Implementation Notes for entropy estimation based on NIST SP 800-90B non-IID track

Gen'ya SAKURAI

September 28, 2025

1 Implementation notes for numerical computation of entropy estimates

1.1 Notes for the Markov estimate

In this section, we follow the convention using a column vector of probabilities $\boldsymbol{P}$ and a left stochastic matrix $\boldsymbol{T}$.

$$\boldsymbol{P} \equiv \begin{bmatrix} p_0 \\ p_1 \end{bmatrix} \tag{1}$$

$$\boldsymbol{T} \equiv \begin{bmatrix} t_{0,0} & t_{0,1} \\ t_{1,0} & t_{1,1} \end{bmatrix} \tag{2}$$

Note here that 6.3.3 of NIST SP 800-90B[1] uses the convention of a row vector of probabilities and a right stochastic matrix. So the transition matrix components $t_{i,j}$ in this section will be interpreted as $P_{j,i}$ in 6.3.3 of NIST SP 800-90B[1].

Considering a sequence of binary-valued samples of length λ, there are following four possible combinations of the first sample value and the last sample value:

a) $0 \to 0$
b) $0 \to 1$
c) $1 \to 0$
d) $1 \to 1$

For each combination, the probability of occurrence of the most likely sequence is expressed by the following equation, by using the parameters μ and ν:

a)

$$f_a(\boldsymbol{T}, \mu, \nu) \equiv p_0 t_{00}^{\nu} t_{11}^{\lambda-1-2\mu-\nu} t_{01}^{\mu} t_{10}^{\mu} \tag{3}$$

$$\begin{cases} 0 \le 2\mu < \lambda - 1 \\ 0 \le \nu \le \lambda - 1 \\ 0 \le \lambda - 1 - 2\mu - \nu \le \lambda - 1 \end{cases} \tag{4}$$

b)

$$f_b(\boldsymbol{T}, \mu, \nu) \equiv p_0 t_{00}^{\nu} t_{11}^{\lambda-2-2\mu-\nu} t_{01}^{\mu} t_{10}^{\mu+1} \tag{5}$$

$$\begin{cases} 0 \le 2\mu < \lambda - 1 \\ 0 \le \nu < \lambda - 1 \\ 0 \le \lambda - 2 - 2\mu - \nu \le \lambda - 2 \end{cases} \tag{6}$$

c)

$$f_c(\boldsymbol{T}, \mu, \nu) \equiv p_1 t_{00}^{\nu} t_{11}^{\lambda-2-2\mu-\nu} t_{01}^{\mu+1} t_{10}^{\mu} \tag{7}$$

$$\begin{cases} 0 \le 2\mu < \lambda - 1 \\ 0 \le \nu < \lambda - 1 \\ 0 \le \lambda - 2 - 2\mu - \nu \le \lambda - 2 \end{cases} \tag{8}$$

d)

$$f_d(\boldsymbol{T}, \mu, \nu) \equiv p_1 t_{00}^{\nu} t_{11}^{\lambda-1-2\mu-\nu} t_{01}^{\mu} t_{10}^{\mu} \tag{9}$$

$$\begin{cases} 0 \le 2\mu < \lambda - 1 \\ 0 \le \nu \le \lambda - 1 \\ 0 \le \lambda - 1 - 2\mu - \nu \le \lambda - 1 \end{cases} \tag{10}$$

If we denote κ as the exponent of t_{11}, then the possible values of μ, ν, and κ are expressed as integer coordinates in the approximately triangular region (to be precise, truncated triangle) shown in Figure 1.

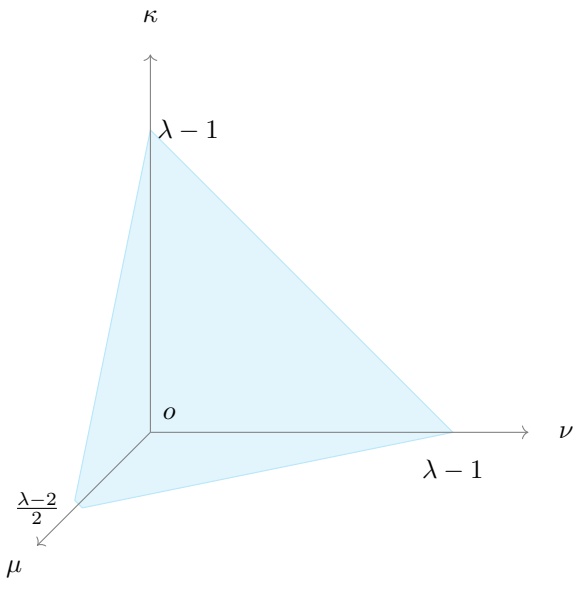

Figure 1 Parameter plane of the exponents κ, μ, ν

In order to calculate the extrema of f_a, f_b, f_c, and f_d, assuming that μ, and ν are continuous variables, we first consider the partial derivatives of $f_a(\boldsymbol{T}, \mu, \nu)$ with respect to μ, and ν.

$$\begin{aligned}\frac{\partial}{\partial\mu} f_a(\boldsymbol{T},\mu,\nu) &= (-2\ln t_{11} + \ln t_{01} + \ln t_{10}) f_a(\boldsymbol{T},\mu,\nu) \\ &= \ln\frac{t_{01}t_{10}}{t_{11}^2} f_a(\boldsymbol{T},\mu,\nu) \qquad (11)\\ \frac{\partial}{\partial\nu} f_a(\boldsymbol{T},\mu,\nu) &= (\ln t_{00} - \ln t_{11}) f_a(\boldsymbol{T},\mu,\nu) \\ &= \ln\frac{t_{00}}{t_{11}} f_a(\boldsymbol{T},\mu,\nu) \qquad (12)\end{aligned}$$

Taking into accout eqs. (11) and (12), and the property that the value $f_a(\boldsymbol{T}, \mu, \nu)$ is non-negative, a rough phase diagram will be obtained as shown in Figure 2, by taking the horizontal axis as $\ln\frac{t_{00}}{t_{11}}$, and the vertical axis as $\frac{1}{2}\ln\frac{t_{01}t_{10}}{t_{11}^2}$.

