2011–12 年度

Information Theory Lecture Notes;
「情報理論」 講義概要

Michael Cohen; **公園 マイケル**

original author: Sugiyama Masahide; **杉山 雅英**

translation: Ishikawa Kimitaka; **石川 君孝**, Suzuki Kenji; **鈴木 健司**, Kinugasa Yasutaka; **蓋 康高**, Takeichi Osamu; **武市 修**, Adachi Kazuya; **足立 和弥**, Nakamura Takenori; **中村 健雄**, Hattori Takayuki; **服部 恭幸**, Sato Hiromitsu; **佐藤 浩晃**, Saze Shougo; **佐瀬 翔吾**, Shiratori Shun; **白鳥 峻**

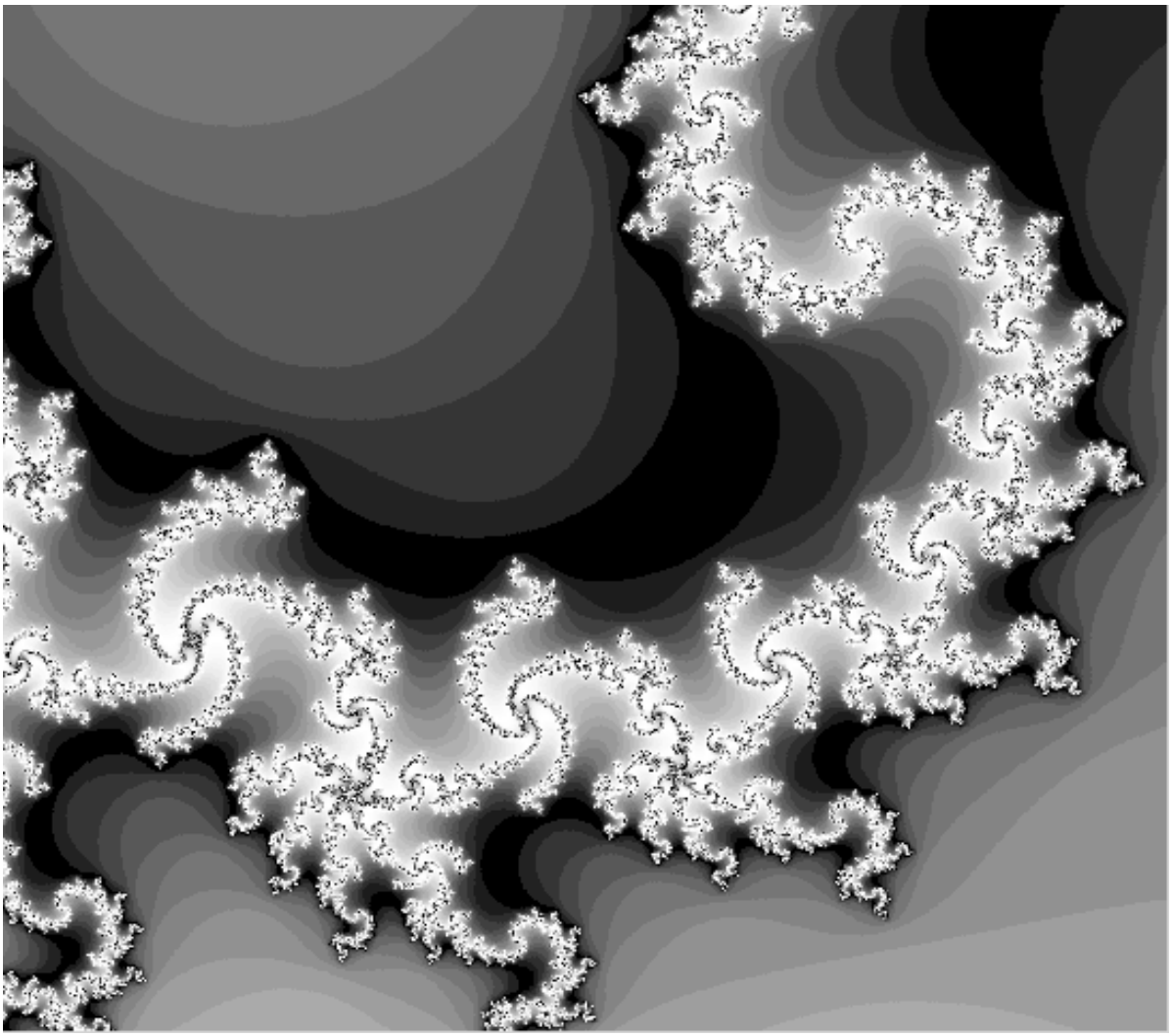


図 1: Part of the Mandelbrot Set, which has low Kolmogorov complexity; 低いコルモゴロフ・コンプレキシティをもったマンデルブロー集合の一部

第1章　1: Introduction; はじめに

1.1　Sets and Probability; 集合(しゅうごう)と確率(かくりつ)

1.1.1　Membership

1.1.1.1　Empty Set; 空集合(くうしゅうごう)

- ϕ (Greek letter *phi* [fi], perhaps since it looks like a slashed zero); 名前の由来は0に斜め線が入っている外見から受け取れる、ϕ（ファイ）
- {}

Example:

$$\text{Black} \quad = \quad \{\}$$

1.1.1.2　Singleton: Set with only one element; 単集合: ただ一つの集合

Example: {living Japanese emperors}　　例: { 現存している天皇 }

Example:

$$\text{Red} \quad = \quad \{\text{red}\}$$

1.1.1.3　Venn Diagrams; ベン図

1.1.2　Structure

1.1.2.1　Sets; 集合(しゅうごう)

Set: order irrelevant, duplicates removed. Example: colors for additive source (like computer monitor):

集合：順序に無関係で、重複の無いもの。例：（コンピュータのモニターのような）色源の加算のための色

$$\begin{aligned} \text{Yellow} &= \{\text{red}, \text{green}\} \\ \text{Cyan} &= \{\text{green}, \text{blue}\} \\ \text{Magenta} &= \{\text{red}, \text{blue}\} \\ \text{White} &= \{\text{red}, \text{green}, \text{blue}\} \end{aligned}$$

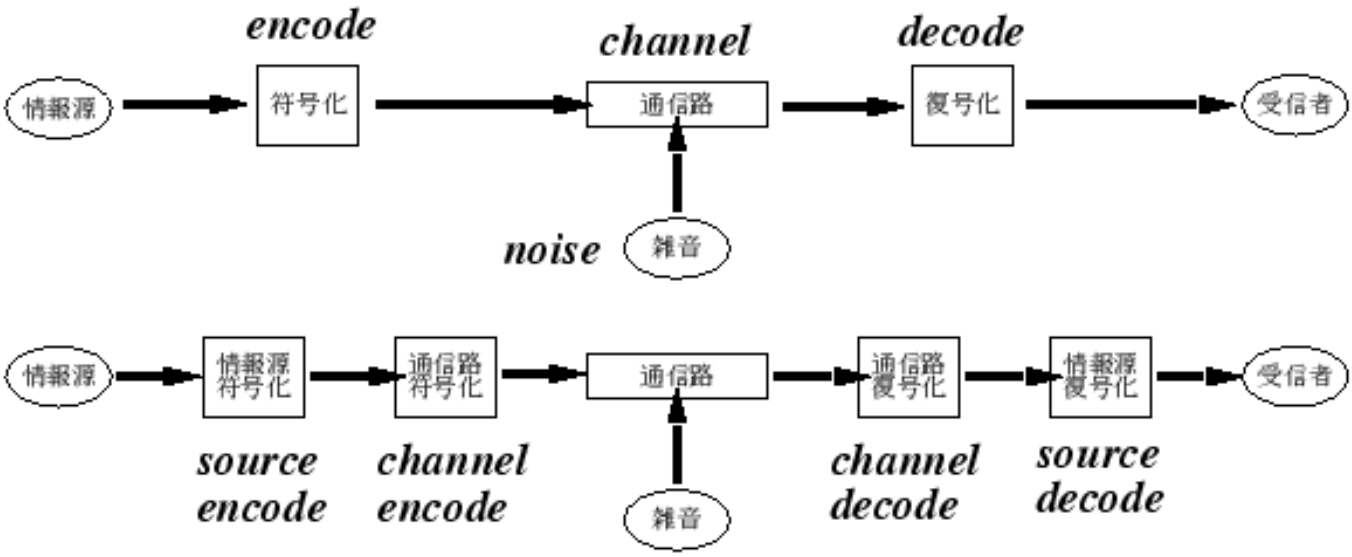


図 1.1: Communication Model: source→sink (receiver); コミュニケーションモデル：情報源 → 受信者. Source encoding minimizes representation (compression), and channel encoding increases robustness (enabling error detection/correction). 情報源符号化は情報の表現を最小にする (圧縮)、通信路符号化は堅牢性を高める (エラー検知/修正を可能にする)。

