Information Theory
Degree in Data Science and Engineering
Lesson 3: Information of data sources

Jordi Quer, Josep Vidal

Mathematics Department, Signal Theory and Communications Department
{jordi.quer, josep.vidal}@upc.edu

2019/20 - Q1

Sources of data and information

Most natural signals or symbols generated by a source convey information in a very dilute form: large amount of data contain a small amount of information.

There are two main reasons for that:

- Data that are close to each other tend to have similar values (e.g. pixels in an image, pixels in consecutive images in a video sequence, temporal samples in an audio signal), or are related to each other (e.g. letters in English, video recordings of the same scene at closely spaced cameras, samples of stereo audio recordings).
- Not all values generated by our source of data are equally frequent. We know already that those less frequent carry more information.

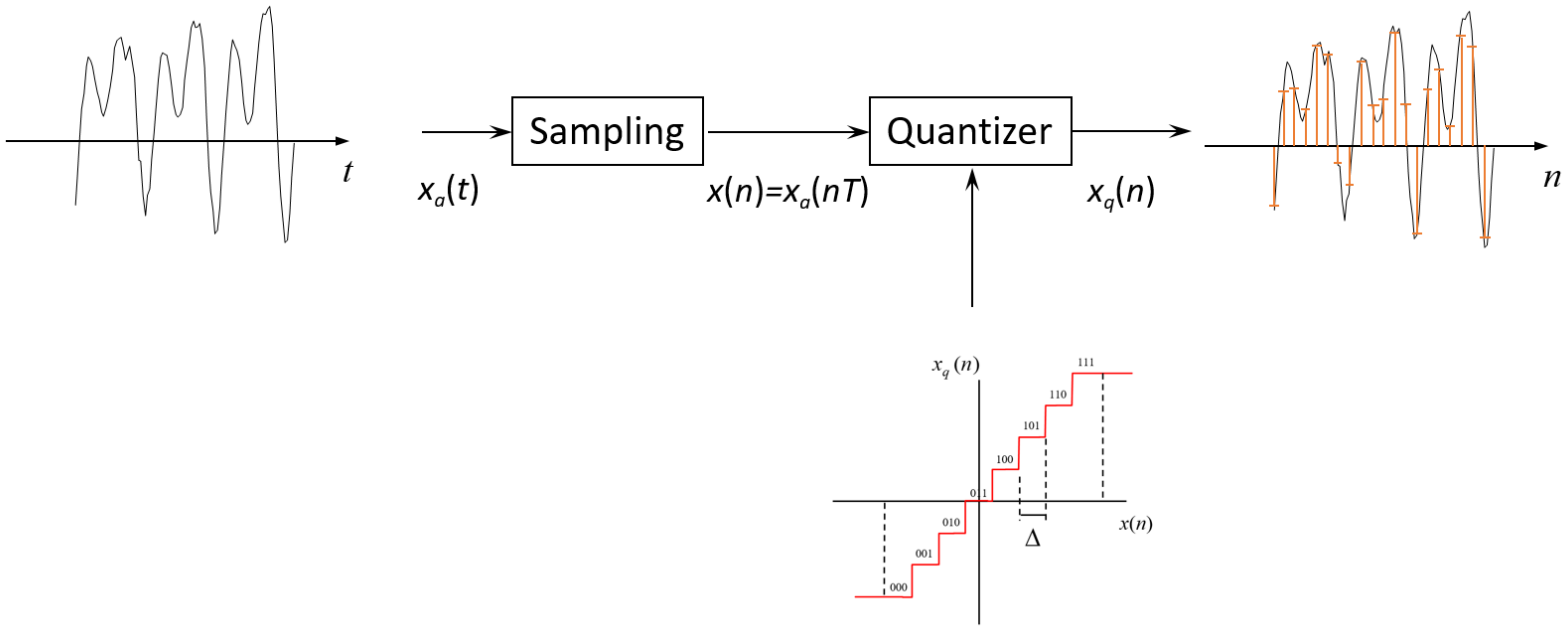
Examples of redundant sources

- Neighbour pixels of an image.

87	89	101	106	118	130	142	155
85	91	101	105	116	129	135	149
86	92	96	105	112	128	131	144
92	88	102	101	116	129	135	147
88	94	94	98	113	122	130	139
88	95	98	97	113	119	133	141
92	99	98	106	107	118	135	145
89	95	98	107	104	112	130	144

Examples of redundant sources

- Samples obtained from the digitalization of an audio signal.



- Predictability of letters in English increases with the context:

$$\text{"}Oh\ my\ God,\ the\ vulcano\ is\ eru\_\_\_\text{"}$$

Examples of redundant sources

- Frequency of letters in English is far from being uniform in normal text (values have been estimated from *The Frequently Asked Questions Manual for Linux*).

i	a_i	p_i
1	a	0.0575
2	b	0.0128
3	c	0.0263
4	d	0.0285
5	e	0.0913
6	f	0.0173
7	g	0.0133
8	h	0.0313
9	i	0.0599
10	j	0.0006
11	k	0.0084
12	l	0.0335
13	m	0.0235
14	n	0.0596
15	o	0.0689
16	p	0.0192
17	q	0.0008
18	r	0.0508
19	s	0.0567
20	t	0.0706
21	u	0.0334
22	v	0.0069
23	w	0.0119
24	x	0.0073
25	y	0.0164
26	z	0.0007
27	–	0.1928

Purpose of the chapter

- Can we find short and yet reversible descriptions of a sequence of random observations $x_1 x_2 \dots x_n$?
- How short can this description be? How much can we compress data?

This will be highly relevant for the purposes of storage and communication of sequences of symbols.

Alphabets

Values generated by a source of random data X belong to an *alphabet* (see also *alphabet*) which is a finite set $\mathcal{X} = \{\mathtt{a}_1, \mathtt{a}_2, \ldots, \mathtt{a}_q\}$ of $|\mathcal{X}| = q$ elements $\mathtt{a}_i$, called letters or symbols. For example:

- Many natural languages are written using *variants* of the ISO basic *Latin alphabet* of 26 letters:

$$\mathtt{a},\ \mathtt{b},\ \mathtt{c},\ \mathtt{d},\ \mathtt{e},\ \mathtt{f},\ \mathtt{g},\ \mathtt{h},\ \mathtt{i},\ \mathtt{j},\ \mathtt{k},\ \mathtt{l},\ \mathtt{m},$$
$$\mathtt{n},\ \mathtt{o},\ \mathtt{p},\ \mathtt{q},\ \mathtt{r},\ \mathtt{s},\ \mathtt{t},\ \mathtt{u},\ \mathtt{v},\ \mathtt{w},\ \mathtt{x},\ \mathtt{y},\ \mathtt{z}.$$

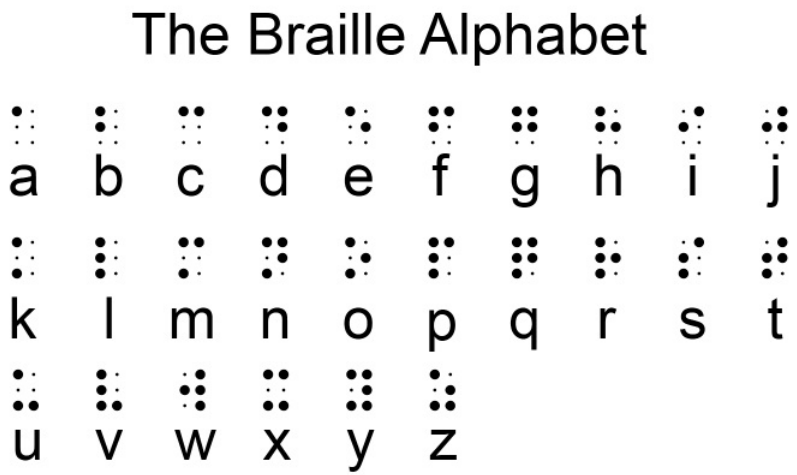
- The *Greek alphabet* of 24 letters:

$$\alpha,\ \beta,\ \gamma,\ \delta,\ \epsilon,\ \zeta,\ \eta,\ \theta,\ \iota,\ \kappa,\ \lambda,\ \mu,$$
$$\nu,\ \xi,\ o,\ \pi,\ \rho,\ \mathcal{X},\ \tau,\ \upsilon,\ \phi,\ \chi,\ \psi,\ \omega.$$

- The *DNA alphabet* of 4 letters $\mathcal{X} = \{A, C, T, G\}$ representing the four nucleotides adenine, cytosine, thymine, guanine, used in *genetics* to write the genome.

Alphabets

- The *Braille alphabet* of 64 letters:



- A digital image is written in the alphabet of pixels, whose letters are d-bit numbers, with d the image *color bit depth*.
- A digital sound is written in the alphabet of wave samples, whose letters are d-bit numbers, with d the *audio bit depth*.
- Of course, the most important to us is the binary alphabet $\mathcal{X} = \{0, 1\}$ consisting of symbols 0 and 1.

Blocks and strings

A sequence of letters of $\mathcal{X}$ is called word, block, string, chain, text, message, etc. depending on the context.

The name word is mostly used for sequences of short fixed length, or for the sequences belonging to a certain particular set (a code).

