Efficient Bernoulli Probability Distribution Estimation for Arithmetic Coding

Megh Shah

CS20BTECH11032

Abstract

1. This paper presents a novel method for Bernoulli probability distribution estimation, based on low pass filtering with varying dominant pole.
2. This solution uses integer-only arithmetic, (and right and left shift operations) without multiplication or division, thus reducing the required resources of IoT devices.

Maximum Likelihood Estimation (MLE)

1. Maximum likelihood estimation is used to estimate an unknown parameter θ for which the likelihood function $L(\theta|x)$ is largest, where x is the recorded observation.
2. Hence we mean to find the value of the parameter for which the observation(s) recorded is (are) the most likely.

ML for Binomial Distribution

For $x \sim \text{Bin}(n, p)$, known observation x, known n and unknown p:
The likelihood function is:

$$L(p|x) = \frac{n!}{x!(n-x)!} p^x (1-p)^{n-x} \tag{1}$$

We know $L(p|x)$ is a continuous, differentiable function.
$L(p|x) = 0$ for $p = 0, 1$ and for $0 \leq p \leq 1$, $L(p|x) > 0$
Hence atleast 1 maxima exists for $\hat{p}$ where $0 < \hat{p} < 1$
Taking the natural logarithm on both sides and differentiating, we get:

$$\frac{1}{L(p|x)} \frac{d(L(p|x))}{dp} = 0 + x \frac{d(\ln p)}{dp} + (n-x) \frac{d(\ln(1-p))}{dp} \tag{2}$$

ML for Binomial Distribution (Cont.)

Setting $\frac{d(L(p|x))}{dp} = 0$

$$0 = x\frac{d(\ln p)}{dp} + (n-x)\frac{d(\ln(1-p))}{dp} \tag{3}$$

$$0 = \frac{x}{p} + \frac{x-n}{1-p} \tag{4}$$

$$p(n-x) = x(1-p) \tag{5}$$

$$p = \frac{x}{n} \tag{6}$$

Since the derivative is 0 at only one point, it is the maxima and $\hat{p} = x/n$

Considering a Symbol Sequence for Data Compression

Consider a sequence of symbols y_i from the symbol set $S = \{S_0, S_1\}$, whose first k symbols are known.

Assuming that the symbol sequence is generated by a Bernoulli process with time-invariant probability distribution $p(y)$:

The probability of symbol y occurring (using MLE): $\hat{p}_k(y) = x_y/k$

Where x_y is the number of occurrences of the symbol y and k is the number of symbols sampled. This can be rewritten as:

$$\hat{p}_k(y) = \left(\sum_{i=1}^{k} eq(y, y_i) \right) / k \tag{7}$$

where the $\hat{p}_k(y)$ is the probability distribution estimated from the first k symbols, and eq is the symbol equality function defined as

$$eq(a, b) = \begin{cases} 1 & \text{if } a = b \\ 0 & \text{if } a \neq b \end{cases} \tag{8}$$

Considering a Symbol Sequence for Data Compression

Now, consider an auxiliary binary sequence, u_i, formed from the symbol sequence as follows:

$$u_i = eq(y_i, S_0) \tag{9}$$

The binary sequence u_i can be represented as a sum of two stochastic signals, which is the expected value of the sample u_i (probability of the sample y_i being equal to the S_0) and the zero-mean noise n_i

$$u_i = p_i(S_0) + n_i \tag{10}$$

The noise variance is a function of the probability $p_i(S_0)$ as in

$$E(n_i^2) - (E(n_i))^2 = p_i(S_0)(1 - p_i(S_0)) - 0 \tag{11}$$

Thus, the worst-case noise power occurs when the $p_i(S_0)$ is 0.5

Considering a Symbol Sequence for Data Compression

Since the symbol probability distribution $p_i(y)$ is slowly changing, the power spectral density of the signal $p_i(S_0)$ is concentrated in low frequency range ($p_i(S_0)$ is a narrow band signal).
So the signal to noise ratio can be improved by filtering u_i with a low pass filter.
If the bandwidth is too narrow, then the filter output will be slow to converge to the true probability estimates.
And if the the bandwidth is too broad, then there will be a lot of noise.
Thus, the filter would have higher bandwidth in the beginning of the sequence y_i, to quickly identify initial conditions, but a reduced bandwidth later, to reject the noise.