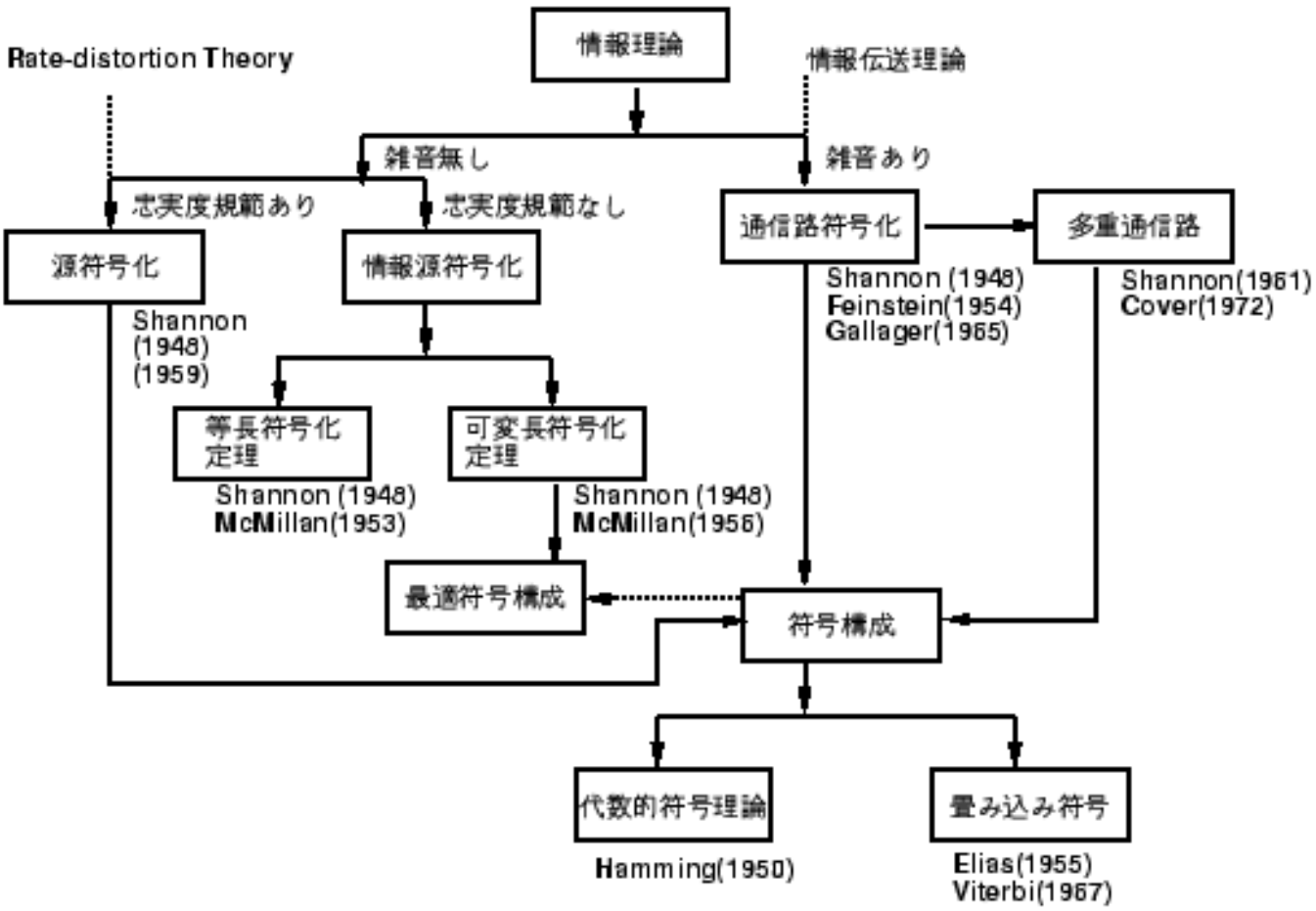


図 1.2: History and Organization of Information Theory; 情報理論の歴史と体系

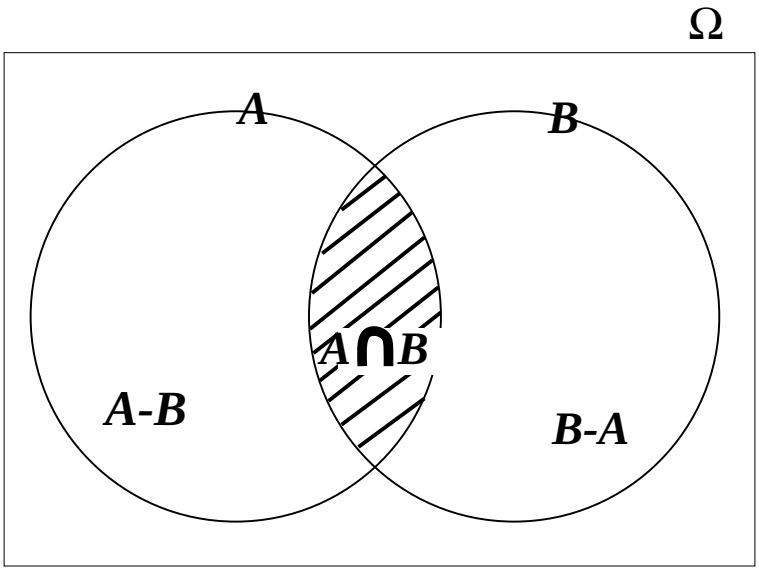


図 1.3: Venn Diagram: Probability Space; 確率 空間. $A \cup B = (A - B) + (A \cap B) + (B - A)$.

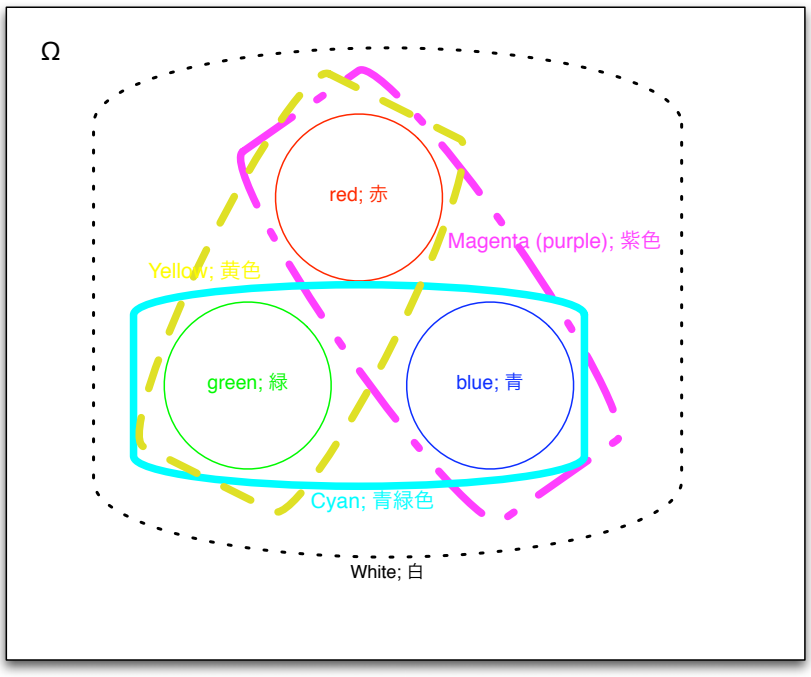


図 1.4: Venn Diagram: Colors— {red, green, blue}$^{\text{primaries}}$, Cyan, Magenta, Yellow, White; ベン図: 色— { 赤, 緑, 青 }、青緑(シアン)、紫(マゼンタ)、黄色(イエロー)、白(ウイト)

1.1.2.2 Elements; 要素(ようそ)、元(げん)

$$\begin{aligned} \text{blue} &\in \text{White} \\ \text{blue} &\in \text{Cyan} \\ \text{blue} &\notin \text{Yellow} \end{aligned}$$

1.1.2.3 Subsets; 部分(ぶぶん)集合: $A \supset B \equiv B \subset A$

$$\begin{aligned} \text{Yellow} &\subset \text{White} \\ \text{White} &\supset \text{Yellow} \\ \text{Yellow} &\not\subset \text{Cyan (since Red} \not\subset \text{Cyan, since red} \notin \text{Cyan)} \end{aligned}$$

1.1.2.4 Proper Subsets; 真部分(しんぶぶん)集合: $B \subsetneq A$

B is a proper subset A iff (if and only if) $B \subseteq A$, $B \neq A$, and $B \neq \phi$.

A が $B \subseteq A$、$B \neq A$、$B \neq \phi$ の条件をすべて満たしたときのみ B は真部分集合である。

$$\begin{aligned} \text{Yellow} &\subseteq \text{White} \\ \text{Yellow} &\neq \text{White} \\ \text{Yellow} &\subsetneq \text{White} \\ \text{Yellow} &\neq \phi \\ \text{White} &\subset \text{White} \\ \text{White} &\subseteq \text{White} \\ \text{White} &= \text{White} \\ \text{White} &\subsetneqq \text{White} \end{aligned}$$

1.1.2.5 Infinite Sets; 無限集合

Sets may be finite or infinite. Examples of infinite sets include the prime numbers $\mathcal{P} = \{2, 3, 5, 7, \dots\}$; natural or counting numbers, also known as the positive integers, $\mathcal{N} = \{0, 1, 2, \dots\}$; integers $\mathcal{Z} = \{\dots, -1, 0, 1, \dots\}$; rationals $\mathcal{Q} = \{\frac{a}{b}, a, b \in \mathcal{Z}\}$; reals $\mathcal{R}$; and complex numbers $\mathcal{C} = \{a + bi\}$. These can be arranged into a containment hierarchy:

集合は有限でも無限でもよい。自然数つまり正の整数素数 $\mathcal{P} = \{2, 3, 5, 7, \dots\}$; 自然数 $\mathcal{N} = \{0, 1, 2, \dots\}$; 整数 $\mathcal{Z} = \{\dots, -1, 0, 1, \dots\}$; や有理数 $\mathcal{Q} = \{\frac{a}{b}, a, b \in \mathcal{Z}\}$; 実数 $\mathcal{R}$; 複素数 $\mathcal{C} = \{a + bi\}$ などが使える。これらは以下のような包含階層に入れられる。

$$\overset{\text{prime; 素数}}{\mathcal{P}} \subseteq \overset{\text{natural; 自然数}}{\mathcal{N}} \subseteq \overset{\text{integer; 整数}}{\mathcal{Z}} \subseteq \overset{\text{rational; 有理数}}{\mathcal{Q}} \subset \overset{\text{real; 実数}}{\mathcal{R}} \subset \overset{\text{complex; 複素数}}{\mathcal{C}} \tag{1.1}$$

Equation	Solution	Class of Numbers
$2x = 4$	$x = 2$	natural numbers (unsigned)
$2x + 4 = 0$	$x = -2$	integers (signed)
$4x = 2$	$x = \frac{1}{2}$	rationals
$x^2 = 2$	$x = \pm\sqrt{2}$	reals
$x^2 + 2 = 0$	$x = \pm\sqrt{2}\ i$	imaginary
$x^2 - 4x + 6 = 0$	$x = 2 \pm \sqrt{2}\ i$	complex

表 1.1: Classes of Algebraic Solutions

1.1.3 Operations

1.1.3.1 Set Subtraction; 差集合: $A - B$ (non-commutative; 可換でない)

Subtrahend only removed if present in original set; there is no negative-valued membership.

元の集合が存在するならば、減数を削除するだけでよい。帰属関係に負はない。

$$\begin{aligned} A - B &= \{x \overset{\text{such that}}{|} x \in A \text{ and } x \notin B\} && (1.2a) \\ &= A \cap \overline{B} && (1.2b) \\ &\not\equiv B - A && (1.2c) \end{aligned}$$

$$\begin{aligned} \text{Cyan} - \text{Yellow} &= \{\text{blue}, \text{green}\} - \{\text{red}, \text{green}\} \\ &= \{\text{blue}\} \\ &\neq \text{Yellow} - \text{Cyan} \\ \text{Yellow} - \text{Cyan} &= \{\text{red}\} \end{aligned}$$

$$A \cup B = (A - B) \cup (A \cap B) \cup (B - A) \quad (1.3)$$

as shown in Figure 1.3.

図 1.3。

1.1.3.2 Set Complement; 補集合: $\overline{A} = \Omega - A$

$$\begin{aligned} \overline{A} &= \{x | x \notin A\} && (1.4) \\ \overline{\text{Red}} &= \overline{\{\text{red}\}} \\ &= \Omega - \{\text{red}\} \\ &= \{\text{red}, \text{green}, \text{blue}\} - \{\text{red}\} \\ &= \{\text{green}, \text{blue}\} = \text{Cyan} \end{aligned}$$

1.1.3.3 Set Union; 和集合: $A \overset{\text{“cup”}}{\cup} B$ (commutative; 可換)

Like addition, but without duplicates. 加算的だが重複は無い。

$$A \cup B = \{x | x \in A \textbf{ or } x \in B\} \tag{1.5a}$$
$$\equiv B \cup A \tag{1.5b}$$

$$\begin{aligned} \text{Yellow} \cup \text{Cyan} &= \{\text{red}, \text{green}\} \cup \{\text{green}, \text{blue}\} \\ &= \{\text{red}, \text{green}, \text{blue}\} \quad (\textbf{not } \text{“\{red, green, green, blue\}”}) \\ &= \text{White} \\ &= \text{Cyan} \cup \text{Yellow} \end{aligned}$$

1.1.3.4 Set Intersection; 積集合: $A \overset{\text{“cap”}}{\cap} B$ (commutative; 可換)

$$A \cap B = \{x | x \in A \textbf{ and } x \in B\} \tag{1.6a}$$
$$\equiv B \cap A \tag{1.6b}$$

$$\begin{aligned} \text{Yellow} \cap \text{Cyan} &= \{\text{red}, \text{green}\} \cap \{\text{green}, \text{blue}\} \\ &= \{\text{green}\} \\ &= \text{Cyan} \cap \text{Yellow} \end{aligned}$$

1.1.3.5 DeMorgan's Laws; ド・モルガンの法則

See Figure 1.5. 図 1.5

Set notation: 集合についての注釈:

$$\overline{A \cup B} \equiv \overline{A} \cap \overline{B} \tag{1.7a}$$
$$\overline{A \cap B} \equiv \overline{A} \cup \overline{B} \tag{1.7b}$$

Logic notation: 論理演算についての注釈:

$$A \text{ NOR } B \equiv \text{NOT } A \text{ AND NOT } B \tag{1.8a}$$
$$A \text{ NAND } B \equiv \text{NOT } A \text{ OR NOT } B \tag{1.8b}$$

Java boolean notation: Java のブーリアンについての注釈:

$$!(A \mid B) == !A \;\&\; !B \tag{1.9a}$$
$$!(A \;\&\; B) == !A \mid !B \tag{1.9b}$$

C and Mathematica logical expression notation (as '&' and '|' are bitwise operators): C 言語と Mathematica 上での論理演算表現についての注釈 (「&」と「|」はビットに関する演 算子であるため):

$$!(A \;||\; B) == !A \;\&\&\; !B \tag{1.10a}$$
$$!(A \;\&\&\; B) == !A \;||\; !B \tag{1.10b}$$

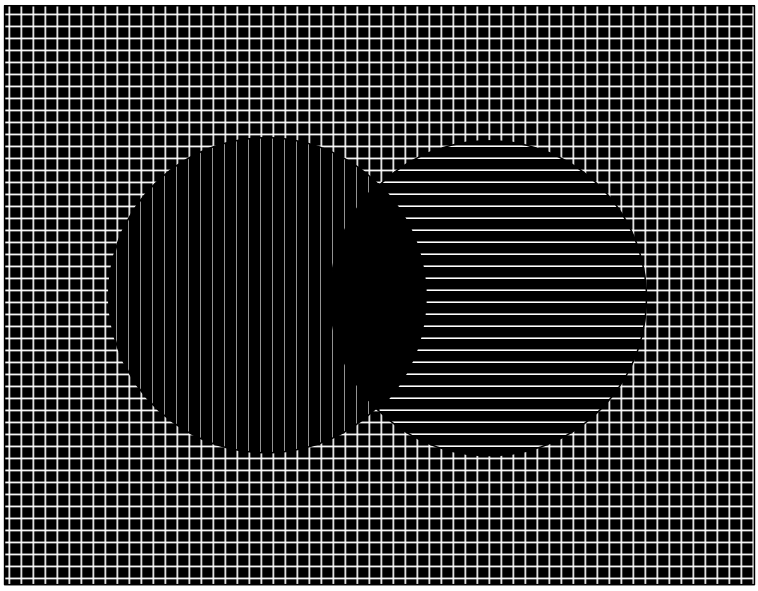


図 1.5: DeMorgan's Law: the intersection of the two circular regions is the complement of the union of the complementary areas (vertically and horizontally striped regions); ド・モルガンの法則：２つの円の交差領域は、補集合領域の結合部分を補う部分である (縦線と横線で示された領域)— $AB = A \cap B = \overline{\overline{A} \cup \overline{B}}$

1.1.4 Cardinality; 要素数

The cardinality of a set is the number of elements in it.

