysis of Intel's Ivy Bridge Digital Random Number Generator*.

[HD] Patrick Hagerty and Tom Draper. *Entropy Bounds and Statistical Tests*. https://csrc.nist.gov/csrc/media/events/random-bit-generation-workshop-2012/documents/hagerty_entropy_paper.pdf

[J 2017] Johnston, David. *STS-2.1.2 and SP800-90B Assessment Suite Anomalous results*. https://github.com/dj-on-github/90B_check

[SP800-90B] Turan, Barker, Kelsey, McKay, Baish, and Boyle. *Special Publication 800-90B: Recommendation for Entropy Sources Used for Random Bit Generation*. January 2018.