$\mathcal{X}^n$ is the set of the q^n words (or blocks) of n letters:

$$\mathcal{X}^n = \{x^n = \mathtt{a}_1\mathtt{a}_2\ldots\mathtt{a}_n : \mathtt{a}_i \in \mathcal{X}\}.$$

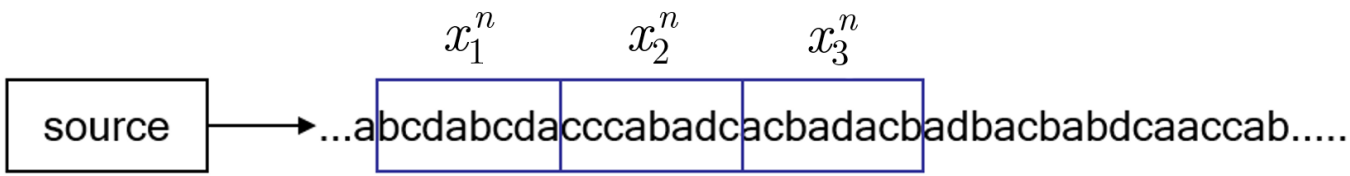
$\mathcal{X}^*$ is the infinite set of strings of arbitrary length:

$$\mathcal{X}^* = \{x^* = \mathtt{a}_1\mathtt{a}_2\ldots\mathtt{a}_n : \mathtt{a}_i \in \mathcal{X}, n \geq 0\} = \bigcup_{n\geq 0} \mathcal{X}^n.$$

We denote $\ell(x^n) = n$ the length of the string (number of letters). The empty string ϵ with $\ell(\epsilon) = 0$ is considered an element of $\mathcal{X}^*$.

Codes

A source code $\mathcal{C}_n$ is a mapping that re-labels an n-length sequence of symbols belonging to an alphabet, into a codeword of symbols belonging possibly to another alphabet. The value of n is chosen when designing the code.



$$\mathcal{C}_n \; : \; \mathcal{X}^n \to \mathcal{B}^*$$
$$c = \mathcal{C}_n(x^n)$$

If the length of the codewords $\ell(c)$ is the same for all x_i^n, we have a fixed-length code. Otherwise, the code is said a variable-length code.

Codes

Some definitions for a code...

- Extension code is the concatenation of codewords.
- Codebook is the set of codewords corresponding to the set of source-words,
$$\mathcal{C}_n = \{c : c = \mathcal{C}_n(x^n), x^n \in \mathcal{X}^n\}$$
- Non-singularity: no two different source words get mapped to the same codeword,
$$c = \mathcal{C}_n(x_1^n) = \mathcal{C}_n(x_2^n) \implies x_1^n = x_2^n$$
that is, $\mathcal{C}_n(x_1^n)$ is an *injective* mapping. This ensures that the encoding is reversible, and zero-error decoding is possible. All codes will be injective.
- Rate of the code: the ratio of the encoded sequence length to the source sequence length.

Efficiency

Coding is more efficient if words of $n > 1$ symbols are encoded. Let us take a fair 6-sides dice, and name X the random variable associated to the outcome of a throw.

$\mathcal{X} = \{1, 2, 3, 4, 5, 6\}$ and $H(X) = \log 6 = 2.58$ bits/outcome.

- If we want to encode a sequence of outcomes using a binary alphabet $\{0, 1\}$, $\mathcal{C}_1 = \{000, 001, 010, 011, 100, 101, 110, 111\}$ are the 8 codewords needed (we recognise that two of them will not be used). The rate is 3 binary digits/outcome.

  Efficiency of the code: $\eta = \frac{H(X)}{3} = 0.86$ bits/binary digit

- We can do better if we encode words of 4 outcomes in each codeword. There are now $|\mathcal{X}| = 6^4 = 1296$ possible source words for which we need $|\mathcal{C}_4| = 2^{11}$ codewords of 11 binary digits. The rate is $11/4 = 2.75$ binary digits/outcome.

  Efficiency of the code: $\eta = \frac{H(X)}{2.75} = 0.938$ bits/binary digit

These are fixed-length codes.

Efficiency

In general, efficiency can also be defined for q-ary codewords, as

$$\text{Efficiency of the code: } \eta = \frac{H_q(X)}{q-\text{ary symbols per outcome}}$$

where $H_q(X) = -\sum_{i=1}^{q} p(x_i) \log_q p(x_i)$

Teaser...

A sequence of letters is the observation of a sequence of random variables (that is, an ergodic process) governed by a probability distribution. The probability distribution provides all the necessary information about the achievable bounds for data compression.

In this lesson, we prove that it is possible to encode an n-length sequence of random symbols with an efficiency that approaches 1 with high accuracy, when n is large enough. We will assume first that $X_1X_2...X_n$ are independent and identically distributed (i.i.d.) random variables, and will drop the assumption at the end.

Stochastic process

A *stochastic process* is an infinite sequence $\mathbf{X} = X_1 X_2 X_3 \ldots$ of random variables, each taking values in the same set $\mathcal{X}$. A finite set of n random variables will be denoted by $X^n = X_1 X_2 \ldots X_n$.

Examples of sources generating stochastic processes:

- a *language model* for English, Catalan, etc. with an alphabet $\mathcal{X} = \{\mathsf{a}, \mathsf{b}, \ldots, \mathsf{z}, \text{-}, !, ?, (,), \ldots, ;, :\}$ that includes letters, space and punctuation characters,
- a sequence of n dice throws,
- the n samples of a sampled recording of a particular phoneme pronounced by many speakers,

such that a given length-n sequence $x_1 x_2 \ldots x_n \in \mathcal{X}^n$ is associated to a certain probability:

$$p(x_1, x_2, \ldots, x_n) = \Pr(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n)$$

satisfying probability rules (joint, marginal, conditional). Of course, the random variables X_i may not be mutually independent.

Stationary stochastic process

A stochastic process is said to be stationary if the joint distribution is invariant to shifts in the index

$$\Pr(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) \\ = \Pr(X_{1+l} = x_1, X_{2+l} = x_2, \ldots, X_{n+l} = x_n)$$

for every shift l and for all values $x_i \in \mathcal{X}$.

As a consequence, the statistical magnitudes that can be computed on each X_i do not depend on the index:

$$\mathbb{E}[g(X_1)] = \mathbb{E}[g(X_2)] = \cdots = \mathbb{E}[g(X_n)]$$

where g is a function that applies on the value taken by a random variable X_i and hence its output is itself a random variable.

For example, take $g(x) = x^2$. If the process is stationary, the second order moment does not depend on the index and we can write

$$\mathbb{E}[X_i^2] = \mathbb{E}[X^2] \quad \forall i$$

Stationary stochastic process

A process is called i.i.d. (independent identically distributed) if each random variable is independent of the rest:

$$\Pr(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = \prod_{i=1}^{n} \Pr(X_i = x_i)$$

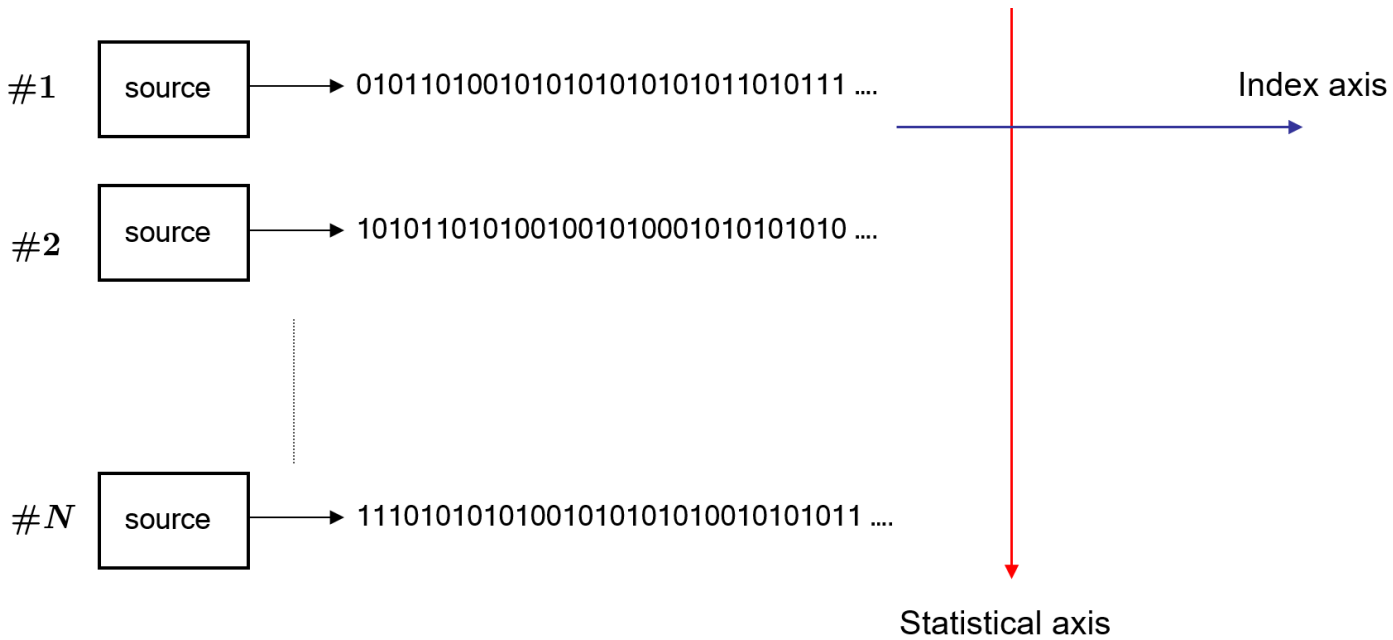
Or in a simpler notation:

$$p(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} p(x_i) \quad \text{with} \quad x_i \in \mathcal{X}$$

If random variables are associated to observations in time, then an i.i.d. process has no memory of past or future.