ある集合の濃度はその集合に含まれている要素の数である。

Note that for infinite sets, intuition about cardinality is unreliable. For instance (from eqn. (1.1)),

注意点：無限集合においては濃度における直感は当てにならない。例えば式 (1.1) より

$$|\mathcal{P}| \leq |\mathcal{N}| \leq |\mathcal{Z}| \leq |\mathcal{Q}| < |\mathcal{R}| < |\mathcal{C}| \tag{1.11}$$

but the number of rational numbers can be put into one-to-one correspondence with (has the same cardinality as) the integers, using diagonalization (as illustrated by Table 1.2)!

しかし、有理数は整数と（共に無限大の個数を保持しているので）一対一の関係づけが可能である、対角化で (表 1.2):

$$|\mathcal{Z}| = |\mathcal{Q}|. \tag{1.12}$$

1.1.4.1 Power Set (Set of Subsets); 累乗集合 (巾集合、冪集合)　部分集合の集合

The cardinality of a power set equals $2^{(\text{number of elements in original set})}$.

累乗集合の濃度は２の「元の集合の要素の個数」のべき乗である。

$$|2^A| = \sharp(2^A) = 2^{\sharp(A)} \tag{1.13}$$

Here, "$\sharp(A)$" means the cardinality, the number of elements in the set.

ここで $\sharp(A)$: A の要素の数.

	1	2	3	4	5	...
1	$\frac{1}{1}$	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{4}$	$\frac{1}{5}$	...
		/	/	/	/	/
2	$\frac{2}{1}$	$(\frac{2}{2})$	$\frac{2}{3}$	$(\frac{2}{4})$	$\frac{2}{5}$	...
		/	/	/	/	/
3	$\frac{3}{1}$	$\frac{3}{2}$	$(\frac{3}{3})$	$\frac{3}{4}$	$\frac{3}{5}$	...
		/	/	/	/	/
4	$\frac{4}{1}$	$(\frac{4}{2})$	$\frac{4}{3}$	$(\frac{4}{4})$	$\frac{4}{5}$	...
		/	/	/	/	/
5	$\frac{5}{1}$	$\frac{5}{2}$	$\frac{5}{3}$	$\frac{5}{4}$	$(\frac{5}{5})$	...
		/	/	/	/	/
⋮						⋱

表 1.2: Diagonal ennumeration of rational numbers; 有理数の対角線上での列挙

$$\begin{aligned} A &= \{a_1, a_2, \ldots, a_n\} \\ \sharp(2^A) &= \overbrace{2 \times 2 \times 2 \times \ldots \times 2}^{n} = 2^n \end{aligned}$$

$$\begin{aligned} |2^A| &= |\{B|B \subseteq A\}| && (1.14a) \\ &= |\overbrace{\{0,1\} \times \{0,1\} \times \{0,1\} \times \ldots}^{|A|}| && (1.14b) \\ &= |\{0,1\}^{|A|}| && (1.14c) \end{aligned}$$

Example: three-bit color　　例： 3-ビット色

$$\begin{aligned} 2^{\text{White}} &= 2^{\overbrace{\{\text{red, green, blue}\}}^{n=3}} \\ &= \overbrace{\{\{\}, \{\text{red}\}, \{\text{green}\}, \{\text{blue}\}, \{\text{red, green}\}, \{\text{red, blue}\}, \{\text{green, blue}\}, \{\text{red, green, blue}\}\}}^{2^n=2^3=8} \\ &= \{\phi, \{\text{red}\}, \{\text{green}\}, \{\text{blue}\}, \{\text{red, green}\}, \{\text{red, blue}\}, \{\text{green, blue}\}, \{\text{red, green, blue}\}\} \end{aligned}$$

$$|2^A| = \sum_{n=0}^{|A|} \binom{|A|}{n} = \sum_{n=0}^{|A|} {}_{|A|}C_n \qquad (1.15)$$

$$|2^{\text{White}}| = \overset{\phi}{1} + \overset{\{\text{r}\},\{\text{g}\},\{\text{b}\}}{3} + \overset{\text{Y:\{r,g\},M:\{r,b\},C:\{g,b\}}}{3} + \overset{\text{W:\{r,g,b\}}}{1} = 2^3 = 8$$

1.2 Trials; 試行、試行列と事象

A trial is an experiment with an outcome.　　試行とは結果つきの試み

Example: throw a die　　例: さいころを振る

Example: check today's weather　　例：今日の天気を確かめる

1.2.1 Stochastic Process; 確率過程

A stochastic process is one that yields a random output.

確率過程とはランダムな結果をもたらすものの一つである。

Example: count of thrown die

例: さいころの出た目の出現の現象

1.2.2 Event; 事象

An event is a classification of a trial outcome.

事象とは、試行錯誤を行なったのちに分類した結果である。

Example: a face of a die $\{1, 2, 3, 4, 5, 6\}$. (The location and orientation of the die are irrelevant; only the value of the top face matters.)

例:さいころの目 全体の 集合 $= \{1, 2, 3, 4, 5, 6\}$。(さいころの位置や方向は関係なく、上を向いている面の値のみ)

All these rolls have the same value, and are classified as the same event:

すべてのこれらさいころの目は同じ値を持っており、同じ事象として分類される。

Example: weather is one of {fair ☼, rainy ////, snowy ❄}

例：天気は {晴れ ☼, 雨 ////, 雪 ❄} のいずれかである

1.3 Probability and Probability Space; 確率と確率空間

1.3.1 Intervals; 間隔

Intervals may be continuous or discrete, and open or closed on either end.

区間は継続的でも不連続でもよい。また開区間でも閉区間でもよい

Example: azimuth (real, continuous)

例: 方位角 (実数, 連続)

$$[0, 360^\circ) \text{ or } [0, 2\pi \text{ radians})$$

Example: population (integer, discrete)

例: 人口 (整数 、 離散)

$$[0, \infty)$$

Example: throw of die (integer, discrete)

例: さいころの出目 (整数、離散)

$$[1, 6]$$

Letter	Uppercase	Conventional Interpretation	Lowercase	Conventional Interpretation
alpha	A		α	
beta	B		β	
gamma	Γ		γ	
delta	Δ		δ	
epsilon	E		ϵ, ε	**small threshold**
zeta	Z		ζ	
eta	H		η	
theta	Θ		θ (ϑ)	**azimuth**
iota	I		ι	
kappa	K		κ	
lambda	Λ		λ	(wavelength,) **eigenvalue**
mu	M		μ	micro-
nu	N		ν	
xi	Ξ		ξ	
pi	Π	product	π (ϖ)	circumference:diameter
rho	P		ρ (ϱ)	(density, range)
sigma	Σ	**sum**	σ (ς)	**standard deviation**
tau	T		τ	period
upsilon	Υ		υ	
phi	Φ	(Physics)	ϕ, φ	(phase, elevation)
chi	X		χ	
psi	Ψ	(Psychology)	ψ	(roll)
omega	$\mathbf{\Omega}$	(ohms,) **universal set**	ω	(angular velocity,) **event**

表 1.3: Greek Letters; ギリシア文字

1.3.2 Probability Space; 確率空間

$$\Omega, P \text{ の } \overset{\text{くみ}}{\text{組}} (\Omega, P)$$

$\overset{\text{Omega}}{\Omega}$**:** universal set (set of all possible states of system); 全体集合 (事象($\overset{\text{じしょう}}{\text{事象}}$)の集合($\overset{\text{しゅうごう}}{\text{集合}}$))

P**:** function mapping power set to real numbers in the interval [0,1]; 区間内の実数に累乗集合をマッピングするための関数。

$$2^{\Omega} \to [0, 1]$$

Universal and existential quantifiers:

$\forall$ "for **a**ll" (upside-down 'A')

$\exists$ "there **e**xists" (backwards 'E')

For all states A, these 3 properties are satisfied:

全称記号と存在数量詞:

$\forall$ "for **a**ll" 「任意の」または「すべての」

$\exists$ "there **e**xists" 「存在する」 (かたかな「ヨ」みたい)

$\forall A \in 2^{\Omega}$ に対して関数 $P(A) \in [0, 1]$ が以下の３つの条件を満たす:

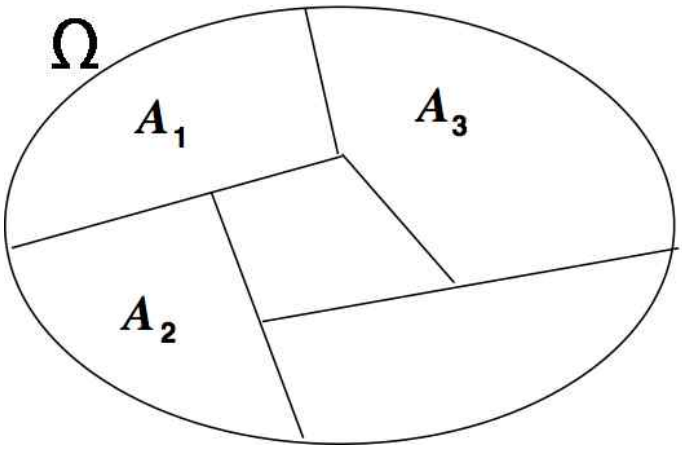


図 1.6: Conditional Probability; 条件付き確率

Probability Space; 確率空間

P1 $P(\phi) = 0$; Experiment $\Rightarrow$ *something* happens

P2 $P(\Omega) = 1$; P(something happening) is unity

P3 $\forall A = \{\omega_1, \omega_2, \ldots, \omega_M\} \subset \Omega$,

$$P(A) = \sum_{n=1}^{M} P(\omega_n),$$

where $P(\omega_n) = P(\{\omega_n\})$

Supplement: above expresses that an event can be composed of (partitionable into) discrete sub-events.

補足：上記は「事象とは (分割可能な) 不連続な部分事象からなる」ということを表現する。

Example: even and odd rolls of a die:

例: サイコロの偶数、奇数:

{⚀, ⚂, ⚅} vs. {⚁, ⚃, ⚄}

For all M states composed of exclusive sub-states A_n,

$\forall A = \cup A_n$ (ただし $A_n \cap A_m = \phi \quad (n \neq m)$) の時

$$P(A) = \sum_{n=1}^{M} P(A_n).$$

For example, the chance of an odd roll is p(⚀) + p(⚂) + p(⚅).