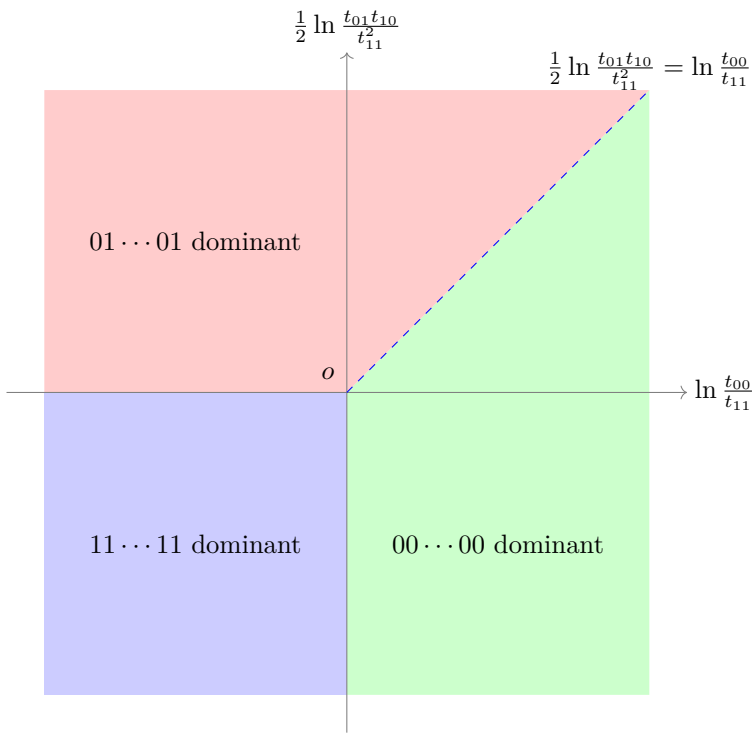

Figure 2 Phase diagram of most likely sequence based on $f_a(\boldsymbol{T}, \mu, \nu)$ with respect to components of stochastic matrix $\boldsymbol{T}$

First, if we consider the third quadrant, for example, the conditional probability t_{11} is greater than or equal to t_{00} and $\sqrt{t_{01}t_{10}}$, then the sequence $11\cdots11$ will be dominant.

Next, for the other quadrants, let us consider whether $00\cdots00$ or $01\cdots01$ is the dominant sequence. In particular, for the first quadrant, since right hand sides of eqs. (11) and (12) are positive, let us examine whether $00\cdots00$ or $0101\cdots0101$ is the dominant sequence. Since at least $11\cdots11$ is not a dominant sequence, we can limit the discussion with $\kappa = 0$, 1, the parameter region of possible values of μ and ν is restricted to a line segment in Figure 1. In this case, the total differentiation of f_a is then expressed by the following equation:

$$\begin{aligned}\Delta f_a &= \left(\frac{\partial f_a}{\partial\mu} + \frac{\partial f_a}{\partial\nu}\frac{\mathrm{d}\nu}{\mathrm{d}\mu}\right)\Delta\mu \\ &= \left(\ln\frac{t_{01}t_{10}}{t_{00}^2}\right)\Delta\mu \qquad (13)\end{aligned}$$

From this equation, if RHS of eq. (13) is positive, f_a increases with respect to μ and $01\cdots01$ becomes the dominant sequence, and conversely, if RHS of eq. (13) is negative, $00\cdots00$ becomes the dominant sequence. This will be determined by the following coefficient:

$$\ln\frac{t_{01}t_{10}}{t_{00}^2} = \ln\frac{t_{01}t_{10}}{t_{11}^2} - 2\ln\frac{t_{00}}{t_{11}} \qquad (14)$$

Figure 2 shows a dashed diagonal straight line in the first quadrant, on which the RHS of eq. (14) is zero. In other words, in the region above this line, $01\cdots01$ is the dominant sequence, and $00\cdots00$ is the dominant sequence in the region below the line.

From the above, we have to consider the following three dominant sequences:

i) $00\cdots00$ dominant
ii) $01\cdots01$ dominant
iii) $11\cdots11$ dominant

For each combination of $f_a, f_b, f_c,\ f_d$, and above three dominant sequences i), ii), and iii), the following expressions are obtained giving the maximum probability with respect to μ and ν:

a)-i) $00\cdots00$ dominant

$$\max_{\mu,\nu}(f_a(\boldsymbol{T},\mu,\nu)) = p_0 t_{00}^{\lambda-1} \tag{15}$$

a)-ii) $01\cdots01$ dominant

$$\max_{\mu,\nu}(f_a(\boldsymbol{T},\mu,\nu)) = p_0 t_{00} t_{01}^{(\lambda-2)/2} t_{10}^{(\lambda-2)/2}, \text{or} \tag{16}$$

$$\max_{\mu,\nu}(f_a(\boldsymbol{T},\mu,\nu)) = p_0 t_{11} t_{01}^{(\lambda-2)/2} t_{10}^{(\lambda-2)/2} \tag{17}$$

a)-iii) $11\cdots11$ dominant

$$\max_{\mu,\nu}(f_a(\boldsymbol{T},\mu,\nu)) = p_0 t_{11}^{\lambda-3} t_{01} t_{10} \tag{18}$$

b)-i) $00\cdots00$ dominant

$$\max_{\mu,\nu}(f_b(\boldsymbol{T},\mu,\nu)) = p_0 t_{00}^{\lambda-2} t_{10} \tag{19}$$

b)-ii) $01\cdots01$ dominant

$$\max_{\mu,\nu}(f_b(\boldsymbol{T},\mu,\nu)) = p_0 t_{01}^{(\lambda-2)/2} t_{10}^{\lambda/2} \tag{20}$$

b)-iii) $11\cdots11$ dominant

$$\max_{\mu,\nu}(f_b(\boldsymbol{T},\mu,\nu)) = p_0 t_{11}^{\lambda-2} t_{10} \tag{21}$$