Ergodic processes

Imagine the following experiment: observe N times the output of a stochastic process consisting of n random variables $X_1 X_2 \ldots X_n$. Each observation of the n values taken by these random variables is called a realization of the process.



Stochastic processes can be defined at will. For example, we could randomly take N English books and observe the sequence of the first n letters. Or take just one book, open at random N pages and observe the sequence of the first n letters.

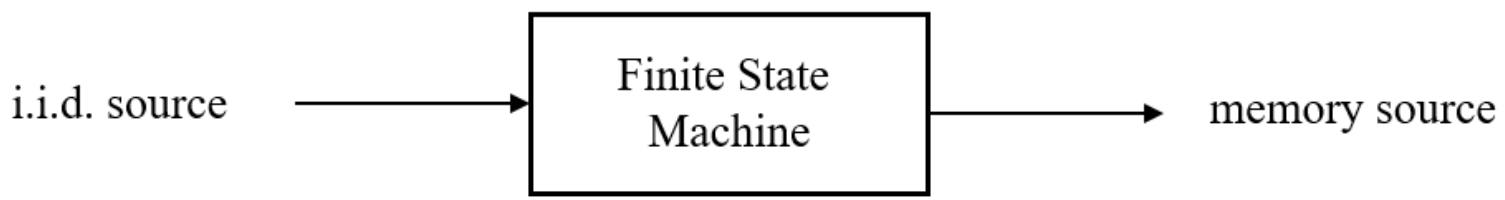
Ergodic processes

Assume the process is stationary. We define the process as *ergodic* if statistical magnitudes can be evaluated from temporal averages done on any single realization:

$$\mathbb{E}[g(X)] = \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} g(x_k)$$

Otherwise said, if the process is ergodic, we do not need all possible realizations to infer statistical information of the process.

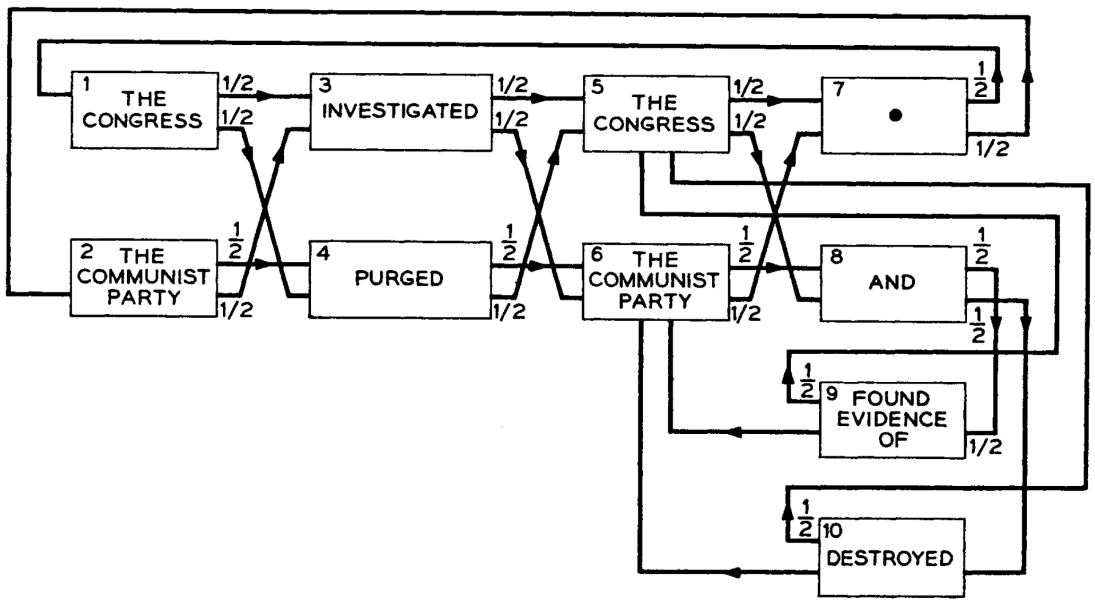
In general, ergodic processes have memory when the random variables $X_1, X_2, \ldots, X_n$ are not independent. A memory process can be generated from an i.i.d. process as



where an FSM generates outputs depending on the current and past values of the input. Only ergodic processes will be considered in the sequel.

A finite state machine producing English text

Assume the values adopted by X are a set of English words. This FSM can generate a number of distinct ergodic output sequences if the inputs are random binary values that select outputs of the states:



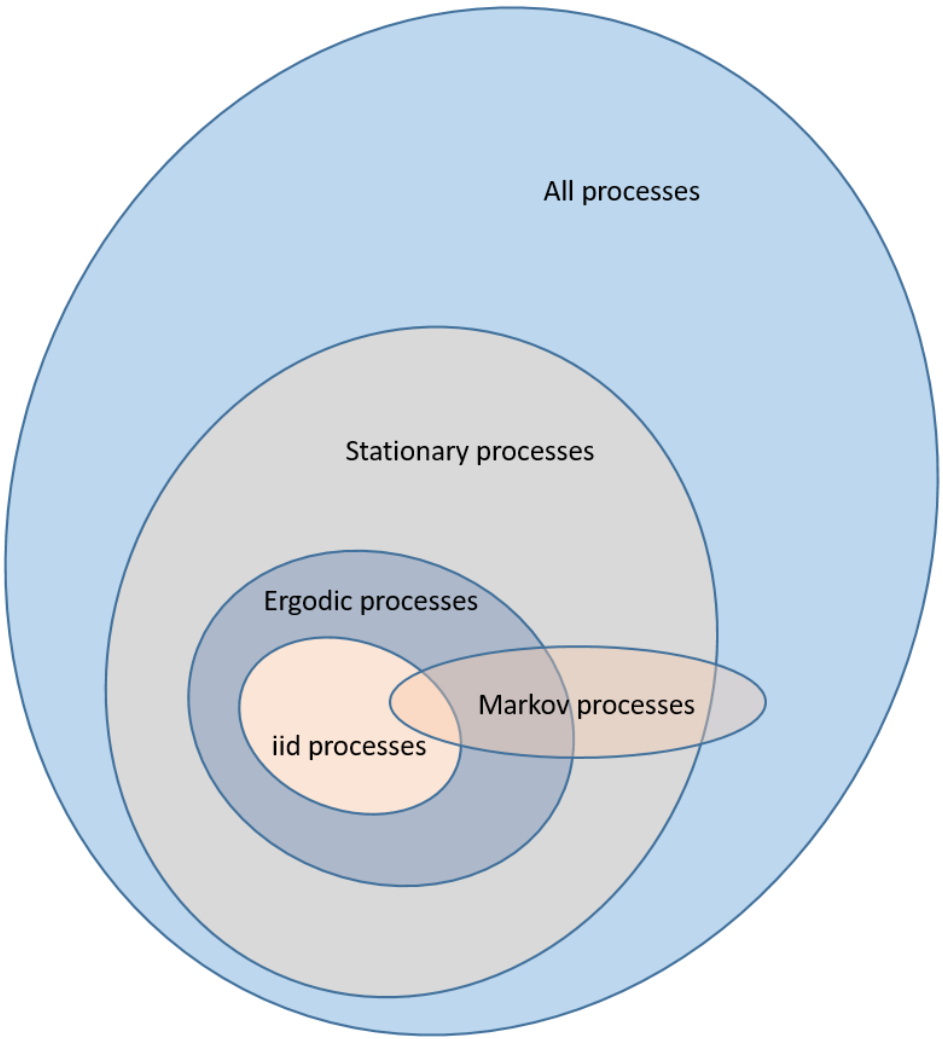
Possible output sequences:

THE COMMUNIST PARTY INVESTIGATED THE CONGRESS.

THE CONGRESS INVESTIGATED THE COMMUNIST PARTY AND FOUND EVIDENDE OF THE CONGRESS DESTROYED THE COMMUNIST PARTY.

THE CONGRESS PURGED THE CONGRESS.