1.4 Random Variables: Distribution; 確率変数: 分布

1.4.1 Random Variables; 確率変数

A function defined on universal set Ω

Ω の上で定義された関数

Example: sum of two dice　　例: 2つのサイコロの合計

$$\begin{aligned} X : X((m,n)) &= m+n \\ \Omega &= \{(m,n)|m,n \in [1,6]\} \end{aligned}$$

1.4.2 Probability Distribution; 確率(かくりつ)分布(ぶんぷ)

If certainty is 100%, then probability $= \overset{\text{unity}}{1}$.　　もし確実性が１００%ならば確率が1に等しい。

$$p_n \in [0,1] : 0 \leq p_n \leq 1 \qquad \forall n$$

$$\begin{aligned} p_n &= P(X = \omega_n) && (1.16) \\ \sum_n p_n &= 1 && (1.17) \end{aligned}$$

1.4.2.1 Example: coin; 例: コイン

front ("heads") ↑　　表(おもて) ↑

back ("tails") ↓　　裏(うら) ↓

Example: fair coin

$$\begin{pmatrix} \omega \\ P \end{pmatrix} = \begin{pmatrix} \uparrow & \downarrow \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

1.4.2.2 Example: fair die; 例: さいころ

$$\begin{pmatrix} \omega \\ P \end{pmatrix} = \begin{pmatrix} \boxdot & \boxdot & \boxdot & \boxdot & \boxdot & \boxdot \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \end{pmatrix}$$

Note that the distribution, shown in Figure 1.7, has unity area.　　この度数分布 (図1.7) は、1となる。

These two distributions happen to be uniform (flat, equiprobable), but such uniformity does not hold in general.　　この二つの度数分布が一様である。しかし、これは一般性において乏しい。

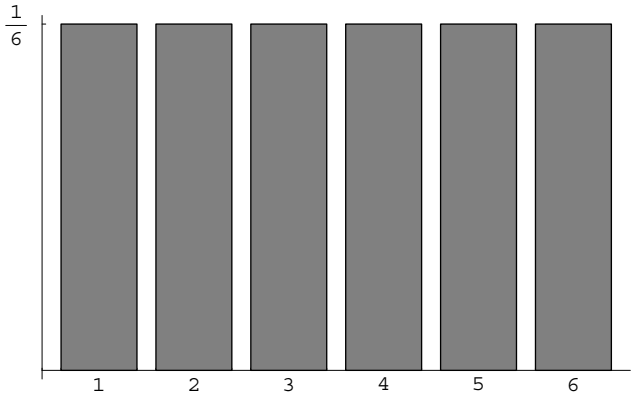


図 1.7: Distribution of throw of a single die; 一つのさいころの出目の分布

+	⚀	⚁	⚂	⚃	⚄	⚅
⚀	2	3	4	5	6	7
⚁	3	4	5	6	7	8
⚂	4	5	6	7	8	9
⚃	5	6	7	8	9	10
⚄	6	7	8	9	10	11
⚅	7	8	9	10	11	12

表 1.4: Sum of throw of two dice; 二つのさいころの出目の和

1.4.2.3 Example of random variable: sum of two dice throws; 確率変数の演算の例

$$
\begin{aligned}
X_1((m,n)) &= m \\
X_2((m,n)) &= n \\
X_3((m,n)) &= m+n \\
X_3 &= X_1 + X_2
\end{aligned}
$$

$$
\begin{pmatrix} \omega \\ P \end{pmatrix} = \begin{pmatrix} 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\ \frac{1}{36} & \frac{1}{18} & \frac{1}{12} & \frac{1}{9} & \frac{5}{36} & \frac{1}{6} & \frac{5}{36} & \frac{1}{9} & \frac{1}{12} & \frac{1}{18} & \frac{1}{36} \end{pmatrix}
$$

As seen in Table 1.4 and Figure 1.8, the distribution is no longer equiprobable (flat), but the area under the curve is still unity. Snake eyes (⚀⚀) and boxcars (⚅⚅) have less liklihood than "lucky seven"s (⚀⚅, ⚁⚄, ⚂⚃, ...). The distribution of the sum of random variables is the convolution of their respective distributions.

表1.4と図1.8からわかるように確率分布は一様ではありませんが、グラフの下部の面積はやはり1です。ピンゾロ (⚀⚀) と6のゾロ目 (⚅⚅) は「ラッキーセブン」(⚀⚅, ⚁⚄, ⚂⚃, ...) より出る可能性が低いと言えます。確率変数の和の確率分布は、それぞれの分布の畳み込みです。

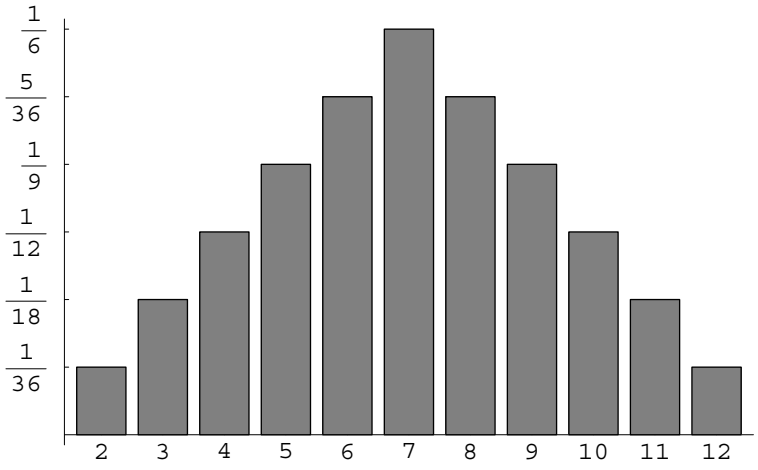


図 1.8: Distribution of sum of two dice; 二つのさいころの出目の和の分布

1.5 Mean (Average) & Variance; 平均と分散

Given probability space P on Ω (real number) and random variable X

確率空間 (Ω, P) が与えられ、Ω の上の (実数値) 確率変数 X

1.5.1 Mean; 平均, Expectation; 期待値, Expected value; X の確率平均

Note: '$\stackrel{\Delta}{=}$' means "is defined as."

注: '$\stackrel{\Delta}{=}$' の意味は「と定義する」。

$$E[X] \stackrel{\Delta}{=} \sum_{\omega\in\Omega} X(\omega)P(\omega) \tag{1.18a}$$

$$= \sum_{n} x_n p_n \qquad x_n = X(\omega_n), \quad p_n = P(\omega_n) \tag{1.18b}$$

Expectation is a linear function:

期待値は 1 次関数:

$$E[X+Y] = E[X] + E[Y] \tag{1.19a}$$

$$E[cX] = cE[X] \tag{1.19b}$$

$$E[aX+b] = aE[X] + b \tag{1.19c}$$

$$\begin{aligned} \text{Expectation of the roll of a die} &= E[X] \\ &= ⚀\cdot\frac{1}{6} + ⚁\cdot\frac{1}{6} + ⚂\cdot\frac{1}{6} + ⚃\cdot\frac{1}{6} + ⚄\cdot\frac{1}{6} + ⚅\cdot\frac{1}{6} \\ &= \frac{21}{6} \end{aligned}$$

Note: Just because the "expected" value of a throw of a die is 3.5 doesn't mean we wouldn't be surprised to see, as in Figure 1.9, a die with $3\frac{1}{2}$ dots!

注: サイコロの期待値は 3.5 であるが $3\frac{1}{2}$ というサイコロの面を見て私たちが驚かないと言うわけではありません (図 1.9)。

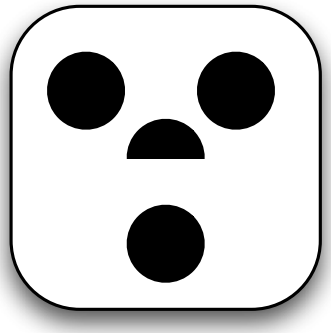


図 1.9: Expectation of a single die throw!; 一つのさいころの期待値のイメージ図！

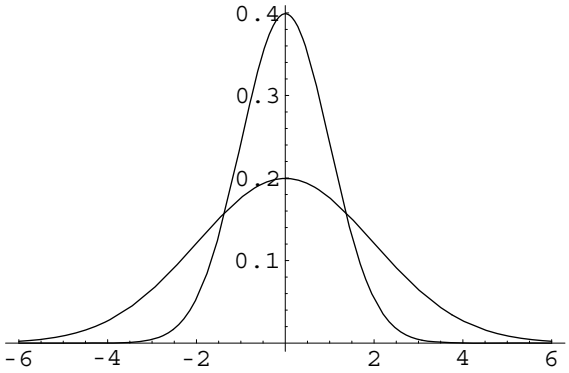


図 1.10: Probability density function of normal (Gaussian) distribution: Two distributions with unequal variances. The area under both curves is the same. 正規 (ガウス) 分布の確率密度関数：異なる分散による 2 つの分布。両方の曲線の下の範囲は同じです。($\overset{\text{unity}}{1} = 100\%$.) A normal or Gaussian distribution has the form $e^{-(t-t_0)^2}$.

1.5.2 Variance; 分散(ぶんさん)

Variance is the expected value of spread or "smear," variation from expected value. It can be thought of as

- the "randomness of a random variable,"
- "the 'error' of a signal," or
- "the energy in the noise."

分散とは確率変数の分布が期待値からどれだけ散らばっているかを示す値のこと。それは以下のように考えられる。

- 任意の数のランダム性
- 信号に対するエラー率
- 音の中のエネルギー度合

$$V[X] \triangleq E[(X - E[X])^2] \tag{1.20}$$

Variance is the dispersion, the mean square about the mean value.

分散はばらつき、平均値についての平均平方。

For a variable that has constant value (c), the variance vanishes:

c は定数、分散なくなります。

$$\begin{aligned} V[c] &= E[(c - E[c])^2] \\ &= E[(c - c)^2] \\ &= 0 \end{aligned}$$

Example: number of heads in one coin flip:　例: 1 枚のコインの返りかた：

	Ω		
	$\uparrow$	$\downarrow$	$\sum$
$p(x)$	$\frac{1}{2}$	$\frac{1}{2}$	$(1 = 100\%)$
$x = \sharp(\uparrow)$	1	0	1
$xp(x)$	$\frac{1}{2}$	0	$E[X] = \frac{1}{2}$
$(x - E[x])^2 p(x)$	$(1 - \frac{1}{2})^2 \frac{1}{2} = \frac{1}{8}$	$(0 - \frac{1}{2})^2 \frac{1}{2} = \frac{1}{8}$	$V[X] = \sigma^2 = \frac{1}{4}$

Example: number of heads in two coin flips:　例: 2 枚のコインの返りかた：

	Ω				
	$(\uparrow, \uparrow)$	$(\uparrow, \downarrow)$	$(\downarrow, \uparrow)$	$(\downarrow, \downarrow)$	$\sum$
$p(x)$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	(1)
$x = \sharp(\uparrow)$	2	1	1	0	4
$xp(x)$	$\frac{2}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{0}{4}$	$E[X] = 1$
$(x - E[x])^2 p(x)$	$\frac{(2-1)^2}{4} = \frac{1}{4}$	$\frac{(1-1)^2}{4} = \frac{0}{4}$	$\frac{(1-1)^2}{4} = \frac{0}{4}$	$\frac{(0-1)^2}{4} = \frac{1}{4}$	$V[X] = \sigma^2 = \frac{1}{2}$

Another example: single (6-faced, fair) die throw:　別の例: サイコロ 1 つを投げるとき:

	Ω						
x	⚀	⚁	⚂	⚃	⚄	⚅	$\sum$
$p(x)$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$(\sum p(x) = 1)$
$xp(x)$	$\frac{1}{6}$	$\frac{1}{3}$	$\frac{1}{2}$	$\frac{2}{3}$	$\frac{5}{6}$	1	$E[X] = \sum xp(x) = 3\frac{1}{2} = \frac{7}{2}$
$(x - E[x])^2 p(x)$	$\frac{-2.5^2}{6}$	$\frac{-1.5^2}{6}$	$\frac{-.5^2}{6}$	$\frac{.5^2}{6}$	$\frac{1.5^2}{6}$	$\frac{2.5^2}{6}$	$V[X] = \sigma^2 = \frac{17.5}{6} \approx 2.9$

1.5.2.1 Standard Deviation; 標準偏差(ひょうじゅんへんさ)

Another way of expressing $V[X]$ is "σ^2_{XX}", the square of the standard deviation σ, which has the same units as the original random variable. For instance, if the basic unit (dimensions of a random variable, like population height) is meters (m), then σ^2 will be in m^2 and σ will be in m.