c)-i) $00\cdots00$ dominant

$$\max_{\mu,\nu}(f_c(\boldsymbol{T},\mu,\nu)) = p_1 t_{00}^{\lambda-2} t_{01} \tag{22}$$

c)-ii) $01\cdots01$ dominant

$$\max_{\mu,\nu}(f_c(\boldsymbol{T},\mu,\nu)) = p_1 t_{01}^{\lambda/2} t_{10}^{(\lambda-2)/2} \tag{23}$$

c)-iii) $11\cdots11$ dominant

$$\max_{\mu,\nu}(f_c(\boldsymbol{T},\mu,\nu)) = p_1 t_{11}^{\lambda-2} t_{01} \tag{24}$$

d)-i) $00\cdots00$ dominant

$$\max_{\mu,\nu}(f_d(\boldsymbol{T},\mu,\nu)) = p_1 t_{00}^{\lambda-3} t_{01} t_{10} \tag{25}$$

d)-ii) $01\cdots01$ dominant

$$\max_{\mu,\nu}(f_d(\boldsymbol{T},\mu,\nu)) = p_1 t_{00} t_{01}^{(\lambda-2)/2} t_{10}^{(\lambda-2)/2}, \text{or} \tag{26}$$

$$\max_{\mu,\nu}(f_d(\boldsymbol{T},\mu,\nu)) = p_1 t_{11} t_{01}^{(\lambda-2)/2} t_{10}^{(\lambda-2)/2} \tag{27}$$

d)-iii) $11\cdots11$ dominant

$$\max_{\mu,\nu}(f_d(\boldsymbol{T},\mu,\nu)) = p_1 t_{11}^{\lambda-1} \tag{28}$$

Note here that we considered the property that λ is even in the above expressions. Also note that all bounary conditions are considered in the above calculation, combination of first sample values and last sample values.
From the above, we have to compare 14 expressions listed in Table 1.

Table 1 Probabilities of the most or second most likely sequences of length λ

No.	Sequence	Probability	$-\log_2(\text{Probability})/\lambda$	Notes
1	$0000\cdots0000$	$p_0 \times t_{00}^{\lambda-1}$	$-\left[\frac{\lambda-1}{\lambda}\log_2 t_{00} + \frac{1}{\lambda}\log_2 p_0\right]$	[a], [b]
2	$0101\cdots0101001010\cdots1010$	$p_0 \times t_{00} \times t_{01}^{(\lambda-2)/2} \times t_{10}^{(\lambda-2)/2}$	$-\left[\frac{\lambda-2}{2\lambda}\log_2 (t_{01}t_{10}) + \frac{1}{\lambda}\log_2 (p_0 \times t_{00})\right]$	[b]
3	$0101\cdots0101101010\cdots1010$	$p_0 \times t_{11} \times t_{01}^{(\lambda-2)/2} \times t_{10}^{(\lambda-2)/2}$	$-\left[\frac{\lambda-2}{2\lambda}\log_2 (t_{01}t_{10}) + \frac{1}{\lambda}\log_2 (p_0 \times t_{11})\right]$	
4	$0111\cdots1110$	$p_0 \times t_{11}^{\lambda-3} \times t_{01} \times t_{10}$	$-\left[\frac{\lambda-3}{\lambda}\log_2 t_{11} + \frac{1}{\lambda}\log_2 (p_0 \times t_{01} \times t_{10})\right]$	
5	$0000\cdots0001$	$p_0 \times t_{00}^{\lambda-2} \times t_{10}$	$-\left[\frac{\lambda-2}{\lambda}\log_2 t_{00} + \frac{1}{\lambda}\log_2 (p_0 \times t_{10})\right]$	
6	$0101\cdots0101$	$p_0 \times t_{01}^{(\lambda-2)/2} \times t_{10}^{\lambda/2}$	$-\left[\frac{\lambda-2}{2\lambda}\log_2 (t_{01}t_{10}) + \frac{1}{\lambda}\log_2 (p_0 \times t_{10})\right]$	[a], [b]
7	$0111\cdots1111$	$p_0 \times t_{11}^{\lambda-2} \times t_{10}$	$-\left[\frac{\lambda-2}{\lambda}\log_2 t_{11} + \frac{1}{\lambda}\log_2 (p_0 \times t_{10})\right]$	[a], [c]
8	$1000\cdots0000$	$p_1 \times t_{00}^{\lambda-2} \times t_{01}$	$-\left[\frac{\lambda-2}{\lambda}\log_2 t_{00} + \frac{1}{\lambda}\log_2 (p_1 \times t_{01})\right]$	[a], [b]
9	$1010\cdots1010$	$p_1 \times t_{01}^{\lambda/2} \times t_{10}^{(\lambda-2)/2}$	$-\left[\frac{\lambda-2}{2\lambda}\log_2 (t_{01}t_{10}) + \frac{1}{\lambda}\log_2 (p_1 \times t_{01})\right]$	[a], [b]
10	$1111\cdots1110$	$p_1 \times t_{11}^{\lambda-2} \times t_{01}$	$-\left[\frac{\lambda-2}{\lambda}\log_2 t_{11} + \frac{1}{\lambda}\log_2 (p_1 \times t_{01})\right]$	
11	$1000\cdots0001$	$p_1 \times t_{00}^{\lambda-3} \times t_{01} \times t_{10}$	$-\left[\frac{\lambda-3}{\lambda}\log_2 t_{00} + \frac{1}{\lambda}\log_2 (p_1 \times t_{01} \times t_{10})\right]$	
12	$1010\cdots1010100101\cdots0101$	$p_1 \times t_{00} \times t_{01}^{(\lambda-2)/2} \times t_{10}^{(\lambda-2)/2}$	$-\left[\frac{\lambda-2}{2\lambda}\log_2 (t_{01}t_{10}) + \frac{1}{\lambda}\log_2 (p_1 \times t_{00})\right]$	
13	$1010\cdots1010110101\cdots0101$	$p_1 \times t_{11} \times t_{01}^{(\lambda-2)/2} \times t_{10}^{(\lambda-2)/2}$	$-\left[\frac{\lambda-2}{2\lambda}\log_2 (t_{01}t_{10}) + \frac{1}{\lambda}\log_2 (p_1 \times t_{11})\right]$	[c]
14	$1111\cdots1111$	$p_1 \times t_{11}^{\lambda-1}$	$-\left[\frac{\lambda-1}{\lambda}\log_2 t_{11} + \frac{1}{\lambda}\log_2 p_1\right]$	[a], [b]

[a] Included in 6.3.3 of NIST SP 800-90B[1].