A taxonomy of processes



Markov process

A *Markov process* is an ergodic stochastic process in which past has no influence on the future, given the present. In general, in an order k Markov process, each variable depends only on the previous k:

$$\begin{aligned}\Pr(X_{n+k} = x_{n+k} | X_{n+k-1} = x_{n+k-1}, \ldots, X_1 = x_1) \\ = \Pr(X_{n+k} = x_{n+k} | X_{n+k-1} = x_{n+k-1}, \ldots, X_n = x_n).\end{aligned}$$

The simplest example of a discrete-time Markov process is an order 1 Markov process:

$$\begin{aligned}\Pr(X_{n+1} = x_{n+1} | X_n = x_n, X_{n-1} = x_{n-1}, \ldots, X_1 = x_1) \\ = \Pr(X_{n+1} = x_{n+1} | X_n = x_n).\end{aligned}$$

If the transition probabilities do not depend on n, the Markov process is called invariant or homogeneous:

$$\Pr(X_{n+1} = x | X_n = y) = \Pr(X_{m+1} = x | X_m = y) \qquad \forall n, m$$

Markov process

For an order 1 Markov process, the joint pdf is given by

$$p(x_1, x_2, \ldots, x_n) = p(x_1)p(x_2|x_1)p(x_3|x_2)\ldots p(x_n|x_{n-1})$$

The probability distribution of states at time $n+1$ is given by

$$p(x_{n+1}) = \sum_{x_n \in \mathcal{X}} p(x_n)p(x_{n+1}|x_n)$$

which can be written in matrix form as

$$\mathbf{p}(n+1) = \mathbf{p}(n)\mathbf{P}, \quad n \geq 0$$

where

- $[\mathbf{P}]_{i,j} = \Pr(X_{n+1} = x_j | X_n = x_i)$, are the elements of the transition matrix $\mathbf{P}$,
- $[\mathbf{p}(n)]_j = \Pr(X_n = x_j)$ are the elements of the row vector $\mathbf{p}(n)$ containing the probabilities of all $|\mathcal{X}|$ states at time n.

Markov process

A Markov process is said invariant if matrix $\mathbf{P}$ does not depend on n.
A Markov process is stationary if $\mathbf{p}(n)$ does not depend on n, so $\mathbf{p}(0)$ must be an eigenvector of $\mathbf{P}$.

If all elements of $\mathbf{P}$ are positive, the *Perron-Frobenius theorem* applies to conclude that:

1. The largest left-eigenvalue of $\mathbf{P}$ is simple and its value is 1.
2. The entries of the associated eigenvector are real and positive.
3. All other left-eigenvalues are smaller in modulus (may be complex).

In this case, it turns out that

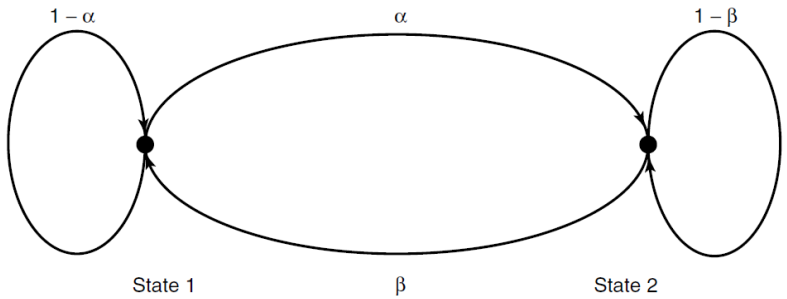
$$\lim_{n\to\infty} \mathbf{p}(n) = \lim_{n\to\infty} \mathbf{p}(0)\mathbf{P}^n = \mathbf{p}$$

and hence $\mathbf{p}$ is an eigenvector of $\mathbf{P}$ associated to the unit left-eigenvalue.

For non-negative $\mathbf{P}$ other results apply (not considered here).

Exercise

A two-states Markov parocess, where $\mathcal{X} = \{\text{State 1}, \text{State 2}\}$, can be represented graphically as



where $[\mathbf{P}]_{i,j} = \Pr(X_{n+1} = \text{State } j | X_n = \text{State } i)$. The transition matrix is given by

$$\mathbf{P} = \begin{bmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{bmatrix}$$

- What are the asymptotic stationary probabilities $\mathbf{p}$ as $n \to \infty$, regardless the value of $\mathbf{p}(0)$?
- What is $\mathbf{p}(0)$ for the Markov process to be stationary? That is, $\mathbf{p}(n)$ does not depend on n.
- What are the values of α and β for an i.i.d. Markov process?

Some preliminaries...

We will exploit some properties of stochastic processes to achieve efficient coding. A few well known theorems are needed.

Lemma (Markov's inequality)

For any non-negative random variable X and any $\alpha > 0$

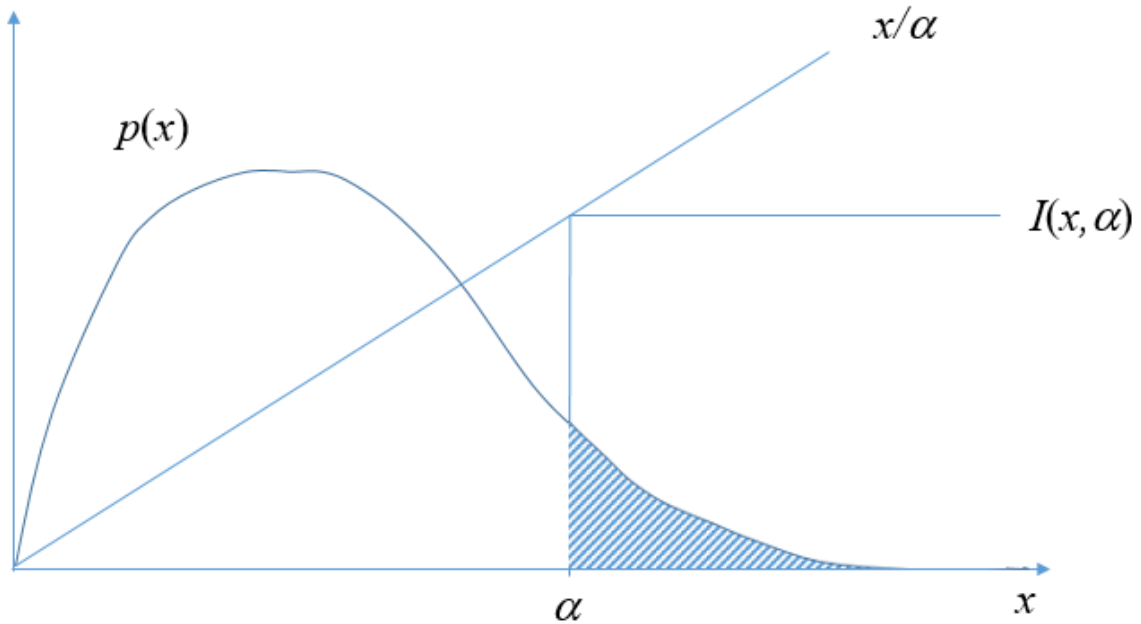
$$\Pr(X \geq \alpha) \leq \frac{\mathbb{E}[X]}{\alpha}$$

Proof. Define the indicator function as $I(x, \alpha) = \begin{cases} 1 & x \geq \alpha \\ 0 & x < \alpha \end{cases}$

$$\Pr(X \geq \alpha) = \mathbb{E}[I(x, \alpha)] \leq \mathbb{E}[X/\alpha] = \frac{\mathbb{E}[X]}{\alpha} \quad \square$$

This inequality can be used to bound the tails of a distribution.

Some preliminaries...



Some preliminaries...

Lemma (Chebyshev's inequality)

$$\Pr\left(|X - \mathbb{E}[X]| \geq \beta\right) \leq \frac{\sigma_x^2}{\beta^2}$$

Proof. Consider

$$\Pr\left(|X - \mathbb{E}[X]| \geq \beta\right) = \Pr\left(|X - \mathbb{E}[X]|^2 \geq \beta^2\right) \leq \frac{\sigma_x^2}{\beta^2}$$

where the *Markov's inequality* has been applied in the last inequality. □

Some preliminaries...

The average of n random variables can be made arbitrarily close to the mean by increasing n.

Theorem (The weak law of large numbers)

Consider a sequence of i.i.d. random variables X_i, and $\hat{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$, then

$$\Pr\left(\left|\hat{X}_n - \mathbb{E}[X_i]\right| \geq \epsilon\right) \leq \frac{\sigma^2}{n\epsilon^2}$$

where σ^2 is the variance of X_i.

Proof. Apply *Chebyshev's inequality*. □

It is said that $\hat{X}_n$ converges in probability to $\mathbb{E}[X_i]$:

$$\hat{X}_n \to \mathbb{E}[X_i]$$

The asymptotic equipartition property (AEP)

The key theorem is...

Theorem

If $X_1, X_2, \ldots, X_n$ are i.i.d. random variables of pdf $p(X)$, then

$$-\frac{1}{n}\log p(X_1, X_2, \ldots, X_n) \to \mathbb{E}[-\log p(X_1, X_2, \ldots, X_n)] = H(X)$$

that is

$$\lim_{n\to\infty} \Pr\left(\left|-\frac{1}{n}\log p(X_1, X_2, \ldots, X_n) - H(X)\right| \geq \epsilon\right) = 0$$

Proof. Apply the weak law of large numbers to the random variable

$$-\frac{1}{n}\log p(X_1, X_2, \ldots, X_n) = -\frac{1}{n}\sum_{i=1}^{n}\log p(X_i). \quad \square$$

Properties of typical sequences

Typical sequences are those whose log of the probability approaches $nH(X)$ within a small value ϵ.