V[X] の別の表現として、これは元の確率変数と同じ単位を持つ σ^2_{XX}(標準偏差 σ の平方) がある。例えば、基本単位がメーター (m) である場合、σ^2 は m^2 にあるし。σ は m にある。

For example, as might be represented by Figure 1.10, the average income of the U.S.A. and Japan is about the same, but the standard deviation for the U.S. is larger, since the gap between rich and poor is bigger.

例えば、図 1.10 に、アメリカの年収も日本の年収も平均は大体同じだが、アメリカでは裕福な人と貧しい人の格差が大きいので、標準偏差はアメリカの方が大きい。

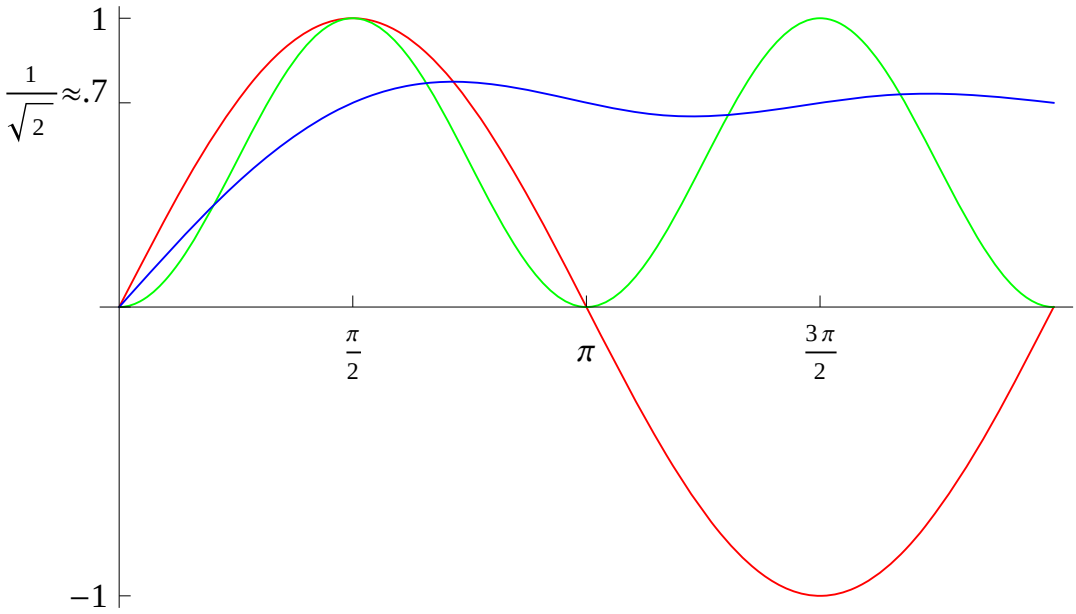


図 1.11: RMS: **r**oot **m**ean **s**quare: original unit sinusoid, its square, and its cummulative RMS; 二乗平均平方根：初期単位の正弦曲線、その平方とその RMS

σ is essentially the same as the root mean square of the AC component of a signal (with zero mean), or RMS. For example, as seen in Figure 1.11, the RMS of a unit sinusoid is about 0.7:

σ は平均が 0 である AC 信号の平均平方根や二乗平均平方根と基本的に同じです。例えば（図 1.11）、単位シヌソイドの二乗平均平方根はおよそ 0.7 です：

$$\text{RMS of unit sinusoid} = \sqrt{\frac{1}{2\pi}\int_0^{2\pi} \sin^2(\theta)\, d\theta} \tag{1.21a}$$

$$= \sqrt{\frac{1}{2\pi}\int_0^{2\pi} \frac{1-\cos(2\theta)}{2}\, d\theta} \tag{1.21b}$$

$$= \sqrt{\frac{1}{2\pi}\frac{\theta - \frac{1}{2}\sin(2\theta)}{2}\Big|_0^{2\pi}} \tag{1.21c}$$

$$= \sqrt{\frac{1}{2}} \tag{1.21d}$$

$$\approx .7 \tag{1.21e}$$

So current from an ordinary Japanese power outlet, nominally at “100 V,” actually has a peak-to-peak excursion (extent, or range) of about ±143 V.

通常、日本のコンセントの電圧は名目上 100 V ですが、実際には最大と最小の間の振れ（範囲）はおよそ ±143 V です。

The energy or power of a information signal, in analogy to one measured in voltage or current amplitude, is the sum of the square of the respective values. The RMS represents the DC amplitude that would be needed to produce the same energy or power as the original signal.

$$\text{voltage} = \text{current}\ \overset{\text{amperes}}{[\text{A}]} \times \text{impedance}\ \overset{\text{ohms}}{[\Omega]} \qquad \text{Ohm's Law}$$

$$\text{power}\ \overset{\text{watts}}{[\text{W}]} = \frac{\text{energy}\ \overset{\text{joules}}{[\text{J}]}}{\text{time}\ \overset{\text{seconds}}{[\text{s}]}}$$

$$\begin{aligned} \text{power} &= \text{voltage} \times \text{current} && \text{Joule's Law} \\ &= \text{voltage}^2/\text{impedance} \\ &= \text{current}^2 \times \text{impedance} \end{aligned}$$

A constant coefficient (multiplier) emerges squared:

一定の係数 (乗数) の二乗が現れる:

$$\begin{aligned} V[cX] &= E[(cX - E[cX])^2] \\ &= E[(cX - cE[X])^2] \\ &= E[c^2(X - E[X])^2] \\ &= c^2 E[(X - E[X])^2] \\ &= c^2 V[X] \end{aligned}$$

For example, the variance regarding height in centimeters of a population would be 10,000 times that of the height in meters, but the standard deviation regarding height in centimeters would be 100 times that of height in meters.

例えば、センチメートルの集合に関する分散は、メートルの集合の 10,000 倍であり、しかし、センチメートルの標準偏差は メートルの標準偏差の 100 倍です。

1.5.2.2 Combining Variances; 分散の組み合わせ

Example; 例題 (れいだい)

Prove this equality: 以下の式を示せ。

$$\overbrace{\underbrace{V[X]}_{\sigma^2}}^{\text{“AC”}} = \underbrace{E[X^2]}_{\text{RMS}^2} - \overbrace{(E[X])^2}^{\text{“DC”}}$$

Proof (since expectation operator is linear and expectation of expectation is same [constant]): 証明 (期待値は 1 次関数であり、期待値の期待値は同じなので):

$$\begin{aligned} V[X] &\triangleq E[(X - E[X])^2] \\ &= E[(X - E[X])(X - E[X])] \\ &= E[X^2 - 2XE[X] + (E[X])^2] \\ &= E[X^2] - 2(E[X])^2 + (E[X])^2 \\ &= E[X^2] - E^2[X] \quad \surd \end{aligned}$$

If a signal is centered (has zero mean), then its standard deviation is the same as its RMS.

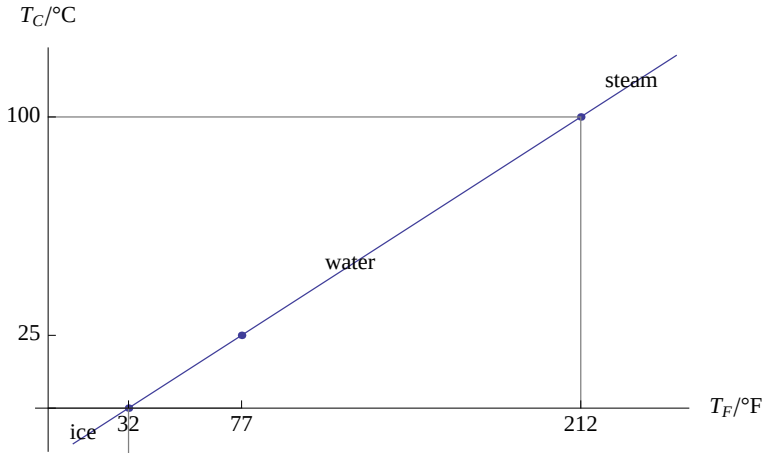


図 1.12: Temperature Conversion from Fahrenheit to Centigrade (Celsius); 華氏から摂氏への温度変化

Constant displacement doesn't affect variance: 定数項の変位は分散に影響しない:

$$V[aX+b] = a^2V[X] \tag{1.25a}$$
$$\sigma[aX+b] = |a|\sigma[X] \tag{1.25b}$$

As plotted in Figure 1.12, the conversion between temperature in degrees Fahrenheit and degrees Centigrade is

図 1.12 で示されるように、華氏度と摂氏度の温度の間の変化は次のように示される

$$T_C = \frac{5}{9}(T_F - 32) \tag{1.26}$$

Example of variance in temperature scales from five samples:

５つのサンプルにおける温度率の分散の例:

	25°C	100°C	0°C	0°C	0°C	$\sum$
$p(x)$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	$(\sum p(x) = 1)$
$xp(x)$	$\frac{25}{5} = 5$	$\frac{100}{5} = 20$	$3\frac{0}{5} = 0$			$E[T_C] = \sum xp(x) = 25$
$(x - E[x])^2 p(x)$	$\frac{(25-25)^2}{5} = 0$	$\frac{(100-25)^2}{5} = 1125$	$3\frac{(0-25)^2}{5} = 375$			$V[T_C] = \sigma^2 = 1500$
	77°F	212°F	32°F	32°F	32°F	
$p(x)$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{1}{5}$	$(\sum p(x) = 1)$
$xp(x)$	$\frac{77}{5} = 15.4$	$\frac{212}{5} = 42.4$	$3\frac{32}{5} = 19.2$			$E[T_F] = \sum xp(x) = 77.$
$(x - E[x])^2 p(x)$	$\frac{(77-77)^2}{5} = 0$	$\frac{(212-77)^2}{5} = 3645$	$3\frac{(32-77)^2}{5} = 1215$			$V[T_F] = \sigma^2 = 4860$

Confirm: 確認:

$$\begin{aligned} V[T_C] &\stackrel{?}{=} V[\frac{5}{9}(T_F - 32)] \\ &= (\frac{5}{9})^2 V[T_F] \\ 1500 &= \frac{25}{81} \cdot 4860 \quad \surd \end{aligned}$$

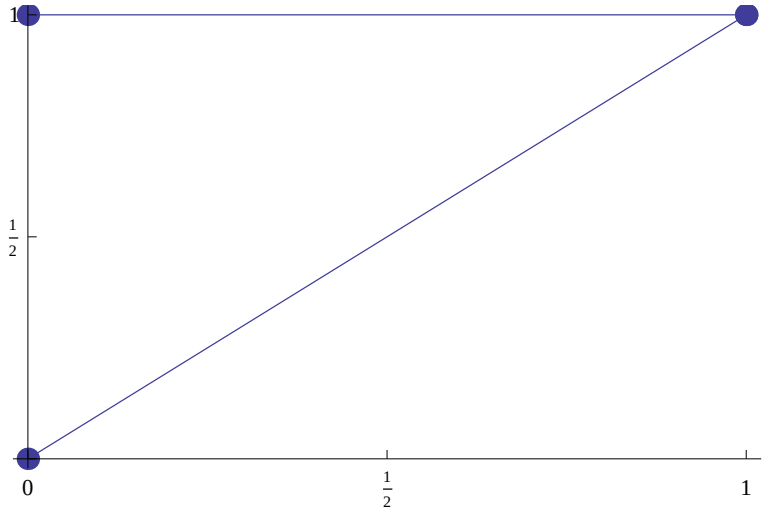


図 1.13: Independent vs. Correlated Datasets

Extra note: Variance of linear combination *of independent variables*:

特記事項：**独立変数**の 1 次結合の分散

$$V[aX + bY] = a^2V[X] + b^2V[Y]. \tag{1.27}$$

In general, for random variables that are not independent, the expectation of a product *does not* equal the product of the respective expectations:

一般的に、独立ではない確率変数において、期待値の論理積は、それぞれの予想の期待値に等しく**ありません**。

$$E[XY] \neq E[X]E[Y]. \tag{1.28}$$

For example, in Figure 1.13,

	X	Y	XY
	0	1	0
	1	1	1
E[·] = avg.	$\frac{1}{2}$	1	$\frac{1}{2} = \frac{1}{2}$
	0	0	0
	1	1	1
E[·] = avg.	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{4} \neq \frac{1}{2}$

In multiplicative composition, the whole is greater than the sum of the parts. Similarly, the variance in the sum of random variables is greater than the sum of the respective variances.