[b] Explicitly included in paragraph 516 of AIS 20/31 Version 2.35 - DRAFT [2], by using the relation $m \equiv \lambda - 1$.

[c] Implicitly included in paragraph 516 of AIS 20/31 Version 2.35 - DRAFT [2], by using the relation $m \equiv \lambda - 1$ and by relabeling the state space $\Omega = \{0, 1\}$.

1.2 Notes for the collision estimate

In this section, we try to rewrite the following equation using elementary functions:

$$F(1/z) = \Gamma(3, z)z^{-3}\exp(z) \tag{29}$$

In NIST SP 800-90B[1], it is documented to evaluate incomplete gamma function using continued fraction. However, we try to obtain simpler expression using known properties of incomplete gamma function without using continued fraction.

First we try to use the following equations (see 8.4.8 and 8.4.11 in [3]).

$$\Gamma(n+1, z) = n!\exp(-z)e_n(z), \tag{30}$$

$$e_n(z) = \sum_{k=0}^{n}\frac{z^k}{k!} \tag{31}$$

With these two equations, $F(1/z)$ can be rewritten to as follows:

$$\begin{aligned}
F(1/z) &= \Gamma(3, z)z^{-3}\exp(z) \\
&= 2!\exp(-z)e_2(z)z^{-3}\exp(z) \\
&= 2z^{-3}e_2(z) \\
&= 2z^{-3}\sum_{k=0}^{2}\frac{z^k}{k!} \\
&= 2z^{-3}\left(1 + z + \frac{z^2}{2}\right) \\
&= z^{-1}\left(2z^{-2} + 2z^{-1} + 1\right)
\end{aligned} \tag{32}$$

By replacing the argument $1/z$ with q, the following expression can be obtained:

$$F(q) = q\left(2q^2 + 2q + 1\right) \tag{33}$$

From this expression, the range of $F(q)$ is $[0, \frac{5}{4}]$, with respect to the domain $q \in [0, 0.5]$.

If we denote $g(p)$ as the right hand side (RHS) of equation to be solved, in step 7 of 6.3.2 of NIST SP 800-90B[1], $g(p)$ can be rewritten by using Eq.(33).

$$\begin{aligned}
g(p) &\equiv pq^{-2}\left[1 + \frac{1}{2}(p^{-1} - q^{-1})\right]F(q) - pq^{-1}\frac{1}{2}(p^{-1} - q^{-1}) \\
&= pq^{-2}\left[1 + \frac{1}{2}(p^{-1} - q^{-1})\right]q\left(2q^2 + 2q + 1\right) - pq^{-1}\frac{1}{2}(p^{-1} - q^{-1}) \\
&= pq^{-1}\left[1 + \frac{1}{2}(p^{-1} - q^{-1})\right]\left(2q^2 + 2q + 1\right) - pq^{-1}\frac{1}{2}(p^{-1} - q^{-1}) \\
&= pq^{-1}\left[1 + \frac{1}{2}(p^{-1} - q^{-1})\right] - pq^{-1}\frac{1}{2}(p^{-1} - q^{-1}) \\
&+ pq^{-1}\left[1 + \frac{1}{2}(p^{-1} - q^{-1})\right]\left(2q^2 + 2q\right) \\
&= pq^{-1} + 2p\left[1 + \frac{1}{2}(p^{-1} - q^{-1})\right](q+1) \\
&= pq^{-1} + 2p\left[(q+1) + \frac{1}{2}p^{-1}(q+1) - \frac{1}{2}(1 + q^{-1})\right] \\
&= pq^{-1} + p\left[2(q+1) + p^{-1}(q+1) - (1 + q^{-1})\right] \\
&= p\left[2(q+1) + p^{-1}(q+1) - 1\right] \\
&= p\left[2q + 1 + p^{-1}(q+1)\right] \\
&= (2pq + p + q + 1) \\
&= (2pq + 2) \\
&= 2(pq + 1) \\
&= 2\left[p(1-p) + 1\right] \\
&= 2\left[-\left(p - \frac{1}{2}\right)^2 + \frac{5}{4}\right]
\end{aligned} \tag{34}$$

Figure 3 shows Eq.(34) graphically.

If $\bar{X}'$ is in range $[2, \frac{5}{2}]$, the solution p of step 7 of 6.3.2 of NIST SP 800-90B[1] can be expressed by the following equation:

$$p = \frac{1}{2} + \sqrt{\frac{5}{4} - \frac{\bar{X}'}{2}}. \tag{35}$$

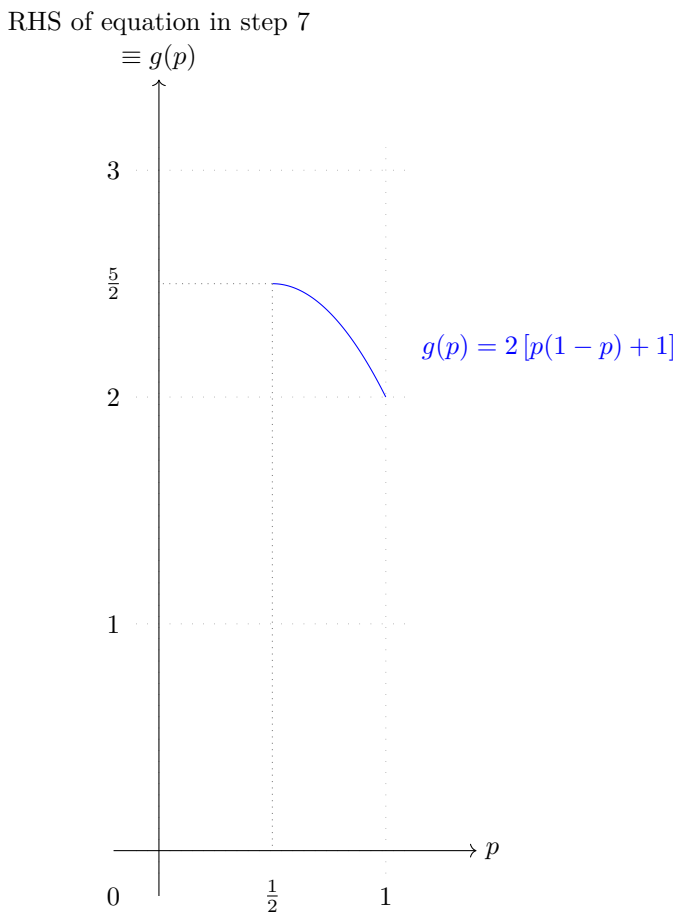

Figure 3 The right hand side of the equation in step 7 of 6.3.2 of NIST SP 800-90B

1.3 Notes for the compression estimate

As documented in [4], Eq.(36) should be replaced by Eq.37.

$$G(z) = \frac{1}{\nu} \sum_{t=d+1}^{L} \sum_{u=1}^{t} \log_2(u) F(z,t,u) \tag{36}$$

$$G(z) = \frac{1}{\nu} \sum_{t=d+1}^{\lfloor L/b \rfloor} \sum_{u=2}^{t} \log_2(u) F(z,t,u) \tag{37}$$

$F(z,t,u)$ in Eq.(37) can be expressed by the following equation:

$$F(z,t,u) = \begin{cases} z^2(1-z)^{u-1} & \text{if} \quad u < t \\ z(1-z)^{t-1} & \text{if} \quad u = t \end{cases} \tag{38}$$

We have to evaluate Eq.(37), but this expression requires relatively large computational complexity due to its nested summation. First Figure 4 shows parameters u, t, where nested summation must be taken to compute $G(z)$.

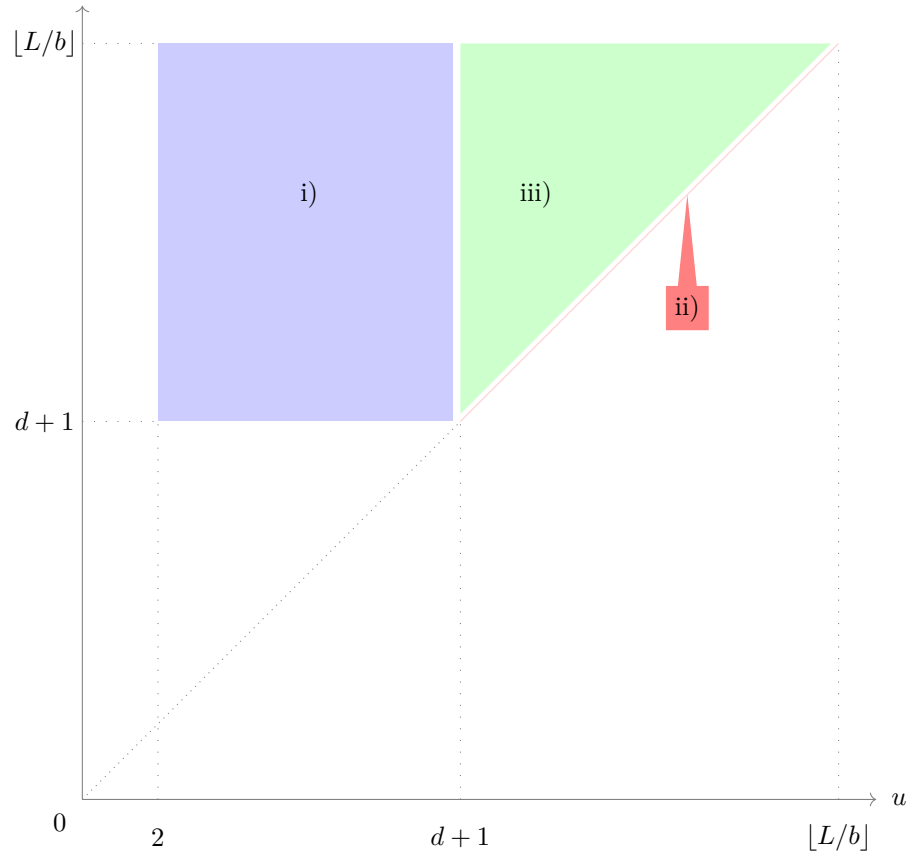

Figure 4 Parameters u, t, where nested summation must be taken

t-dependency of $F(z,t,u)$ must be considered when $u = t$. From the above, the parameters domain where summation must be taken can be divided into the following three groups:

i)

$$\begin{cases} 2 \le u < d+1 \\ d+1 \le t \le \lfloor L/b \rfloor \end{cases} \tag{39}$$

ii)

$$\begin{cases} (u,t) = (\tau,\tau) \\ d+1 \le \tau \le \lfloor L/b \rfloor \end{cases} \tag{40}$$

iii)

$$\begin{cases} d+1 \le u < t \\ d+1 < t \le \lfloor L/b \rfloor \end{cases} \tag{41}$$

For groups i) and iii), the summation with respect to t can be taken first. By using the relation $\nu = \lfloor L/b \rfloor - d$, Eq.(37) can be rewritten to as follows:

$$\begin{aligned} G(z) = & \sum_{u=2}^{d} \log_2(u) z^2 (1-z)^{u-1} \\ & + \frac{1}{\nu} \sum_{u=d+1}^{\lfloor L/b \rfloor} \log_2(u) z (1-z)^{u-1} \\ & + \frac{1}{\nu} \sum_{u=d+1}^{\lfloor L/b \rfloor - 1} (\lfloor L/b \rfloor - u) \log_2(u) z^2 (1-z)^{u-1} \end{aligned} \tag{42}$$

With this expression, the computational complexity can be decreased from $\mathcal{O}(\nu^2)$ to $\mathcal{O}(\nu)$.