Low Pass Filter

Noise Power Spectral Density

The noise power spectral density (PSD) specifies the average power of noise at different frequencies. If we apply this noise to an LTI system, the transfer function of the system will determine the output average power at different frequencies.

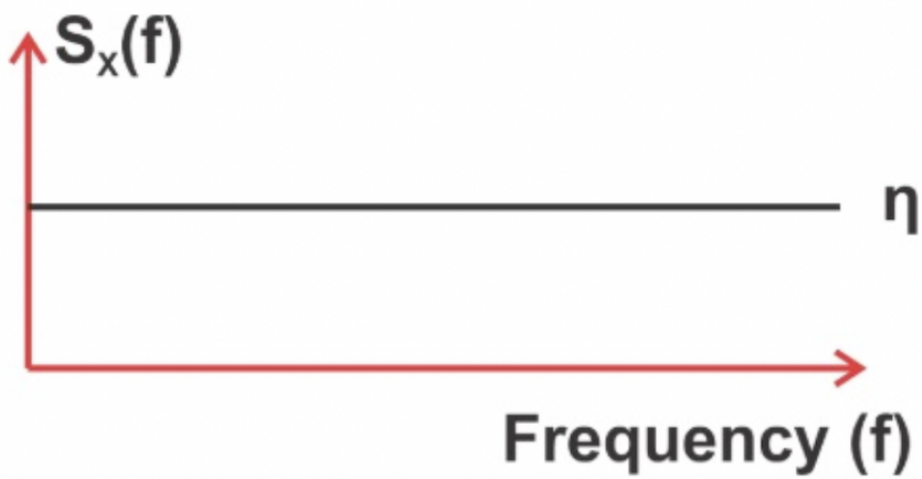

Figure: Figure 1 shows the spectrum of a hypothetical noise source that exhibits the same average power at all frequencies, i.e. $S_X(f) = \eta$ where η is a constant

Low Pass Filter (Cont.)

In case if the system is an ideal low-pass filter with a DC gain of 1, after applying the filter, all of the noise frequency components in the stop-band will be suppressed and the frequency components in the pass-band will be unaffected.

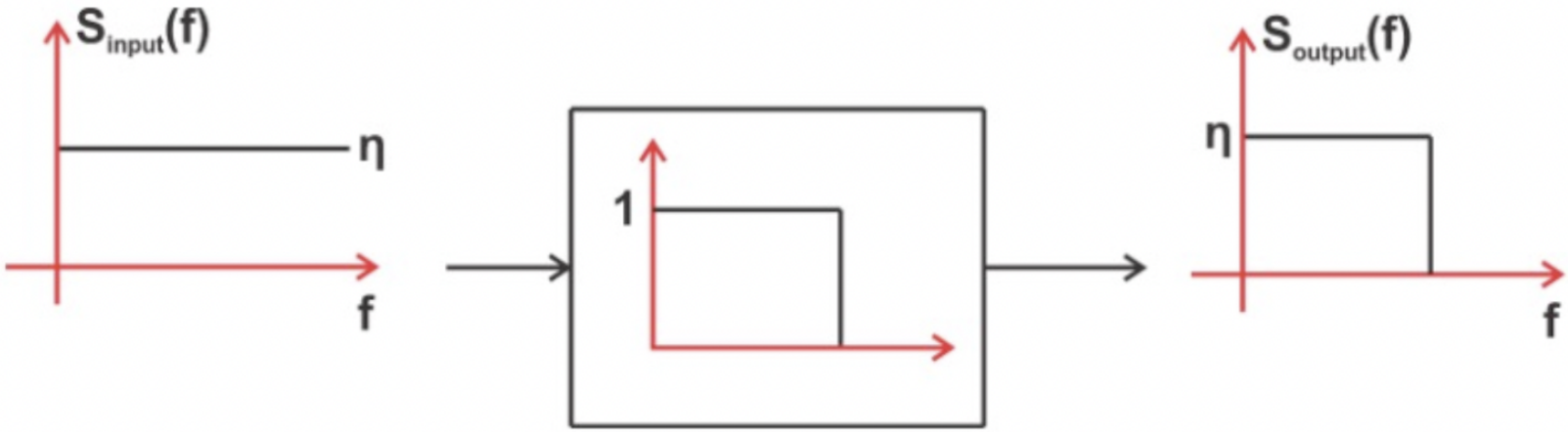

Figure: Figure 2 shows the application of an ideal low-pass filter on the previous hypothetical case