倍数構成で、全体値は部分の和より大きい。同様に、ランダム変数の合計での分散は、それぞれの分散の合計値より大きい。

$$V[X + Y] = V[X] + V[Y] + 2 \text{ Covariance}[X, Y] \tag{1.29}$$

Proof:

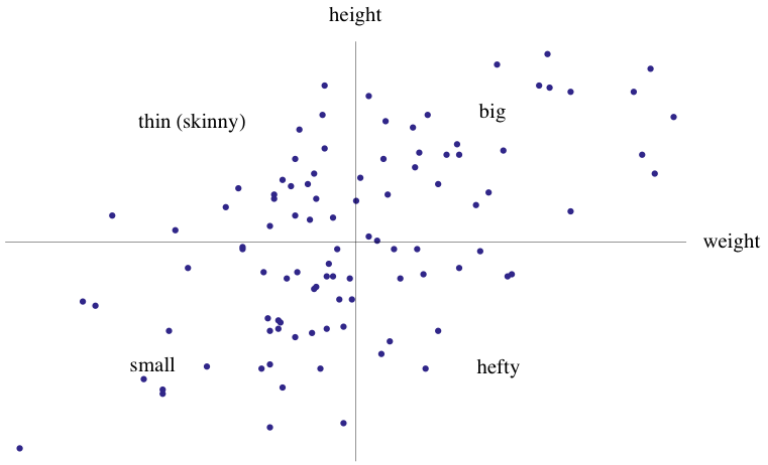


図 1.14: Height vs. Weight for a Population Sample; あるサンプル個体群における身長 対 体重

$$
\begin{aligned}
V[X+Y] &\triangleq E[((X+Y)-E[X+Y])^2] \\
&= E[(X+Y-E[X+Y])(X+Y-E[X+Y])] \\
&= E[X^2+Y^2+2XY-2(X+Y)E[X+Y]+E^2[X+Y]] \\
&= E[X^2]+E[Y^2]+E[2XY]-2E[X+Y]E[X+Y]+(E[X]+E[Y])^2 \\
&= E[X^2]+E[Y^2]+E[2XY]-2(E[X]+E[Y])^2+(E[X]+E[Y])^2 \\
&= E[X^2]+E[Y^2]+2E[XY]-(E^2[X]+E^2[Y]+2E[X]E[Y]) \\
&= (E[X^2]-E^2[X])+(E[Y^2]-E^2[Y])+2E[XY]-2E[X]E[Y] \\
&= V[X]+V[Y]+2(E[XY]-E[X]E[Y])
\end{aligned}
$$

1.5.2.3 Covariance; 共分散 and Correlation; 相関

$$
\begin{aligned}
\text{Covariance}[X,Y] &\triangleq E[(X-E[X])(Y-E[Y])] && \text{(1.30a)} \\
&= E[XY]-E[X]E[Y] && \text{(1.30b)}
\end{aligned}
$$

Since coin flips are independent of each other, covariance of a pair of flips vanishes, and the variance of two coin flips, as calculated near the end of §1.5.2, is twice that for a single flip.

コインの返り方は互いに独立しているため、コインの返り方の共分散は 0 になり、§1.5.2 の最後で計算されているように、二回コイントスをした時の分散は一回した時の 2 倍となる。

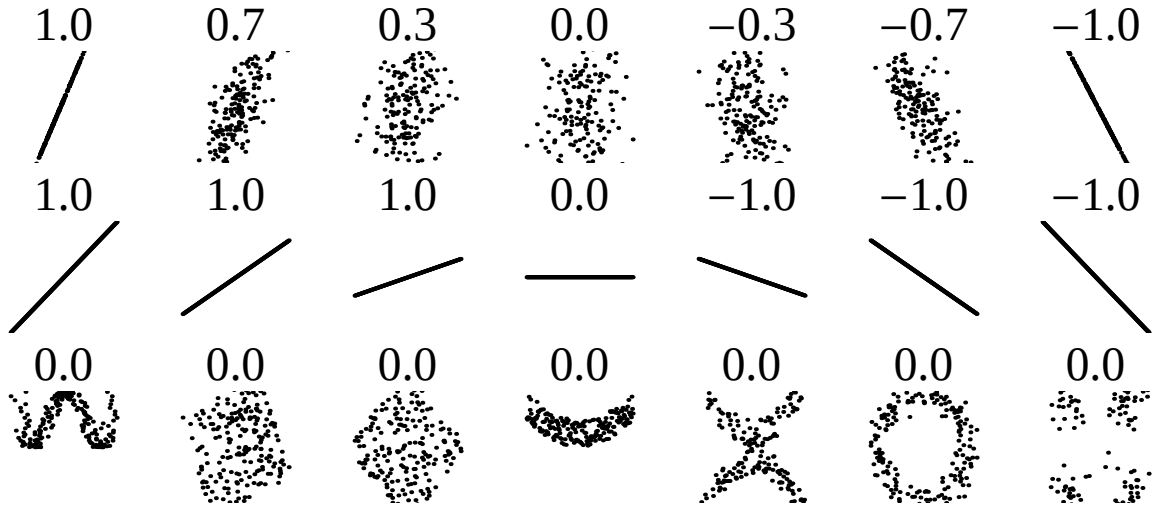


図 1.15: Correlation: Several sets of (x, y) points, with the correlation coefficient of x and y for each set. Note that the correlation reflects the noisiness and direction of a linear relationship (top row), but not the slope of that relationship (middle), nor many aspects of nonlinear relationships (bottom). (Note: The figure in the center has a slope of 0, but in that case the correlation coefficient is undefined because the variance of Y is zero.); 相関: (x, y) で示される点が数セットあり、それぞれのセットにおける x と y の相関係数がある。上段における相関はノイズと線形関係の傾斜を反映しているが、中段における相関においては線形関係の傾きは反映されていない。下段の非線形関係においても同様に反映されない。この点は注意すべきである。(注意: 中央の図の傾きは 0 だが、その場合 Y の分散は 0 になるので相関係数は定義されない。)

Correlation is extracted from covariance, and measures the normalized alignment of variables. For example, as shown in Figure 1.14, the weight and height of individuals is better correlated (nearer unity) than phone number and month of birth (which is close to zero). If alcohol impairs bowling skill, then bowling score would be negatively correlated with amount of beer drunk while playing.

要素同士の関連の強さを表すのが相関である。例えば、個人の身長と体重（図 1.14）は相関が強い (ほぼ 1) のに対し、電話番号と誕生日は相関が弱い (ほぼ 0)。アルコールがボウリング技術を損なうならば、ボウリングのスコアはプレイ中に飲んだビールの量と相関関係があると言えなくもない。

$$\text{Correlation}[X,Y] \triangleq \frac{\text{Covariance}[X,Y]}{\sqrt{V[X]V[Y]}} \tag{1.31a}$$

$$= \frac{E[(X-E[X])(Y-E[Y])]}{\sigma_x\sigma_y} \tag{1.31b}$$

$$r_{xy} = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i-\overline{x})(y_i-\overline{y})}{\sigma_x\sigma_y} \tag{1.31c}$$

$$-1 \le \text{correlation} \le 1$$

1.6 Conditional Probability; 条件付き確率(じょうけんつ かくりつ)

Conditional probability: the probability of a variable conditioned by another. A's probability when B is assumed (or B's value is known).

条件付確立： B という条件（または既知の値）が与えられた時のAの確立

$$P(A \cap B) \equiv P(A, B) \tag{1.32}$$
$$\equiv P(AB) \tag{1.33}$$
$$P_B(A) \equiv P(A|B) \triangleq \frac{P(A \cap B)}{P(B)} \tag{1.34}$$

Note: '|' here is read "given," and means "knowing" or "assuming," and is *not* an "or bar."

注: この「|」 の意味は「……という条件が与えられた」という意味であり決して「又は (OR)」の「|」ではない。

Generally,

$$P(A \cap B) \leq P(A|B) \tag{1.35}$$

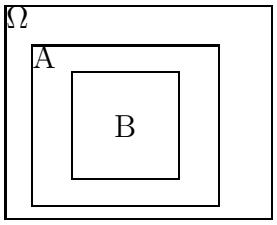
with equality only when the variables are independent.

Example: On any given day, the chance of snow (say 24 days per year, all in winter) is $\frac{24}{365} \approx \frac{1}{15}$, but chance of snow in winter (one of four seasons) is $\frac{24/365}{1/4} = \frac{24}{365/4} \approx \frac{1}{4}$.

例：雪の降る確率は $\frac{24}{365} \approx \frac{1}{15}$(年間 24 日間降るといわれている) だが、冬 (1 年の 1/4) に雪の降る確率は $\frac{24/365}{1/4} = \frac{24}{365/4} \approx \frac{1}{4}$ である。

$$0 \leq P(A|B) \leq 1$$
$$P(B) = P(A \cap B) + P(B - A) \geq P(A \cap B)$$

Q. When does $P(A|B) = 1$?
A. Only when $B \Rightarrow$ A, i.e., when $A \supseteq B$. For example, $P(\text{cold}|\text{snow}) = 1$.



Bayesian analysis is used in spam filters.

ベイズ分析はスパム フィルターに使われている。

Bayes' Theorem; ベイズの定理(ていり)

$$P(B|A) = \frac{P(B)P(A|B)}{P(A)} \tag{1.36}$$

Bayes' Theorem relates joint and conditional entropy.

In general, A and B are *not independent.*

一般に、A と B は独立して**いない。**

$$P(A \cap B) \equiv P(A \text{ and } B) \tag{1.37}$$
$$\neq P(A)P(B) \tag{1.38}$$

Example: the chance of snow in January does *not* equal the chance of snow ($\frac{24}{365}$) times the chance of January ($\frac{1}{12}$).

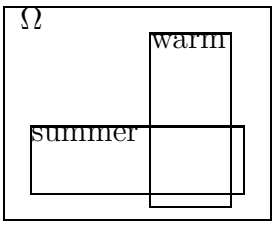
例: １月に雪の降る機会は１年間に雪の降る確率 ($\frac{24}{365}$) 掛ける１月の確率 ($\frac{1}{12}$) では**無い**。

$$P(A \cap B) \stackrel{\Delta}{=} P(A \text{ and } B) \tag{1.39a}$$
$$= P(A, B) \tag{1.39b}$$
$$= P(A)P(B|A) \tag{1.39c}$$
$$= P(B)P(A|B) \tag{1.39d}$$

Q. Does $P(A|B) = P(B|A)$?
A. Conditional probability is not generally symmetric:

$$\frac{P(A \cap B)}{P(B)} \stackrel{?}{=} \frac{P(A \cap B)}{P(A)}$$

That is, $P(A|B) = P(B|A)$ only when $P(A) = P(B)$.



Example:

例:

For single dice roll, define $A = \{x|x \text{ odd}\} = \{\boxdot, \boxdot, \boxdot\}, B = \{\text{first third}\} = \{\boxdot, \boxdot\}$.