The set of typical sequences will be called $\mathcal{A}_\epsilon^{(n)}$ and it is included in $\mathcal{X}^n$. It is the set of observations of $X^n = X_1X_2...X_n$ such that

$$2^{-n(H(X)+\epsilon)} \leq p(x_1, x_2, ..., x_n) \leq 2^{-n(H(X)-\epsilon)}.$$

Therefore, the probability of all typical sequences is nearly the same.

Two relevant properties of the set are proved next.

Properties of typical sequences

What is the probability mass of typical sequences?

Theorem (3.1)

For a sufficiently large value of n, $\Pr\left(\mathcal{A}_\epsilon^{(n)}\right) > 1-\epsilon$

Proof. Apply the weak law of large numbers. □

That is, the set contains most of the probability.
How many typical sequences are there?

Theorem (3.2)

For any value of n, $\left|\mathcal{A}_\epsilon^{(n)}\right| \leq 2^{n(H(X)+\epsilon)}$

For a sufficiently large value of n, $\left|\mathcal{A}_\epsilon^{(n)}\right| \geq (1-\epsilon)2^{n(H(X)-\epsilon)}$

Properties of typical sequences

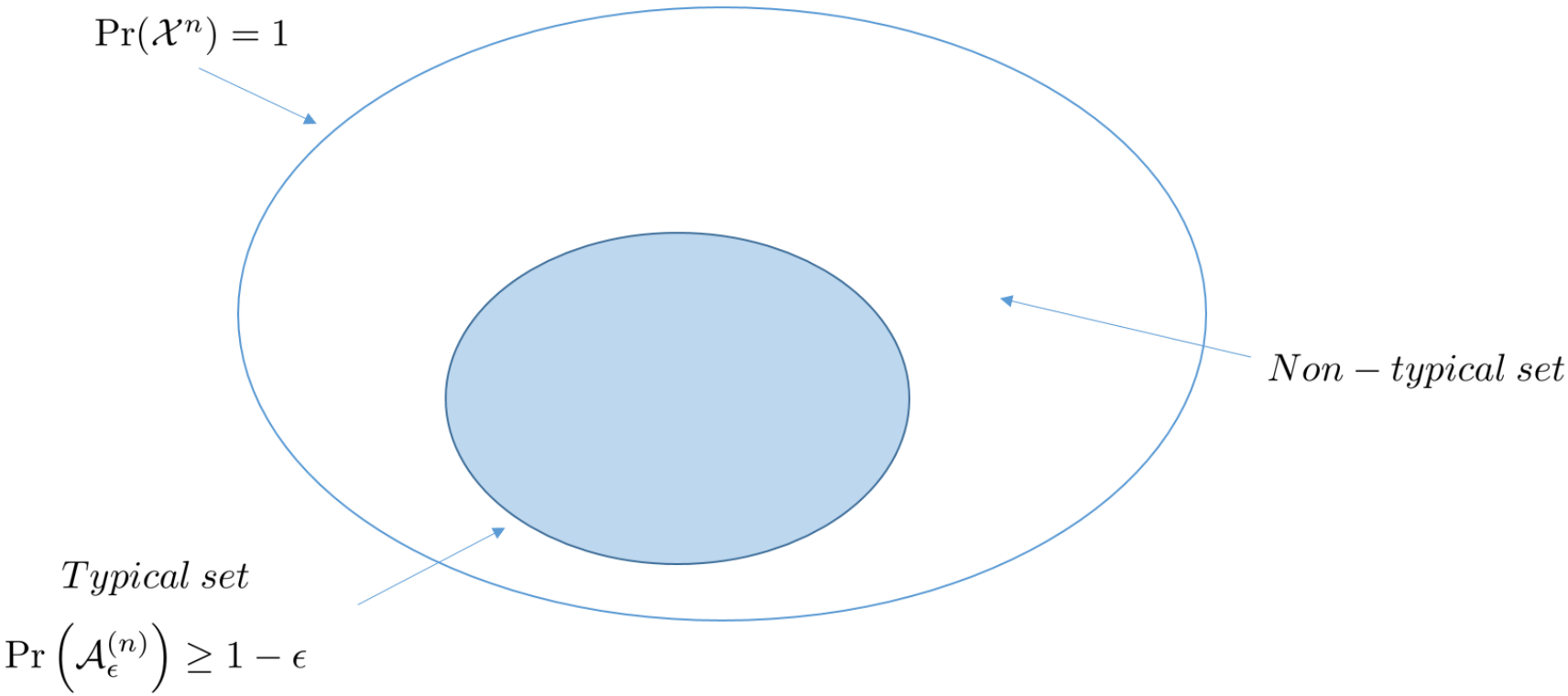
Proof. For the upper bound,

$$1 = \sum_{x^n \in \mathcal{X}^n} p(x^n) \geq \sum_{x^n \in \mathcal{A}_\epsilon^{(n)}} p(x^n) \geq \sum_{x^n \in \mathcal{A}_\epsilon^{(n)}} 2^{-n(H(X)+\epsilon)}$$
$$= 2^{-n(H(X)+\epsilon)} \left| \mathcal{A}_\epsilon^{(n)} \right|$$

For the lower bound, take theorem 3.1, so that

$$1 - \epsilon < \Pr\left(\mathcal{A}_\epsilon^{(n)}\right) \leq \sum_{x^n \in \mathcal{A}_\epsilon^{(n)}} 2^{-n(H(X)-\epsilon)} = 2^{-n(H(X)-\epsilon)} \left| \mathcal{A}_\epsilon^{(n)} \right| . \square$$

The set may be small, its size depends on the entropy of X.


$\Pr(\mathcal{X}^n) = 1$
$Non - typical\ set$
$Typical\ set$
$\Pr\left(\mathcal{A}_\epsilon^{(n)}\right) \geq 1 - \epsilon$

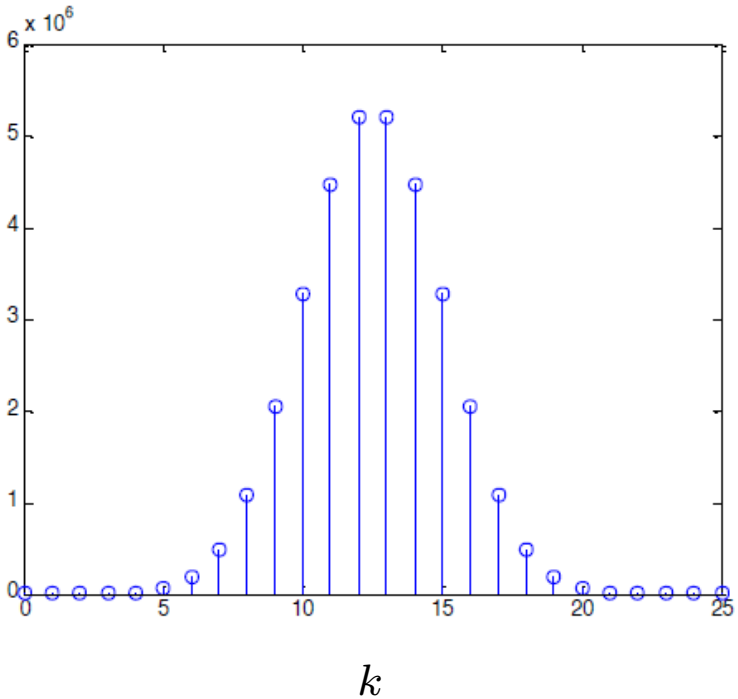
Example: the AEP in a Bernoulli process

Take a sequence of i.i.d. observations of an unfair coin. We'll check the properties of $\mathcal{A}_\epsilon^{(n)}$. The sequence forms a stationary *Bernoulli process*, whose density function is $\Pr(X=1)=p$, $\Pr(X=0)=1-p$.

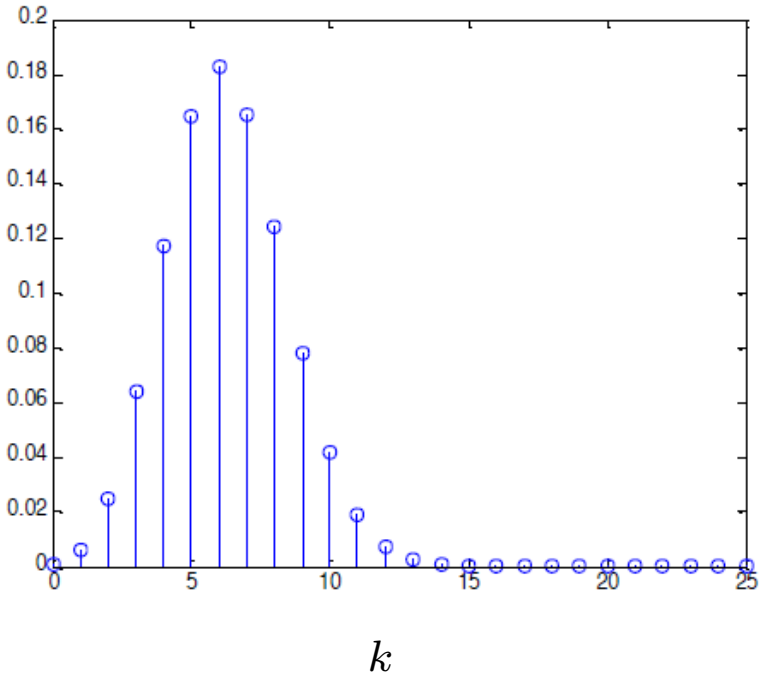
- The number of 1 in a specific sequence x^n is denoted by $k(x^n)$.
- The probability of a specific sequence with k ones is: $\Pr(k)=p^k(1-p)^{(n-k)}$
- The number of sequences of length n with k ones is $N(k)=\left(\begin{array}{c} n \\ k \end{array}\right)$
- The probability of generating a sequence of k ones is $\Pr(n,k)=\left(\begin{array}{c} n \\ k \end{array}\right)p^k(1-p)^{n-k}$
- $\mathbb{E}[k]=pn$
- $std(k)=\sqrt{\mathbb{E}\left[(k-\mathbb{E}[k])^2\right]}=\sqrt{np(1-p)}$. The standard deviation is small with respect to the mean as n increases!

