Preliminary Definitions

Probability Measure Space - Measure space (X, μ) with μ(X) = 1.

Ergodic Transformation - Let (X, A, μ) be a probability space. Suppose that T:X$\rightarrow$X is μ-invariant. Then T is said to be ergodic if E$\in$A satisfies T^{-1}E = E iff μ(E) = 0 or 1.

Entropy of a partition P - Let (X,μ) be a probability space and P = {E_1, ... , E_k} a partition. Then the entropy of P is

$$H(P) = \sum ip_i Log(\tfrac{1}{p_i} = -\sum ip_i Log(p_i)$$

where $p_i = \mu(E_i)$ and we will use 2 as the base of the logarithm.

Entropy of P with respect to measure preserving transformation T - For partition P let $T^{-j}P = \{T^{-j}E_1, ..., T^{-}jE_k\}$ and $P_n = \bigvee_{i=0}^{n} T^{-i}P$. Then the entropy

$$h(T, P) = \lim_{n\rightarrow\infty} \tfrac{1}{n} H(P_n)$$

The entropy of T is then defined as

$$h(T) = \sup_P h(T, P)$$

A major result that is key to proving most of what I will discuss today follows.

Theorem 1 (The first return time formula/Ornstein-Weiss): Let T be the shift transformation on $\prod_{i=1}^{\infty} S$ with shift invariant probability measure μ. Suppose T is ergodic for x = (x_1, x_2, ...) $\in$ X define

$$R'_n(x) = \min\{\ j\geq n\text{: } x_1 = x_{j+1}, \ \dots\ , x_n = x_{j+n}\ \}$$

Then for almost every x, $\lim_{n\rightarrow\infty} \frac{Log[R'_n(x)]}{n}$ = entropy h(T)

This fact was studied by Wyner and Ziv in relation to data compression. They proved convergence in probability. Ornstein and Weiss then proved convergence is with probability 1, as in it converges point wise a.e.

This theorem lets us make entropy estimates by looking at a typical sample sequence. Real world binary sequences are finite, but we can treat them as part of an infinite sequence. As long as n is of appropriate size compared to the full sequence this works well. While the book didn't specifically mention this, I imagine that in real world applications you have to watch out for special

strings. For instance, a file might start with a specific string to give information like data type that would never naturally appear again. To deal with this, my guess is you just ignore these bits when applying the test. There is likely more complications to watch out for. The next lemma gives some relation between R_n and R'_n

Lemma (properties of R_n and R'_n): Let T be an ergodic shift transformation on a shift space X with finite alphabet.

(i) If $R_n(x) < n$, then there exists k, $\frac{n}{3} \leq$ k ¡ n such that $R'_k(x) =$ k

(ii) If T has positive entropy, then for almost every x there exists N = N(x) such that $R'_n(x) = R_n(x)$ for $n \geq N$

Proof: (i) if $R_n(x) =$ k for some k¡n, then $x_1, ..., x_k = x_{k+1}, ..., x_{2k}$ and $R'_k(x) = k$. If $k \geq \frac{n}{3}$ then done, else if $k < \frac{n}{3}$ then $x_1...x_{n+k} = x_1...x_k x_1...x_k x_1...x_l$ for some $l \leq k$. Hence $R'_{mk}(x) = mk$ for some m such that $\frac{n}{3} \leq mk < n$.

(ii) Let E = $\{x \in X : \limsup_{n\to\infty} R'_n(x) - R_n(x) > 0\}$. If $x \in E$, then $R'_n(x) > R_n(x)$ for infinitely many n. Thus $R_n(x) < n$ for infinitely man n. By part (i) we have $R'_k(x) = k$ for infinitely many k and $\liminf_{k\to\infty} \frac{Log(R'_k(x)}{k} = 0$ which by thee Ornstein-Weiss formula implies E has measure 0 as T has positive entropy by assumption meaning that $\lim_{n\to\infty} \frac{Log(R'_n(x)}{n} > 0$ for x almost everywhere. ■

The above theorem tells us that for big n we can essentially treat R_n and R'_n as being similar.

Theorem 3: For an ergodic shift on finitely many symbols, $\lim_{n\to\infty} \frac{Log(R_n(x)}{n} =$ *Entropy* for almost every x.

Proof: If entropy is positive we have by our lemma that $R'_n(x) = R_n(x)$ for some $n \geq N$ a.e. If entropy is zero we have $\limsup_{n\to\infty} \frac{Log(R_n(x))}{n} \leq \limsup_{n\to\infty} \frac{Log(R'_n(x))}{n} = \lim_{n\to\infty} \frac{Log(R'_n(x))}{n} = 0$. ■

$\mathbf{L^p}$ − convergence

Shannon-Mcmillan-Breiman Theorem - Let $p \geq 1$. Consider an ergodic shift space (X, μ) and its partition P_n into the n-blocks determined by the first n symbols. Define

$E_n(x) = \sum_{B\in P_n} \frac{-1}{n} Log(\mu(B))\chi_B(x)$

Then E_n converges to entropy h for almost every x in L^p.