1.4 Notes for t-Tuple estimate, LRS estimate, MultiMMC Prediction and LZ78Y prediction estimate

In the following groups of entropy estimate, we have to handle t-tuples (pairs, triples, etc.)

a) t-Tuple estimate (NIST SP 800-90B 6.3.5)
b) Longest Repeated Substring (LRS) estimate (NIST SP 800-90B 6.3.6)
c) Multi Most Common in Window Prediction estimate (NIST SP 800-90B 6.3.7)
d) The MultiMMC Prediction estimate (NIST SP 800-90B 6.3.9)
e) The LZ78Y Prediction estimate (NIST SP 800-90B 6.3.10)

In order to express t-tuples, bitset or multiprecision integer is used without using array.

1.5 Notes for Multi Most Common in Window prediction estimate

In step 3-a-i of 6.3.7 of NIST SP 800-90B, it is required to compute the mode in the previous window of w_j before s_i. As the order of w_j is 1000, the computational complexity is estimated to be about $\mathcal{O}(1000n)$, and expected to be time-consuming. We attempt to reduce time-complexity by preparing histograms of certain lengths in advance, and in step 3-a-i, we use the histograms that fit within the target window, and compute the unavailable parts on-demand basis.

2 Additional implementation notes for t-Tuple estimate, LRS estimate using Longest Common Prefix

2.1 An issue of naive implementation for t-Tuple estimate and LRS estimate

The naive implementation of t-Tuple estimate and LRS estimate consumes huge memory ≈ few giga-bytes, to store unique t-Tuple. Here, it is stated that Longest Common Prefix can be used for t-Tuple and LRS estimates, and naive algorithm is provided[5]. Note here that [6] provides good introduction for Longest Common Prefix and its calculating algorithm. Next optimized algorithms are provided to perform t-Tuple and LRS estimates[7]. However, it is unclear about how to derive the whole algorithms, except for "Count Aggregation".

2.2 Derivation of more intuitive algorithms for t-Tuple estimate and LRS estimate

So let us consider a more intuitive algorithm to perform t-Tuple and LRS estimates in this subsection.

Provided that Longest Common Prefix array $(LCP[1], \ldots, LCP[L])$ has been calculated from input $\boldsymbol{S}$, $LCP[j]$ changes with respect to index j, so let us consider the following three cases:

a) $LCP[j-1] < LCP[j]$ for $j \in [2, L]$,
b) $LCP[j-1] = LCP[j]$ for $j \in [2, L]$,
c) $LCP[j-1] > LCP[j]$ for $j \in [2, L]$.

We may call case a) as "climb up" case, case b) as "traverse" case, and case c) as "climb down" case.

For brevity, let us use η_j to denote $LCP[j]$, or simply η by omitting index j. Let us consider a run length of LCP array at index j, and denote λ_{η_j} as the run length.

Normally several minimum and maximum points can be found in LCP array, so let us consider the range from one local minimum to the next as a single "peak", and introduce an array $\tilde{q}_{\mathrm{work}}[\eta]$ representing run length value with respect to the "height" η (which is equal to a value in LCP array) in currently climbing peak. Considering that there are multiple peaks in LCP array usually, let us introduce an array $\tilde{q}_{\mathrm{master}}[\eta]$ storing the maximum run length for each η appeared in the LCP array.

For case a), let us store the previous run length value $\lambda_{\eta_{j-1}}$ to $\tilde{q}_{\mathrm{work}}[\eta_{j-1}]$, and assign new run length value λ_{η_j} as 1. Note that the part of algorithm is highlighted in light blue in Algorithms 1 and 2.

For case b), let us increment λ_{η_j} by 1. Note that the part of algorithm is highlighted in light green in Algorithms 1 and 2.

For case c), the run lengths of LCP values for $(LCP[j]+1, \ldots, LCP[j-1])$ can be determined by Algorithm 3. Note that the part of algorithm is highlighted in pale purple in Algorithms 1 and 2.

For case c) or "climb down" case, update $\tilde{q}_{\mathrm{master}}$ to store longer run lengths, and also run lengths for higher η values, by comparing $\tilde{q}_{\mathrm{work}}$ and $\tilde{q}_{\mathrm{master}}$.

Finally the required array $Q[i]$ where $i \in [1, t]$ can be calculated by the following relation:

$$Q[i] = \tilde{q}_{\mathrm{master}}[i] + 1. \tag{43}$$

Note here that Longest Common Prefix array is not suitable for calculating the number of occurrence of W-tuple when the number of occurrences is 1. However, W-tuple is not required where the number of occurrences of W-tuple in $\boldsymbol{S}$ is 1, as described in Step 2 of 6.3.6 of [1]. Therefore, we don't care about the applicability of Longest Common Prefix array to our algorithms.

2.3 Notes about the selection of Suffix Array algorithm and sorting algorithm

Algorithms 1 and 2 are nearly independent from the selection of Suffix Array algorithm and sorting algorithm, so readers can select those algorthms according to their preference. By paying attention to reliability and availability, the author select Suffix Array algorithm described in [6] and sort algorithm in standard C++ library.

2.4 Additional implementation notes for t-Tuple estimate using Longest Common Prefix

Algorithm 1 Rewritten steps 1 and 2 in t-Tuple Estimate

Input: input dataset $\boldsymbol{S} = (s_1, \ldots, S_L)$ where $s_i \in \boldsymbol{A} = \{x_1, \ldots, x_k\}$

Output: an integer array of i-tuple counts for $i \in \{1, 2, \ldots, t\}$

1: Compute the Longest Common Prefix array $LCP[j] (j \in \{1, 2, \ldots, L\})$ from $\boldsymbol{S}$.
2: $\eta_{\max} \leftarrow \mathbf{max}_{j \in \{1,2,\ldots,L\}} LCP[j]$

▷ $\eta_{\max}$ coincide with ν in Longest Repeated Substring (LRS) Estimate.