Example: the AEP in a Bernoulli process

Take $p = \frac{1}{4}, n = 25$



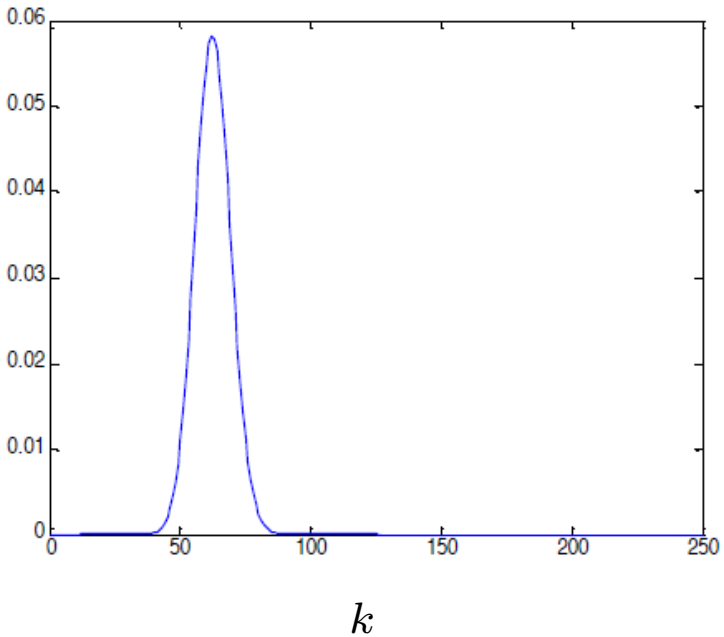
Number of different sequences with k ones



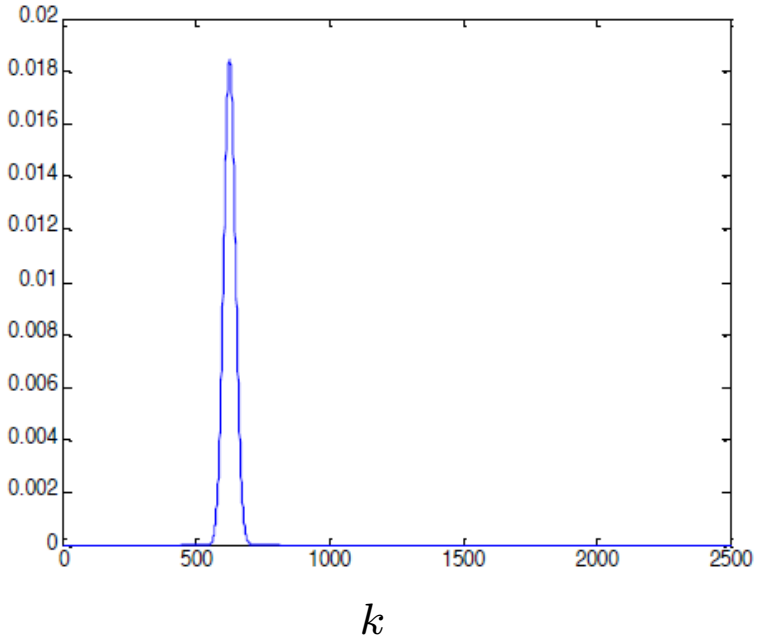
Prob. of generating a sequence of k ones

Example: the AEP in a Bernoulli process

Take $p = \frac{1}{4}$ for increasing n



Prob. of generating a sequence of k ones, $n = 250$



Prob. of generating a sequence of k ones, $n = 2500$

Example: the AEP in a Bernoulli process

From the plots it seems clear that the number of ones in a typical sequence is $k \approx pn$. Let us check it by evaluating the AEP:

$$\begin{aligned}\frac{1}{n}\log p(x^n) &= \frac{1}{n}\log\left(p^k(1-p)^{n-k}\right)\\ &= \frac{1}{n}\left(k\log p + (n-k)\log(1-p)\right)\\ &\approx p\log p + (1-p)\log(1-p) = -H(X)\end{aligned}$$

- **Case 1**. For $n = 2500$, if $p = \frac{1}{4}, H(X) = 0.8113$
  The number of typical sequences is $\approx 2^{nH(X)} = 2^{2029}$
- **Case 2**. For $n = 2500$, if $p = \frac{1}{100}, H(X) = 0.08$
  The number of typical sequences is $\approx 2^{nH(X)} = 2^{200}$

Data compression

As a consequence of the AEP, it is possible to find short descriptions of any realization $x^n = x_1 x_2 ... x_n$ of the random process $X^n = X_1 X_2 ... X_n$.

Theorem (Source coding theorem)

Let X^n be a sequence of i.i.d. random variables, and let $\epsilon > 0$. There exist a code that maps observed sequences x^n of n symbols into binary strings of length $\ell(\mathcal{C}_n(x^n))$ such that the mapping is one-to-one and the average length is

$$\mathbb{E}\left[\frac{1}{n}\ell(\mathcal{C}_n(X^n))\right] \leq H(X) + \epsilon$$

for n sufficiently large.

Data compression

Proof.

1. Let us divide all possibles sequences $\mathcal{X}^n$ into two sets: the typical set $\mathcal{A}_\epsilon^{(n)}$ and its complement $\overline{\mathcal{A}}_\epsilon^{(n)}$.
2. We order the elements in $\mathcal{A}_\epsilon^{(n)}$ and represent each possible sequence by giving an index to it. Since

$$\left|\mathcal{A}_\epsilon^{(n)}\right| \leq 2^{n(H+\epsilon)}$$

we need no more than $(n(H+\epsilon)+1)$ bits.
3. Let us prefix all sequences by a 0 so as to distinguish the typical set from its complement.
4. We order the elements in $\overline{\mathcal{A}}_\epsilon^{(n)}$ and use an index of $(n \log |\mathcal{X}| + 1)$ bits plus a 1 for prefix.

We can now evaluate the average length of the coded message if n is large enough:

Data compression

Proof (cont.).

$$
\begin{aligned}
\mathbb{E}\left[\ell(\mathcal{C}_n(X^n)\right] &= \sum_{x^n \in \mathcal{X}^n} p(x^n)\ell(\mathcal{C}_n(x^n)) \\
&= \sum_{x^n \in \mathcal{A}_\epsilon^{(n)}} p(x^n)\ell(\mathcal{C}_n(x^n)) + \sum_{x^n \in \overline{\mathcal{A}}_\epsilon^{(n)}} p(x^n)\ell(\mathcal{C}_n(x^n)) \\
&\leq \sum_{x^n \in \mathcal{A}_\epsilon^{(n)}} p(x^n)(n(H+\epsilon)+2) + \sum_{x^n \in \overline{\mathcal{A}}_\epsilon^{(n)}} p(x^n)(n\log|\mathcal{X}|+2) \\
&= \Pr\left(\mathcal{A}_\epsilon^{(n)}\right)(n(H+\epsilon)+2) + \Pr\left(\overline{\mathcal{A}}_\epsilon^{(n)}\right)(n\log|\mathcal{X}|+2) \\
&\leq n(H+\epsilon)+2+\epsilon\, n\log|\mathcal{X}|+2\epsilon = n(H+\epsilon')
\end{aligned}
$$

where $\epsilon' = \epsilon + \epsilon\log|\mathcal{X}| + \frac{2}{n}(1+\epsilon)$ can be made small by an appropriate choice of ϵ and n. $\square$

Data compression

In short, the theorem implies that

- Typical sequences are a tiny proportion of all possible sequences (its number depends on $H(X)$);
- Typical sequences occur with a collective probability of about one;
- Each typical sequence occur with about the same probability.

The high probability set

The set $\mathcal{A}_\epsilon^{(n)}$ contains most of the probability but, is it the smallest set?

Let us call $\mathcal{B}_\delta^{(n)} \subset \mathcal{X}^n$ be the smallest set with $\Pr\left(\mathcal{B}_\delta^{(n)}\right) \geq 1-\delta$.

Theorem (3.3)

Let $X_1, X_2, ..., X_n$ be i.i.d. random variables with distribution $p(X)$. For $\delta < \frac{1}{2}$ and $\delta' > 0$, if $\Pr\left(\mathcal{B}_\delta^{(n)}\right) \geq 1-\delta$, then

$$\frac{1}{n}\log\left|\mathcal{B}_\delta^{(n)}\right| > H(X) - \delta'$$

for n large enough.