Theorem: Let $p \geq 1$ and let (X, μ) be an ergodic shift space. Then $\frac{Log(R_n)}{n}$ converges to h in L^p.

proof: Recall that Fatou's lemma says that for a sequence of measurable functions $f_n \geq 0$ we have

$\int_X \liminf_{n\to\infty} f_n d\mu \leq \liminf_{n\to\infty} \int_X f_n d\mu.$

Ornsein-Weiss combined with fatou's lemma gives us that

$h^p \leq \liminf_{n\to\infty} \int_\chi \frac{Log(R_n)}{n})^p d\mu$

claim: $(Ln(x))^p$ is concave for sufficiently large x. For p = 1 its obvious. If $p \geq 2$ then $\frac{d^2}{dx^2}(ln(x))^p = p\frac{(p-1)ln(x)^{p-2} - ln(x)^{p-1}}{x^2} < 0$ when $ln(x) > p-1$. Note that $n \leq nR_n(x)$ for almost every x. Recall Jensen's inequality where if (X, μ) is a probability measure space,$f : X \to R$ a measurable function, and $\phi : R \to R$ is convex and $\phi \circ f$ is integrable then

$\phi(\int_X f d\mu) \leq \int_X \phi(f(x)) d\mu$

Now we apply Jensen's inequality for $(Log(x))^p$ which is concave on $[n, \infty)$ for large n by previous statement. We have

$\limsup_{n\to\infty} \int_X (\frac{Log(nR_n)}{n})^p d\mu =$

$\limsup_{n\to\infty} \sum_{B\in P_n} \int_X (\frac{Log(nR_n)}{n})^p d\mu_B \leq$

$\limsup_{n\to\infty} \frac{1}{n^p} \sum_{B\in P_n} Log(\int_X (\frac{nR_n}{n}) d\mu_B)^p =$

$\limsup_{n\to\infty} \frac{1}{n^p} \sum_{B\in P_n} \mu(B)(Log(n) - Log(\mu(B))^p =$

$\limsup_{n\to\infty} \sum_{B\in P_n} \mu(B)(\frac{Log(n)}{n} - \frac{Log(\mu(B)}{n})^p =$

$\limsup_{n\to\infty} \int_X E_n^p d\mu = h^p$

Hence for $p \geq 1$ we have $lim_{n\to\infty} \int (\frac{(Log(R_n)}{n})^p d\mu = h^p$. Now for every even integer k we see that

$\lim_{n\to\infty} \int (\frac{Log(R_n)}{n} - h)^k d\mu =$

$\lim_{n\to\infty} \sum_{j=0}^k C(k,j)(-h)^{k-j} \int (\frac{Log(R_n)}{n})^j d\mu =$

$\lim_{n\to\infty} \sum_{j=0}^k C(k,j)(-h)^{k-j} h^j d\mu = 0$ ■

The Non-overlapping First Return Time

Def: The nth nonoverlapping first return time $\tilde{R}_n$ is defined by $\tilde{R}_n(x) = min(j \geq 1 : x_1, x_2, ...x_n = x_{jn+1}, ..., x_{(j+1)n})$

Note this is similar to before but we break up X into blocks of length n and look for j s.t. the jth block is the same as the first. As data elements on a computer tend to be represented by a block of several bits (say for instance each pixel in an image file might be several bytes with one byte the position and others the different color intensities) this formulation might be more useful in some cases. It also might be faster to compute in general as the computer can check say 32-bits in a single clock cycle to see if it matches a 32-bit value.

Def: For $0 < r < 1$ define $v(r) = r \sum_1^\infty (1-r)^{i-1} Log(i)$

Before proving a general result lets examine the logarithm of $\tilde{R}_n$ for the $(\frac{1}{2}, \frac{1}{2})$-Bernoulli shift. The values in parenthesis tell us the probability of an element appearing, in this case we'll have 2 elements both with 50% chance of happening.

For any n-block B this means the probability of the block B occurring is $Pr(B) = 2^{-n}$. This gives us that $E[Log(\tilde{R}_n)] = v(r)]$ with $r = 2^{-n}$. To see this, we have the probability of j = 1 is 2^{-n}, j = 2 is 2^{-n} times the probability of not being j = 1 or $2^{-n} * (1 - 2^{-n})$ and so on. With this in mind we have

$\lim_{r \to 0+}[v(r) + Log(r)] = \lim_{s \to 1-}[v(1-s) + Log(1-s)]$

$= \lim_{s \to 1-}[(1-s) \sum_{i=1}^\infty s^{i-1} Log(i) + Log(1-s)]$

$= \lim_{s \to 1-}[\sum_{i=1}^\infty s^{i-1} Log(i) - \sum_{i=1}^\infty s^i Log(i) - \frac{1}{ln(2)} \sum_{i=1}^\infty \frac{1}{i} s^i]$