3: **for** $\eta \leftarrow 1, \mathbf{max}(\eta_{\max}, 1)$ **do**
4: $\quad \tilde{q}_{\text{master}}[\eta] \leftarrow 0$
5: $\quad \tilde{q}_{\text{work}}[\eta] \leftarrow 0$
6: **end for**
7: $\eta_{\text{prev}} \leftarrow LCP[1]$
8: $\lambda \leftarrow 1$
9: **for** $j \leftarrow 2, L$ **do**
10: $\quad \eta_{\text{current}} \leftarrow LCP[j]$
11: $\quad$ **if** $\eta_{\text{prev}} < \eta_{\text{current}}$ **then**
12: $\quad\quad$ **if** $0 < \eta_{\text{prev}}$ **then**
13: $\quad\quad\quad \tilde{q}_{\text{work}}[\eta_{\text{prev}}] \leftarrow \lambda$
14: $\quad\quad$ **end if**
15: $\quad\quad \lambda \leftarrow 1$
16: $\quad$ **else if** $\eta_{\text{prev}} = \eta_{\text{current}}$ **then**
17: $\quad\quad$ **if** $\eta_{\text{current}} = 0$ **then**

▷ This condition corresponds to consecutive 0-values in array LCP.

18: $\quad\quad\quad \lambda \leftarrow 1$

▷ consecutive 0-values in array LCP mean different 1-tuple.

19: $\quad\quad$ **else**
20: $\quad\quad\quad \lambda \leftarrow \lambda + 1$
21: $\quad\quad$ **end if**
22: $\quad$ **else**
23: $\quad\quad$ **AccumulateLambda**$(\tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{prev}}, \eta_{\text{current}}, \lambda)$
24: $\quad\quad$ **UpdateLambda**$(\lambda, \tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{current}})$
25: $\quad\quad$ **UpdateQTildeMaster**$(\tilde{\boldsymbol{q}}_{\text{master}}, \tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{prev}}, \eta_{\text{current}})$
26: $\quad\quad$ **ResetQTildeWork**$(\tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{prev}}, \eta_{\text{current}})$
27: $\quad$ **end if**
28: $\quad \eta_{\text{prev}} \leftarrow \eta_{\text{current}}$
29: **end for**
30: **AccumulateLambda**$(\tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{prev}}, 0, \lambda)$
31: **UpdateQTildeMaster**$(\tilde{\boldsymbol{q}}_{\text{master}}, \tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{prev}}, 0)$
32: $t \leftarrow \mathbf{max}\left(\{\eta \in \{1, \ldots, \eta_{\max}\} \mid \tilde{q}_{\text{master}}[\eta] \geq 34\}\right)$

▷ Find the largest t such that the number of occurrences of the most common t-tuple in $\boldsymbol{S}$ is at least 35. This can be rewritten as shown above using $\tilde{\boldsymbol{q}}_{\text{master}}$.

33: return **CalcQ**$(\tilde{\boldsymbol{q}}_{\text{master}}, t)$

2.5 Additional implementation notes for LRS estimate using Longest Common Prefix

Algorithm 2 Rewritten steps 1, 2 and 3 in Longest Repeated Substring (LRS) Estimate

Input: input dataset $\boldsymbol{S} = (s_1, \ldots, S_L)$ where $s_i \in \boldsymbol{A} = \{x_1, \ldots, x_k\}$

Output: string *status*, a real value $\hat{p}$

Compute the Longest Common Prefix array $LCP[j] (j \in \{1, 2, \ldots, L\})$ from $\boldsymbol{S}$.
$\eta_{\max} \leftarrow \mathbf{max}_{j \in \{1,2,\ldots,L\}} LCP[j]$
if $\eta_{\max} < 1$ **then**
    return ("ν not found", NaN)
▷ ν cannot be found, therefore estimate cannot be computed.
end if
$\nu \leftarrow \eta_{\max}$
▷ It is ensured that $\nu \geq 1$.
for $\eta \leftarrow 1, \nu$ **do**
    $\tilde{q}_{\text{master}}[\eta] \leftarrow 0$
    $\tilde{q}_{\text{work}}[\eta] \leftarrow 0$
    $\sigma[\eta] \leftarrow 0$
end for
$\eta_{\text{prev}} \leftarrow LCP[1]$
$\lambda \leftarrow 1$
for $j \leftarrow 2, L$ **do**
    $\eta_{\text{current}} \leftarrow LCP[j]$
    if $\eta_{\text{prev}} < \eta_{\text{current}}$ **then**
        if $0 < \eta_{\text{prev}}$ **then**
            $\tilde{q}_{\text{work}}[\eta_{\text{prev}}] \leftarrow \lambda$
        end if
        $\lambda \leftarrow 1$
    else if $\eta_{\text{prev}} = \eta_{\text{current}}$ **then**
        if $\eta_{\text{current}} = 0$ **then**
▷ This condition corresponds to consecutive 0-values in array LCP.
            $\lambda \leftarrow 1$
▷ consecutive 0-values in array LCP mean different 1-tuple.
        else
            $\lambda \leftarrow \lambda + 1$
        end if
    else
        AccumulateLambda$(\tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{prev}}, \eta_{\text{current}}, \lambda)$
        UpdateNumerator$(\boldsymbol{\sigma}, \tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{prev}}, \eta_{\text{current}})$
        UpdateLambda$(\lambda, \tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{current}})$
        UpdateQTildeMaster$(\tilde{\boldsymbol{q}}_{\text{master}}, \tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{prev}}, \eta_{\text{current}})$
        ResetQTildeWork$(\tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{prev}}, \eta_{\text{current}})$
    end if
    $\eta_{\text{prev}} \leftarrow \eta_{\text{current}}$
end for
AccumulateLambda$(\tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{prev}}, 0, \lambda)$
UpdateNumerator$(\boldsymbol{\sigma}, \tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{prev}}, 0)$
UpdateQTildeMaster$(\tilde{\boldsymbol{q}}_{\text{master}}, \tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{prev}}, 0)$
$u \leftarrow \mathbf{min}\left(\{\eta \in \{1, \ldots, \nu\} \mid \tilde{q}_{\text{master}}[\eta] < 34\}\right)$
▷ Find the smallest u such that the number of occurrences of the most common u-tuple in $\boldsymbol{S}$ is less than 35. This can be rewritten as shown above using $\tilde{\boldsymbol{q}}_{\text{master}}$.
$\hat{p} \leftarrow$ **CalcPHat**$(\boldsymbol{\sigma}, u, \nu)$
▷ Compute the estimated average collision probability per string symbol as $P_{\max,W} = (P_W)^{\frac{1}{W}}$. Let $\hat{p} = \max\left(P_{\max,u}, \ldots, P_{\max,\nu}\right)$.
return ("Success", $\hat{p}$)