Thus $\left|\mathcal{B}_\delta^{(n)}\right| > 2^{n(H(X)-\delta')}$, so the high probability set and the typical set are about the same size, if $\delta = \epsilon$.

The high probability set

Proof. Start with a comparative analysis of $\mathcal{A}_\epsilon^{(n)}$ and $\mathcal{B}_\delta^{(n)}$

$$\begin{aligned}\Pr\left(\mathcal{A}_\epsilon^{(n)} \cap \mathcal{B}_\delta^{(n)}\right) &= \Pr\left(\mathcal{A}_\epsilon^{(n)}\right) + \Pr\left(\mathcal{B}_\delta^{(n)}\right) - \Pr\left(\mathcal{A}_\epsilon^{(n)} \cup \mathcal{B}_\delta^{(n)}\right) \\ &\geq 1 - \epsilon + 1 - \delta - 1 = 1 - \epsilon - \delta\end{aligned}$$

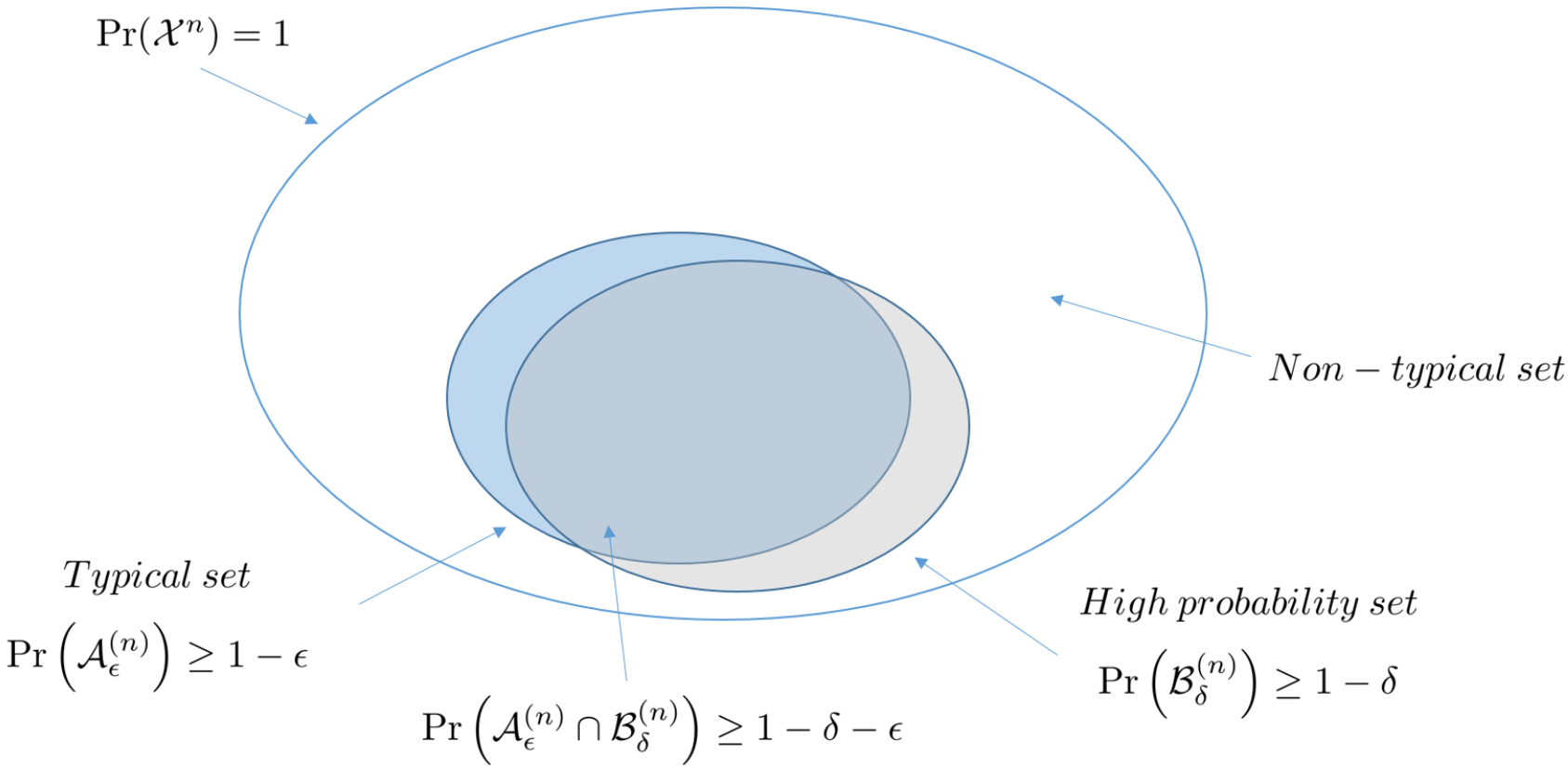
The probability of the intersection of the sets is very large.

$$\begin{aligned}1 - \epsilon - \delta \leq \Pr\left(\mathcal{A}_\epsilon^{(n)} \cap \mathcal{B}_\delta^{(n)}\right) &= \sum_{x^n \in \mathcal{A}_\epsilon^{(n)} \cap \mathcal{B}_\delta^{(n)}} \Pr(x^n) \\ &\leq \sum_{x^n \in \mathcal{A}_\epsilon^{(n)} \cap \mathcal{B}_\delta^{(n)}} 2^{-n(H(X)-\epsilon)} \leq \left|\mathcal{A}_\epsilon^{(n)} \cap \mathcal{B}_\delta^{(n)}\right| 2^{-n(H(X)-\epsilon)} \\ &\leq \left|\mathcal{B}_\delta^{(n)}\right| 2^{-n(H(X)-\epsilon)}\end{aligned}$$

$$\frac{1}{n}\log\left|\mathcal{B}_\delta^{(n)}\right| > \frac{1}{n}\log(1-\epsilon-\delta) + H(X) - \epsilon = H(X) - \delta' \qquad \square$$

The high probability set

Although $\mathcal{A}_\epsilon^{(n)}$ and $\mathcal{B}_\delta^{(n)}$ have nearly the same size, they are not the same. It suffices to show that the most likely sequences (first elements of the δ-sufficient set) are not contained in the ϵ-typical set.



The high probability set

Consider a Bernoulli process with $Pr(X = 1) > Pr(X = 0)$: the most likely sequence is the one having all '1', but it is not present in $\mathcal{A}_\epsilon^{(n)}$ because the sequences in $\mathcal{A}_\epsilon^{(n)}$ contain those whose number of '1' is close to np. Those are also in $\mathcal{B}_\delta^{(n)}$ since the intersection is large.

How to build the high probability set?

It is simple: start from the highest probability sequence(s) and progressively add sequences of deacreasingly smaller probability. This set contains the maximum concentration of probability mass.

Why do we study $\mathcal{A}_\epsilon^{(n)}$ instead of the high probability set?

For compression purposes $\mathcal{B}_\epsilon^{(n)}$ would be more suitable (it has less components), but we cannot count the number of its elements. Additionally, with $\mathcal{A}_\epsilon^{(n)}$ we can use the fact that all sequences have nearly the same probability.

Entropy rate of ergodic sources

The AEP theorem states that $nH(X)$ bits suffice to describe n i.i.d. random variables. What if these variables $X_1, X_2, ..., X_n$ have some statistical dependence?

In this case the entropy rate of a stochastic process is defined as

$$H(\mathbf{X}) = \lim_{n\to\infty} \frac{1}{n} H(X_1, X_2, ..., X_n)$$

the per symbol entropy of n random variables (note that for iid sources we have used so far the notation $\mathbf{X} = X^n$).

We can also define another magnitude:

$$H'(\mathbf{X}) = \lim_{n\to\infty} H(X_n|X_{n-1}, X_{n-2}..., X_1)$$

as the entropy of the last random variable given the past.