$= \lim_{s \to 1-} \sum_i^{infty} s^i [Log(i+1) - Log(i) - \frac{1}{ln2} \frac{1}{i}]$

$= \frac{1}{ln2} \sum_{i=1}^\infty (ln \frac{i+1}{i} - \frac{1}{i}) = \frac{-\gamma}{ln2} = -0.832746...$

Where $\gamma = \lim_{n \to \infty} (\sum i = 1^n \frac{1}{i} - ln(n))$ is called the Euler's constant. So $E[Log(\tilde{R}_n)]$ for the $(\frac{1}{2}, \frac{1}{2})$-Bernoulli shift is approximately equal to $n - \frac{\gamma}{ln2}$ for sufficiently large n. This intuitively makes some sense. A block of size n has a probability of 2^{-n} so we expect to have to pick random blocks around 2^n times before we see it, or the log of the expectation to be about n. It is hard to see any relation to the entropy from this specific case. Lets look now at proving something similar to the above for Bernoulli shifts in general where a relation will become clear.

Theorem: Take p, q such that $0 < p, q < 1$ and $p + q = 1$. Consider the (p,q)-Bernoulli shift with entropy h. For sufficiently large n

$E[Log(\tilde{R}_n)] \approx nh - \frac{\gamma}{ln2}$.

Proof: For the general Bernoulli shift, $h = -pLog(p) - qLog(q)$ Recall that

$(p+q)^n = \sum_{k=0}^{n} C(n,k)p^k q^{n-k}$ and differentiating with respect to p

$n(p+q)^{n-1} = \sum_{k=1}^{n} C(n,k)kp^{k-1}q^{n-k}$.

Now we have that

$E[Log(\tilde{R}_n)] = \sum_{|B|=n} Pr(B)E[Log(\tilde{R}_n)|B]$

$= \sum_{k=0}^{n} C(n,k)p^k q^{n-k} \sum_{i=1}^{infty}(Log(i))(1-p^k q^{n-k})^{i-1}p^k q^{n-k}$

$= \sum_{k=0}^{n} C(n,k)p^k q^{n-k} v(p^k q^{n-k})$

$\approx \sum_{k=0}^{n} C(n,k)p^k q^{n-k}[-Log(p^k q^{n-k}) - \frac{\gamma}{ln2}]$

$= -\sum_{k=0}^{n} C(n,k)p^k q^{n-k} kLog(p) - \sum_{k=0}^{n} C(n,k)p^k q^{n-k}(n-k)Log(q) - \frac{\gamma}{ln2}$

$= -\sum_{k=0}^{n} C(n,k)p^k q^{n-k} kLog(p) - \sum_{k=0}^{n} C(n,k)p^{n-k} q^k kLog(q) - \frac{\gamma}{ln2}$

$= nh - \frac{\gamma}{ln2}$. ■

Now we have our relation to entropy. The reason the entropy didn't seem appear in the $(\frac{1}{2}, \frac{1}{2})$-Bernoulli shift is because in that case $h = 1$. Note also that as expectation is linear,

$E[\frac{Log(\tilde{R}_n}{n}] = h$

For large n giving us a result that is similar to how R_n behaved in the limit.

Lempel-Ziv Coding

LZ-coding is a simple to implement loss-less compression algorithm. To get an idea of how the algorithm works it is probably best to just see its use on a short binary sequence.

1001110010101010

We create a distinct parsing, as in break the above sequence into phrases such that no two are the same. For our specific case, we read left to write and place a comma after a subset of the sequence that has not previously appeared. Our above sequence becomes

1, 0, 01, 11, 00, 10, 101, 010

Then for each phrase above we assign an address being its position. Looking at everything except the right most bit we use the address to tell the reader that this sequence is the same as the sequence found earlier. Then the second digit will be the rightmost bit. This finishes the encoding an looks as follows

(0, 1) (0, 0) (2, 1) (1, 1), (2, 0) (1, 0) (6, 1) (3, 0)

The sequence needs to be reasonably long for any compression to take place. If C is the number of phrases we need approximately Log(C) bits to describe each address plus one bit for the last bit for each phrase in the coding. This gives approximately C(Log(C)+1) total bits after applying the algorithm. A theorem states

$\lim_{N\to\infty} \frac{C(Log(C)+1)}{N} = h$

Where h is the entropy. Experimenting with the matlab code there definitely appears to be a relation, but I don't have a proof of this fact. the above ratio on the left is called the compression ratio. If this ratio is greater than one the algorithm will end up needed more bits to describe the sequence than in the original sequence, so we should only use LZ if this ratio is less than one.

Works Cited

Choe, Geon. **Computational Ergodic Theory**. Verlag Berlin Heidelberg. Springer 2005.