$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{P(\text{odd, first third})}{P(\text{first third})} = \frac{P(\{\boxdot\})}{P(\{\boxdot, \boxdot\})}$$
$$= \frac{1/6}{2/6}$$
$$P(B|A) = \frac{P(A \cap B)}{P(A)} = \frac{P(\text{odd, first third})}{P(\text{odd})} = \frac{P(\{\boxdot\})}{P(\{\boxdot, \boxdot, \boxdot\})}$$
$$= \frac{1/6}{3/6}$$
$$P(A|B) \stackrel{?}{=} P(B|A)$$
$$\frac{1}{2} \neq \frac{1}{3} \qquad X$$

but, for $A = \{x|x \text{ odd}\} = \{\boxdot, \boxdot, \boxdot\}, B = \{\text{first } \underline{\text{half}}\} = \{\boxdot, \boxdot, \underline{\boxdot}\}$

$$P(A|B) = P(\text{odd}|\text{first half}) = \frac{P(\text{odd, first half})}{P(\text{first half})} = \frac{P(\{⚀, ⚂\})}{P(\{⚀, ⚁, ⚂\})}$$
$$P(B|A) = P(\text{first half}|\text{odd}) = \frac{P(\text{odd, first half})}{P(\text{odd})} = \frac{P(\{⚀, ⚂\})}{P(\{⚀, ⚂, ⚄\})}$$
$$P(A|B) \stackrel{?}{=} P(B|A)$$
$$\frac{2/6}{3/6} = \frac{2/6}{3/6} \qquad \bigcirc \qquad \text{by coincidence; 一致する}$$

Example; 例題(れいたい)

$(\Omega, P), B \in 2^\Omega, P(B) > 0$ に対して、

$$(\Omega, P(\ .\ |B))$$

And it will be random space. も確率空間となる。

Proof of properties of conditional probability space (in § 1.3.2): 条件付き確率空間の特性の証明(しょうめ):

P1

$$P(\phi|B) = \frac{P(\phi \cap B)}{P(B)} = \frac{P(\phi)}{P(B)} = 0$$

P2

$$P(\Omega|B) = \frac{P(\Omega \cap B)}{P(B)} = \frac{P(B)}{P(B)} = 1$$

P3

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

The numerator's meaning can be evaluated as 分子の集合 $A \cap B$ を以下のように変形できる。

$$A \cap B = (\cup_{\omega \in A}\{\omega\}) \cap B = \cup_{\omega \in A}(\{\omega\} \cap B)$$
$$(\{\omega_1\} \cap B) \cap (\{\omega_2\} \cap B) = \phi$$

Therefore, the equation on the random variable function obtains であるので、確率空間の関数 P の性質 3 より以下の式を得る。

$$P(A \cap B) = \sum_{\omega \in A} P(\{\omega\} \cap B)$$

Therefore, we obtain the random space property P3 regarding $P(\ .\ |B)$ 従って、$P(\ .\ |B)$ に関しても確率空間の関数の性質 3 を得る。

$$
\begin{aligned}
P(A|B) &= \frac{P(A \cap B)}{P(B)} \\
&= \frac{\sum_{\omega \epsilon A} P(\{\omega\} \cap B)}{P(B)} \\
&= \sum_{\omega \in A} \frac{P(\{\omega\} \cap B)}{P(B)} \\
&= \sum_{\omega \in A} P(\{\omega\}|B)
\end{aligned}
$$

1.7 Independence of Random Variables; 確率変数の独立性

Sequence of random variable values　　確率変数の列

$$X_1, X_2, X_3, \ldots, X_i, \ldots$$

1.7.1 Independence; 独立性

If trials X_1, X_2, ... are independent,　　もし試行 X_1,X_2,... が独立なら、

$$P(X_1 = x_1 \cap X_2 = x_2) = P(X_1 = x_1)P(X_2 = x_2). \tag{1.40}$$

For example, the chance of snow on a Sunday is $\frac{1}{7}$ the unconditional chance of snow.　　日曜日に雪が降る非条件付き確率は、$\frac{1}{7}$。

確率変数 $P(X_1 = x_1|X_2)$ を以下で定義する。

$$
\begin{aligned}
P(X_1 = x_1|X_2)(\omega) &= P(X_1 = x_1|X_2 = \omega) \\
&= \frac{P(X_1 = x_1 \cap X_2 = \omega)}{P(X_2 = \omega)} \\
&= \frac{P(X_1 = x_1)P(X_2 = \omega)}{P(X_2 = \omega)} \\
&= P(X_1 = x_1)
\end{aligned}
$$

Therefore (for independent variables X_1 and X_2)　　したがって、独立変数 X1 と X2 は以下のようになる。

$$P(X_1 = x_1|X_2) = P(X_1 = x_1)$$

1.7.2 Memorylessness; 無記憶性

A random variable is memoryless if it doesn't depend on past events.

もし過去の出来事に影響されなければ、確率変数は無記憶性となる。

$$P(X_n = x_n | \overbrace{X_1}^{\text{first; 一回目}}, \overbrace{X_2}^{\text{second; 二回目}}, \ldots, \overbrace{X_{n-1}}^{\text{previous; 一つ前}}) = P(X_n = x_n) \tag{1.41}$$

Therefore, independence ⇒ memoryless.

従って、独立 ⇒ 無記憶 が成り立つ。逆も成り立つ。

If a fair coin is tossed 8 ×, and is heads each time ($P = (\frac{1}{2})^8 = \frac{1}{2^8} = \frac{1}{256} \approx 0.00391 \approx 0.4\%$), the chance of its being heads again on the 9^{th} throw is still $\frac{1}{2}$. Such memorylessness can be contrasted to other games with memory, like Bingo, trump, etc., in which the appearance of a number or card in a round prevents its recurrence.

表裏同確立で出現するコインを 8 回投げたとき毎回表の出る確立は ($P = (\frac{1}{2})^8 = \frac{1}{2^8} = \frac{1}{256} \approx 0.00391 \approx 0.4\%$) 同じコインを投げて 9 回目に表が出る確立は同様に $\frac{1}{2}$ である。無記憶性は、トランプやビンゴなどのような 1 試合の中での番号やカードの出現がその反復を妨げる記憶性のある他のゲームとは対照的である。

1.7.3 Stationarity; 定常

$$P(X_n = x) = \mathrm{c}$$

where c is a constant.

c は定数

Counter-example: When a human plays a game against a computer at a fixed level, the player's chances of winning improve as her skill increases.

反例：人間がレベルを固定したコンピュータと試合をするとき、プレーヤーは技術が向上するため、プレーヤーの勝つ可能性は上昇する。

Example; 例題 (variance of sum equals sum of variance) [大 93, p. 11: Eq. (1.8)]

If X and Y are independent,

X, Y が独立である時、

$$V[X + Y] = V[X] + V[Y]$$

Proof:

証明:

Let

$$a = E[X], b = E[Y]$$

とする。

That is, given

即ち、

$$\begin{aligned} E[X] = a &= \sum_x xP(X = x) \\ E[Y] = b &= \sum_y yP(Y = y), \end{aligned}$$

we will show that the covariance, the cross-product, vanishes (since X and Y are independent):

以下は、X と Y が独立しているので外積が消える。

$$E[(X - a)(Y - b)] = 0.$$

This co-variance vanishes if the variables are independent.

変数が独立である場合、この共分散は 0 になる。

$$\begin{aligned} E[(X-a)(Y-b)] &= \sum_{x,y}(x-a)(y-b)P(X = x \cap Y = y) \\ &= \sum_{x,y}(x-a)(y-b)P(X = x)P(Y = y) \\ &= \sum_x (x-a)P(X = x)\sum_y (y-b)P(Y = y) \\ &= E[X-a]E[Y-b] \\ &= (E[X]-a)(E[Y]-b) \\ &= (a-a)(b-b) \\ &= 0 \end{aligned}$$

Therefore (variance of sum with cross-product vanishing),

従って、(クロス積の消失を用いた和の分散)

$$\begin{aligned} V[X+Y] &= E[((X+Y) - E[X+Y])^2] \\ &= E[(X-a+Y-b)^2] \\ &= E[(X-a)^2 + (Y-b)^2 + 2(X-a)(Y-b)] \\ &= E[(X-a)^2] + E[(Y-b)^2] + \cancel{2E[(X-a)(Y-b)]} \end{aligned}$$

So we conclude from the independence of X and Y

従って以下の式が示された。

$$V[X+Y] = V[X] + V[Y] \quad \surd$$

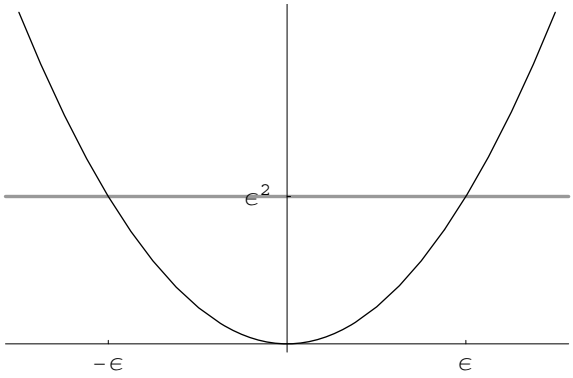


図 1.16: Visualization of Chebyshev Inequality; チェビシェフの不等式の図

1.8 Law of Large Numbers; 大数の法則

1.8.1 Chebyshev Inequality; チェビシェフの不等式

For random variable X of arbitrary (not necessarily normal) distribution, with finite variance σ_{XX} and zero mean,

任意の (正規である必要はない) 分布の確率変数 X のために、有限分散 σ_{XX} とゼロが意味する。

Chebyshev Inequality; チェビシェフの不等式

$$P(|X| \geq \epsilon) \leq \frac{E[X^2]}{\epsilon^2}$$

(Note: 'ϵ' here means an arbitrarily small constant, and has nothing to do with element relational operator '$\in$.')

注意： ここにおいて「ϵ」は任意の小さな定数であって「$\in$」とは無縁である。

$$\begin{aligned}
E[X^2] &= \sum_x x^2 P(X = x) \\
&= \overbrace{\sum_{|x| \geq \epsilon} x^2 P(X = x)}^{\text{beyond or without; 外}} + \overbrace{\sum_{|x| < \epsilon} x^2 P(X = x)}^{\text{within; 内}} \\
&\geq \sum_{|x| \geq \epsilon} x^2 P(X = x) \\
&\geq \sum_{|x| \geq \epsilon} \epsilon^2 P(X = x) \\
&= \epsilon^2 \sum_{|x| \geq \epsilon} P(X = x) \\
&= \epsilon^2 P(|X| \geq \epsilon) \\
P(|X| \geq \epsilon) &\leq \frac{E[X^2]}{\epsilon^2}
\end{aligned} \tag{1.42}$$

If $\epsilon = k\ \sigma_{XX}, k \geq 1$, where $\sigma_{XX} = \sqrt{V[X]}$ (since $E[X^2] = \sigma_{XX}^2$ for centered variable X), then

もし $\epsilon = k\ \sigma_{XX}, k \geq 1$ ならば、ここで $\sigma_{XX} = \sqrt{V[X]}$ (中心の変数 X により $E[X^2] = \sigma_{XX}^2$)、その時

$$P(|X| < \overbrace{k\ \sigma_{XX}}^{\epsilon}) \geq 1 - \frac{\sigma_{XX}^2}{(k\ \sigma_{XX})^2} \tag{1.43}$$

$$= 1 - \frac{1}{k^2} \tag{1.44}$$

No more than $\frac{1}{4} = 25\%$ of the values are more than 2 standard deviations from the mean; no more than $\frac{1}{9} \approx 11\%$ are more than 3 "sigma"s (σs); no more than $\frac{1}{25} = 4\%$ are more than 5; etc. This is looser than the Empirical Rule for normal distributions: 68% of samples within $\pm$ 1 standard deviation; 95% within 2σ; 99.7% within 3; etc.

値のわずか $\frac{1}{4} = 25\%$ でも平均から標準偏差 2σ 以内にある; わずか $\frac{1}{9} \approx 11\%$ でも、3σ となる; わずか $\frac{1}{25} = 4\%$ でも、5σ となるなど。これは正規分布のための経験分布則より不安定である：$\pm 1\sigma$ 標準偏差の中の 68%のサンプル; 2σ の範囲内の 95%; 3σ の範囲内の 99.7%など。

1.8.2 Average of Sequence; 確率変数の列の平均

X_i: Sequence of random variable values

X_i: 確率変数の列

$$E[X_i] = a$$
$$V[X_i] = \sigma^2$$

を満たすとする。

Mean (average or expected value), since expectation is a linear operator.

平均または期待値、数学的期待値は線形演算子である。

$$E[\frac{1}{n}(X_1 + X_2 + \ldots + X_n)] = \frac{1}{n}\sum_{i=1}^{n} E[X_i]$$
$$= \frac{1}{n}na = a$$

1.8.3 (Jakob) Bernoulli Trials; （ヤコブ）ベルヌーイの法則

Three properties of Bernoulli trials:

ベルヌーイ試行の３つの特性:

Binary: Result is either success or failure.