Both magnitudes are equivalent for stationary processes (see T. Cover et al, *Elements of Information Theory*, chapter 4)

Entropy rate of a Markov chain

The entropy rate of a stationary Markov chain (see lesson 1) can be written as

$$H(\mathbf{X}) = \lim_{n\to\infty} H(X_n|X_{n-1}, X_{n-2}..., X_1) = \lim_{n\to\infty} H(X_n|X_{n-1})$$

$$= H(X_2|X_1) = \sum_{x_1\in\mathcal{X}} p(x_1)H(X_2|X_1 = x_1) = \sum_{i,j=1}^{N_{states}} p(i)[\mathbf{P}]_{i,j} \log [\mathbf{P}]_{i,j}$$

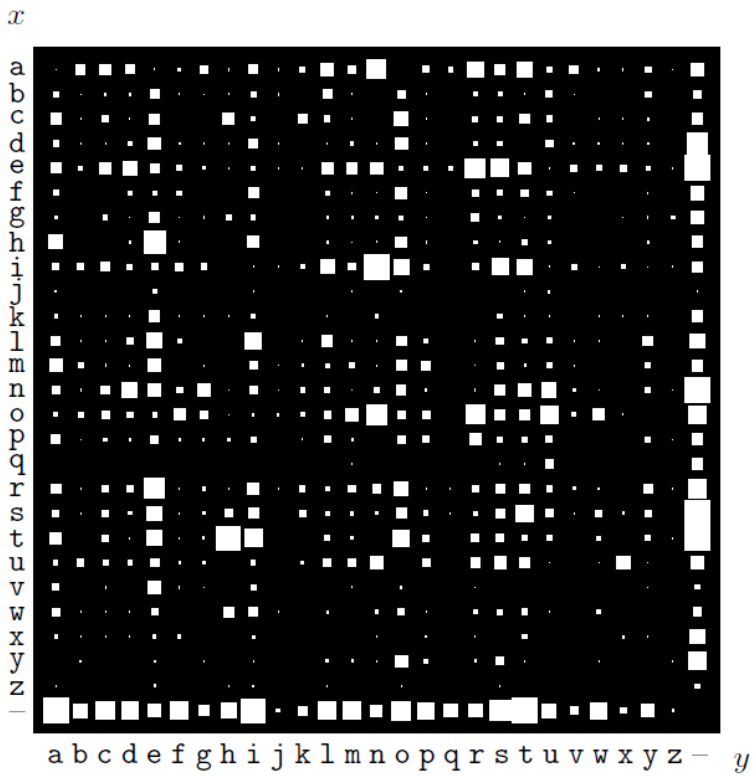
For the two states Markov chain in lesson 1, prove that

$$H(\mathbf{X}) = \frac{\beta}{\alpha+\beta} H(\alpha) + \frac{\alpha}{\alpha+\beta} H(\beta)$$

where $H(\alpha) = \alpha \log \frac{1}{\alpha}$. Check that $H(\mathbf{X}) = H(X)$ for an iid Markov chain.

Example: the correlation of English

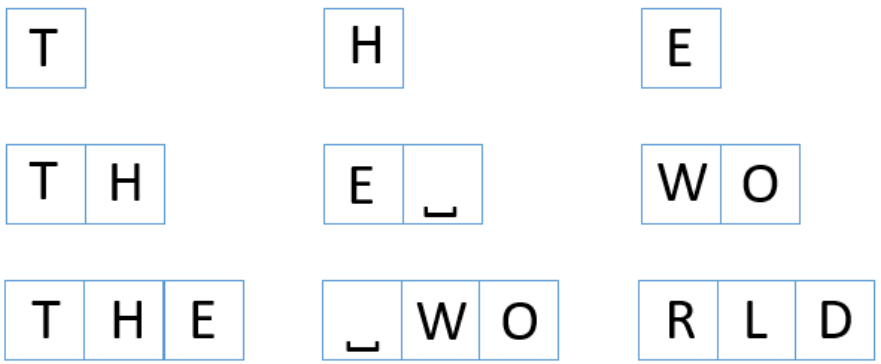
Let us study the depence between blocks of consecutive letters as we increase the number m of letters in a block. As we evaluate the pdf of pairs of letters...



Probability distribution of the 27x27 bigrams in the English language document *The Frequently Asked Questions Manual for Linux* (taken from D. MacKay, *Information Theory, Inference, and Learning Algorithms*).

Example: the correlation of English

From the previous figure, some pairs of letters are quite predictable, given the first letter. Let us increases the block size...



It looks like that the ability to predict each letter from the previous increases, consequently the entropy decreases with m and the prediction of each letter depends less on the letters of other blocks $\rightarrow$ blocks seem to be increasingly independent.

Example: the correlation of English

For an increasing block size m, these are empirical values of the information per letter computed on concatenated long texts (Bible, Shakespeare's works, Moby Dick, etc.) of altogether 7×10^7 characters:

$$H(X_1) = \sum_{i=1}^{27} p(x_i) \log \frac{1}{p(x_i)} = 4.08 \text{ bits/letter}$$

$$H(X_1, X_2) = \frac{1}{2} \sum_{i,j=1}^{27} p(x_i, x_j) \log \frac{1}{p(x_i, x_j)} = 3.32 \text{ bits/letter}$$

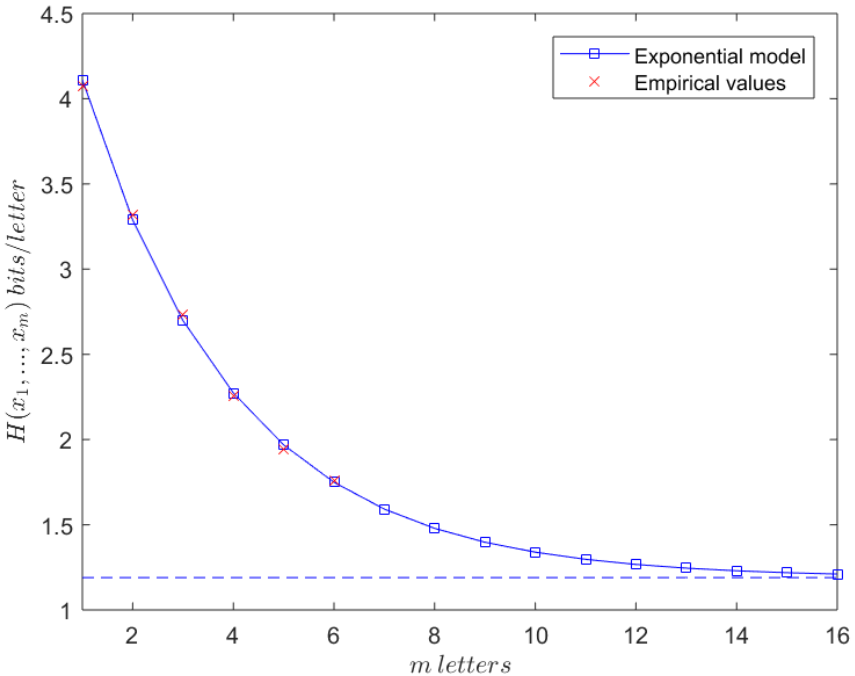
$$H(X_1, X_2, X_3) = \frac{1}{3} \sum_{i,j,k=1}^{27} p(x_i, x_j, x_k) \log \frac{1}{p(x_i, x_j, x_k)} = 2.73 \text{ bits/letter}$$

$$H(\mathbf{X}) = \lim_{m \to \infty} \frac{1}{m} H(X_1, X_2, ..., X_m) = 1.19 \text{ bits/letter}$$

See over for a graphical display (empirical values taken from table I in T. Schürmann, P. Grassberger, "Entropy estimation of symbol sequences", *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 6(3):414-427, $H(\mathbf{X})$ is extrapolated from the set of empirical entropies provided there).

Example: the correlation of English

Let $B_t^m = X_{mt}X_{mt+1}\dots X_{(m+1)t-1}$ be the t-th block of m consecutive English letters.



As m increases, there is less uncertainty per letter in the possible values of the block. Once the block size has grown to $m \approx 14$ (the *correlation length*), the identity of every letter in the block B_t^m depends only on the letters of that block and weakly on those of blocks B_{t-1}^m and B_{t+1}^m. Let us justify it.

Using the chain rule, the joint entropy per letter of blocks B_t^m and B_{t+1}^m is

$$\frac{1}{2m}H(B_t^m, B_{t+1}^m) = \frac{1}{2m}H(B_t^m) + \frac{1}{2m}H(B_{t+1}^m|B_t^m)$$

Example: the correlation of English

From the empirical observation in the plot above, if the block size is $m \geq 14$ the entropy per letter does not change

$$\frac{1}{2m} H(B_t^m, B_{t+1}^m) \approxeq \frac{1}{m} H(B_t^m) \qquad \forall m \geq 14$$

Using both equations, this is achieved if

$$H(B_{t+1}^m | B_t^m) \approxeq H(B_t^m) = H(B_{t+1}^m)$$

where stationarity of English has been applied in the last equality. Hence B_{t+1}^m and B_t^m are nearly independent.

Therefore, we can trivially apply the source coding theorem to compress the source to a number of bits per symbol equal to the entropy rate in this way: assign each B_t^m a value in $\{1, \ldots, |\mathcal{X}|^m\}$, and encode blocks of n of those values (that is $t = 1, \ldots, n$), with n very large.

The SMB Theorem

The SMB theorem formally extends the source coding theorem to ergodic sources:

Theorem (Shannon-McMillan-Breiman Theorem)

For arbritrary $\epsilon > 0$ there exists an integer n_0 such that for every $n > n_0$

$$\lim_{n\to\infty} \Pr\left[\left|-\frac{1}{N}\log p(x_1, x_2, ..., x_n) - H(\mathbf{X})\right| \geq \epsilon\right] = 0$$

where $H(\mathbf{X})$ is the entropy rate. This allows defining the minimum rate of a code for a correlated source. A way to design the code is to resort to the source coding theorem for iid sources applied to blocks of n words, each word of size equal to the correlation length.

Proof. It goes beyond the scope of the course, and can be found in T. Cover et al, *Elements of Information Theory*, chapter 16.

Way through...

- The applications of the AEP and the concept of typicality reach beyond data compression and will be found later in the course.
- We have developed a constructive proof of the source coding theorem, but notice that it only applies to very large sequences.
- Chapter 4 introduces practical codes of finite length that achieve an average length equal to the entropy bound.