System Zeros, Poles and Transfer Functions

Transfer functions, in general, of an electronic or control system component are a mathematical function which theoretically model the device's output for each possible input.
The transfer function is a rational function in the complex variable $s = \sigma + j\omega$:

$$H(s) = \frac{b_m s^m + b_{m-1} s^{m-1} + ... + b_1 s + b_0}{a_n s^n + a_{n-1} s^{n-1} + ... + a_1 s + a_0} \tag{12}$$

Transfer functions are often written in terms of the factors of the numerator and denominator polynomials:

$$H(s) = \frac{N(s)}{D(s)} = \frac{b_m}{a_n} \frac{(s - z_1)(s - z_2)...(s - z_{m-1})(s - z_m)}{(s - p_1)(s - p_2)...(s - p_{n-1})(s - p_n)} \tag{13}$$

System Zeros, Poles and Transfer Functions (Cont.)

Now, the system zeros are simply defined to be roots of the numerator polynomial (z_i) and the system poles are defined to be the roots of the denominator polynomial (p_i).

The poles and zeros are properties of the transfer function, and along with the gain constant b_m/a_n, they completely characterize the differential equation, and provide a total description of the system.

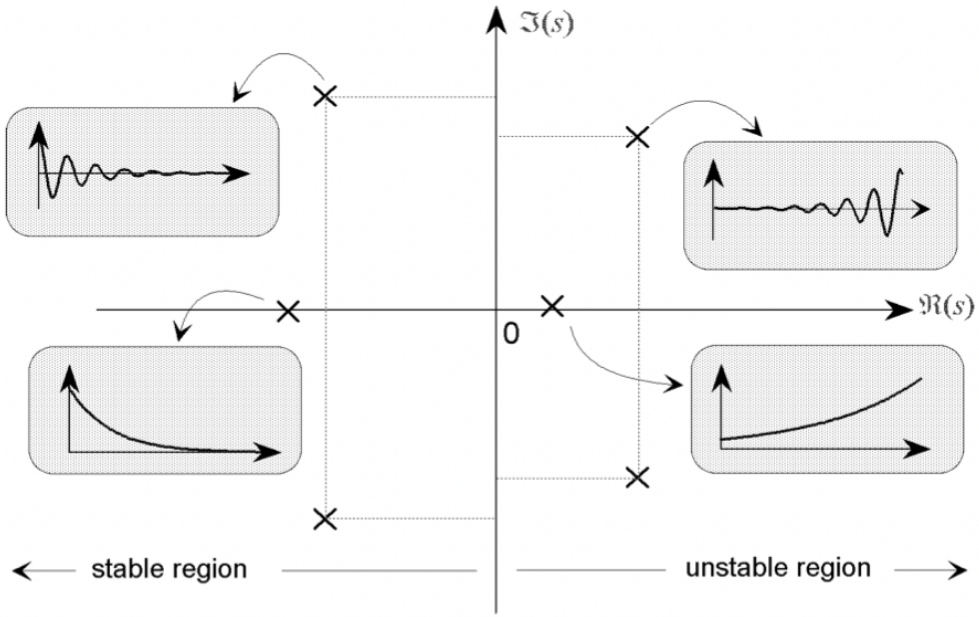

Figure: The various locations of poles in the Complex plane and their corresponding homogeneous responses

Maximum Likelihood Filter

For maximal filter length N (at maximum, previous N symbols are used for probability estimation):

$$h_k = \begin{cases} \left(\sum_{i=1}^{k} T(eq(x_i, S_0)) \right) / k & \text{if } k < N \\ \left(\sum_{i=k-N+1}^{k} T(eq(x_i, S_0)) \right) / N & \text{if } k \geq N \end{cases} \tag{14}$$

(15)

And to remove the division by integers (a costly operation), we use the right shift operator:

$$w = \lfloor \log_2(\min(k, N)) \rfloor \tag{16}$$

$$h_k = 2^{-w} \sum_{i=k-2^w+1}^{k} T(eq(x_i, S_0)) \tag{17}$$