2.6 Auxiliary routines

Algorithm 3 AccumulateLambda

procedure AccumulateLambda($\tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{prev}}, \eta_{\text{current}}, \lambda$)
  $\tilde{q}_{\text{work}}[\eta_{\text{prev}}] \leftarrow \lambda$
  for $\eta \leftarrow (\eta_{\text{prev}} - 1)$ **downto max**$(\eta_{\text{current}}, 1)$ **do**
    $\tilde{q}_{\text{work}}[\eta] \leftarrow \tilde{q}_{\text{work}}[\eta] + \tilde{q}_{\text{work}}[\eta + 1]$
    ▷ sum up λ and update the array $\tilde{q}$
  end for
end procedure

Algorithm 4 UpdateLambda

procedure UpdateLambda($\lambda, \tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{current}}$)
  if $\eta_{\text{current}} = 0$ **then**
    $\lambda \leftarrow 1$
  else
    $\lambda \leftarrow (\tilde{q}_{\text{work}}[\eta_{\text{current}}] + 1)$
  end if
end procedure

Algorithm 5 UpdateQTildeMaster

procedure UpdateQTildeMaster($\tilde{\boldsymbol{q}}_{\text{master}}, \tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{prev}}, \eta_{\text{current}}$)
  for $\eta \leftarrow \eta_{\text{prev}}$ **downto** $(\eta_{\text{current}} + 1)$ **do**
    if $\tilde{q}_{\text{master}}[\eta] < \tilde{q}_{\text{work}}[\eta]$ **then**
      $\tilde{q}_{\text{master}}[\eta] \leftarrow \tilde{q}_{\text{work}}[\eta]$
      ▷ update η-tuple count
    end if
  end for
end procedure

Algorithm 6 ResetQTildeWork

procedure ResetQTildeWork($\tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{prev}}, \eta_{\text{current}}$)
  for $\eta \leftarrow \eta_{\text{prev}}$ **downto max**$(\eta_{\text{current}}, 1)$ **do**
    $\tilde{q}_{\text{work}}[\eta] \leftarrow 0$
    ▷ reset to zero for future computation
  end for
end procedure

Algorithm 7 CalcQ

function CalcQ($\tilde{\boldsymbol{q}}_{\text{master}}, t$)
  for $\eta \leftarrow 1, t$ **do**
    $Q[\eta] \leftarrow \tilde{q}_{\text{master}}[\eta] + 1$
  end for
  return $\boldsymbol{Q} = (Q[1], \ldots, Q[t])$
end function

Algorithm 8 UpdateNumerator

procedure UpdateNumerator($\boldsymbol{\sigma}, \tilde{\boldsymbol{q}}_{\text{work}}, \eta_{\text{prev}}, \eta_{\text{current}}$)
  for $\eta \leftarrow \eta_{\text{prev}}$ **downto** $(\eta_{\text{current}} + 1)$ **do**
    ▷ Here, $\tilde{q}_{\text{work}}[\eta_{\text{current}}]$ is a snapshot at a given point in time, therefore, updating $\sigma[\eta]$ where $\eta > \eta_{\text{current}}$ makes sense.
    if $\tilde{q}_{\text{work}}[\eta] + 1 \geq 2$ **then**
      $\sigma[\eta] \leftarrow \sigma[\eta] + \binom{\tilde{q}_{\text{work}}[\eta] + 1}{2}$
    end if
  end for
end procedure

Algorithm 9 CalcPHat

function CalcPHat($\boldsymbol{\sigma}, u, \nu$)
  for $W \leftarrow u, \nu$ **do**
    $P[W] \leftarrow \frac{\sigma[W]}{\binom{L-W+1}{2}}$
  end for
  $\hat{p} \leftarrow$ **max** $\left((P[u])^{\frac{1}{u}}, \ldots, (P[\nu])^{\frac{1}{\nu}} \right)$
  return $\hat{p}$
end function

3 References

[1] Meltem Sönmez Turan, Elaine Barker, John Kelsey, Kerry A. McKay, Mary L. Baish, Mike Boyle *Recommendation for the Entropy Sources Used for Random Bit Generation*, NIST Special Publication 800-90B, Jan. 2018

[2] M. Peter and W. Schindler, *A Proposal for Functionality Classes for Random Number Generators* Version 2.35 - DRAFT, September 2, 2022. https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Certification/Interpretations/AIS_31_Functionality_classes_for_random_number_generators_e.pdf?__blob=publicationFile&v=7

[3] Franck W. J. Oliver, Daniel W. Lozier, Ronald F. Boisvert, Charles W. Clark, *NIST Handbook of Mathematical Functions*, National Institute of Standards and Technology, 2010

[4] G. Sakurai, *Proposed list of corrections for NIST SP 800-90B 6.3 Estimators*, Dec. 2022 https://github.com/g-g-sakura/AnotherEntropyEstimationTool/blob/main/documentation/ProposedListOfCorrections_SP800-90B.pdf

[5] Joshua E. Hill, *Algorithms for t-tuple and LRS Estimates*, June 11, 2018 https://www.untruth.org/~josh/sp80090b/NIST.SP.800-90B%20implementation%20comments%2020191219.pdf

[6] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein, *Introduction to Algorithms (fourth edition)*, The MIT Press. https://mitpress.mit.edu/9780262046305/introduction-to-algorithms/

[7] Aaron Kaufer, *Further Improvements for SP 800-90B Tuple Counts*, April 10, 2019 https://www.untruth.org/~josh/sp80090b/Kaufer%20Further%20Improvements%20for%20SP%20800-90B%20Tuple%20Counts.pdf