２択: 結果は、成功か失敗かのどちらか

Stationary: Probability of success (failure) is same in every trial.

定常: どの試行でも成功 (失敗）する確率は同じ

Independent: Each outcome has no effect on later trials.

独立: 成り行きは、後の試行に影響を与えない

1.8.4 Law of Large Numbers; 大数の法則

How does frequency of success in a long sequence of trials approach probability of success for a single trial?

試行の過程の中での成功頻度は一回の試行確立に近づくだろうか？

The "Law of Large Numbers," a.k.a. the "Central Limit Theorem," states that the sum (and average) of n random variables tend towards a normal (Gaussian) distribution as n becomes large, independent of the distributions of the original random variables.

中心極限定理として知られる大数の法則は「n個の乱数の合計（平均）はnが十分大きなとき乱数の分布に関係なく正規分布（ガウス分布）に近づく」ということ。

Law of Large Numbers; 大数の法則

$$P(|\frac{1}{n}\sum_{i=1}^{n}(X_i - a)| < \epsilon) \geq 1 - \frac{\sigma^2}{n\epsilon^2}$$

where $a = E[X_i]$ and $\sigma^2 = V[X_i]$

Proof (of Law of Large Numbers), using Chebyshev inequality:

チェビシェフの不等式を用いて証明する:

The r^{th}-order moment is

$$E[X^r] = \sum_{\omega\in\Omega} X^r(\omega)P(\omega) \tag{1.45}$$

so $r = 1$ obtains mean $E[X]$ whereas $r = 2$ obtains mean square of X. The r^{th}-order moment about a point c is defined by

$r = 1$は平均値$E[X]$を得る、一方$r = 2$はXの平均平方を得る。cに関するr次モーメントは以下のように定義される。

$$E[(X - c)^r] = \sum_{\omega\in\Omega} (X(\omega) - c)^r P(\omega) \tag{1.46}$$

(a) lever; でこ　　(b) see-saw; シーソー or balance

図 1.17: (1^{st}-order) Mechanical Moment Arms Create Torque; (一次) 機械的モーメントアーム (支点から力の作用点に降ろした垂線の距離) によるトルク

A very useful and familiar case is the central moment of a variable, i.e. about its mean value. For example, variance is the 2^{nd}-order central moment:

非常に有用でよく知られている場合としては変数の平均値に対して近い値をとる。例えば、分散は平均値に対して2番目の値の時。

$$\sigma_{XX}^2 = V[X] = \sum_{\omega\in\Omega}(X(\omega) - E[X(\omega)])^2 P(\omega). \tag{1.47}$$

Applying the Chebyshev Inequality (eqn. (1.42)) and letting the random variable be the average of a mean-centered sequence,

チェビシェフの不等式 (eqn. (1.42)): 確率変数を平均調整された配列の調節された平均にする。

$$P(|\frac{1}{n}\sum(X_i - a)| \geq \epsilon) \leq \frac{E[|\frac{1}{n}\sum(X_i - a)|^2]}{\epsilon^2}, \tag{1.48}$$

which transforms (since constant $\frac{1}{n}$ emerges squared):

ここで以下のように変形される

$$E\left[\left|\frac{1}{n}\sum(X_i - a)\right|^2\right] = \frac{1}{n^2}E\left[\left|\sum(X_i - a)\right|^2\right].$$

This last term is the expectation of the square of the sum of the central moments,

最後の項は、二次中心モーメントの合計の二乗の期待値

$$V\left[\sum_{i=1}^{n} X_i\right] \triangleq E\left[\left|\sum_{i=1}^{n}(X_i - a)\right|^2\right], \tag{1.49}$$

and because of independence of separate trials from a sequence (as shown in the last section, §1.7), the variance of a sum is the sum of the variance:

であり、独立性をもっているので以下のように計算される (§1.7)。分散の和は、和の分散と同じ結果になります。

$$V\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} V[X_i] = n\sigma_{XX}^2. \tag{1.50}$$

Therefore,

従って、

$$\begin{aligned} E\left[\left|\frac{1}{n}\sum(X_i - a)\right|^2\right] &= \frac{1}{n^2}E\left[\left|\sum_{i=1}^{n}(X_i - a)\right|^2\right] \\ &= \frac{1}{n^2}V\left[\sum_{i=1}^{n} X_i\right] \\ &= \frac{n\sigma^2}{n^2}. \end{aligned}$$

So, one obtains (via eqn. (1.48))

となり、以下を得る (方程式1.48)。

$$\begin{aligned} P\left(\left|\frac{1}{n}\sum(X_i - a)\right| \geq \epsilon\right) &\leq \frac{E\left[\left|\frac{1}{n}\sum(X_i - a)\right|^2\right]}{\epsilon^2} \\ &= \frac{\sigma^2/n}{\epsilon^2}. \end{aligned}$$

Changing the direction of the inequality obtains　従って、以下を得る。

$$P\left(\left|\frac{1}{n}\sum(X_i - a)\right| < \epsilon\right) = \mathbf{1} - P\left(\left|\frac{1}{n}\sum(X_i - a)\right| \geq \epsilon\right) \tag{1.51a}$$

$$\geq 1 - \frac{\sigma^2}{n\epsilon^2} \tag{1.51b}$$

This condition has the form of a Gaussian or normal distribution, $p(x) = \frac{e^{\frac{-(x-\mu)^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}}$, like that in Figure 1.10.

この確率はガウス分布、または正規分布の形になる：$p(x) = \frac{e^{\frac{-(x-\mu)^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}}$ (図 1.10)。

The chance that the average difference from the mean is within a given threshold approaches unity as the number of samples increases.

サンプル数が増加するにつれて、中間値との平均差が与えられた閾値にあるという確率は 1 に近づく。

Example: given 10 (n) fair games of *Janken*, one expects to win half, i.e., 5. One might also sometimes lose all the games, or win eight, etc. Eight is three (ϵ) games off from the expected value. If one plays 1000 games, the expected number (a) of wins is 500, but it would be surprising to actually win exactly 500, or even 503. The denominator scales the fraction.

例:じゃんけんを 10 回 (n) 行ったとき、半分を勝つことを予想する、つまり 5 回. 時々はゲームすべてに負けるとか、8 回勝つなどいろいろな場合がある。8 回勝つということは、期待値から 3 つ (ϵ) 離れたゲームである。1000 回ゲームを行う場合、予想される勝利数 a は 500 である。しかし現実にちょうど 500、あるいは 503 回勝つことは驚くべきことです。分母はその分数の比率である。

1.8.5 Weak Law of Large Numbers; 大数の弱法則

For tolerance ϵ_1 and probability ϵ_2,　許容範囲 ϵ_1 と確率 ϵ_2 において、

$$\overset{\text{“for all”}}{\forall}\ \epsilon_1, \epsilon_2 > 0, \quad \overset{\text{“there exists”}}{\exists}\ N \text{ such that } \forall n > N$$

$$a - \epsilon_1 \leq \frac{1}{n}(X_1 + X_2 + \ldots + X_n) \leq a + \epsilon_1 \tag{1.52}$$

with probability no less than $1 - \epsilon_2$.　となる確率は $1 - \epsilon_2$ 以上である。

$$\left|\frac{1}{n}(X_1 + X_2 + \ldots + X_n) - a\right| \leq \epsilon_1 \tag{1.53}$$

From results of last subsection (eqn. (1.51b)),　と変形できる。

$$P\left(\left|\frac{1}{n}(X_1 + X_2 + \ldots + X_n) - a\right| \leq \epsilon_1\right) \geq 1 - \frac{\sigma^2}{n\epsilon_1^2} \tag{1.54}$$

Choose n (large enough) such that　とできるので、以下を満たすような非常に大きな値の n をとる

$$\frac{\sigma^2}{n\epsilon_1^2} < \epsilon_2$$

$$n > \frac{\sigma^2}{\epsilon_2\epsilon_1^2}$$

So, take this many samples to get the tolerance specified:

とすれば良い。従って、

$$N = \lceil \frac{\sigma^2}{\epsilon_2\epsilon_1^2} \rceil$$

where "$\lceil\cdot\rceil$" (read "ceiling") means the smallest integer no less than the operand.

とすれば良いことになる。シーリングは引数と同等の最小の整数を意味する.

For example, for a fair die, if tolerance $\epsilon_1 = 1$ and probability $\epsilon_2 = \frac{1}{2}$, then $N = \lceil \frac{2.9}{\frac{1}{2}1^2} \rceil = 6$. If we roll the die at least six times, chances are better than half that the average will be within one of the expected value, i.e., between 2.5 and 4.5.

例えば、さいころを振ることを考える。もし許容範囲 $\epsilon_1 = 1$、確率 $\epsilon_2 = \frac{1}{2}$ なら $N = \lceil \frac{2.9}{\frac{1}{2}1^2} \rceil = 6$ である。さいころを少なくとも 6 回振った場合、確率は平均が期待値の範囲内 (2.5 から 4.5) に収まるであろう確率 1/2 より高くなる。

1.9 Markov Process; マルコフ過程(かてい)

$$P(X_n = x_n | X_1, X_2, \ldots, X_{n-1}) = P(X_n = x_n | X_{n-1}) \tag{1.55}$$

Markov process trials are not independent, but have state-dependent transition probabilities.

マルコフ過程の試行は独立ではなく、推移確率に依存する。

First-order Markov process; 単純マルコフ過程:

$$P(X_1 = x_m | X_2 = x_n) = a_{mn} \tag{1.56}$$

Symbol	Meaning	Example
E	Energy	$\mathrm{E} = mc^2$
ϵ	epsilon	for $x < \epsilon \ldots$
$\in$	element of	$0 \in \{\text{Integers}\}$
e	Euler's number, base of natural logarithm	$\ln e = 1$
E	expectation (average, mean)	$E[X] = \frac{1}{N}\sum_{i=1}^{N} x_i$
$\exists$	existential qualifier: "there exists"	$\forall x \; \exists \; x + 1$

表 1.5: Easily confused symbols that resemble 'E'ach other

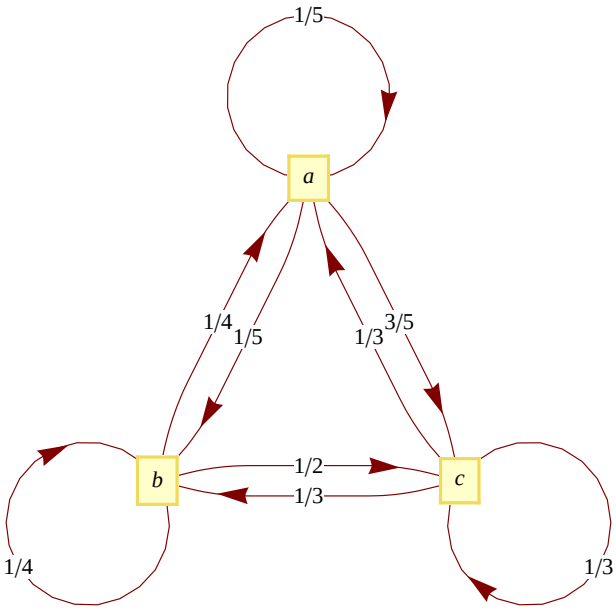


図 1.18: First-order Markov Process Model; 単純マルコフ過程(かてい)のモデル. Example: Each day's weather depends in part on the previous day's. 例: 前日の天気に応じた各日の天気。

第2章　2: Source Coding; 情報源符号化

2.1 Information Source Model and Encoding; 情報源符号

2.1.1 Model of Information Source and Source Encoding Problem; 情報源のモデルと情報源符号化問題

Statistical modeling of information source S　　情報源の統計的モデル化

- information expressed as finite set of symbols (alphabet); 情報は有限個の記号で表される
- information source is original stream of symbols; 情報源は記号の発生源である
- use information source's characters; 情報源の確率的な性質を用いる

Digital (not analog) information, with symbols drawn from finite alphabet　有限の文字から成るシンボル

$$a_1, a_2, \ldots, a_M$$

S: memoryless, stationary, discrete information source　S: 無記憶定常離散的情報源

$$P(X_n = a_i) = \mathrm{c}$$

where c is a constant.　c は定数

Probability doesn't depend on n (the trial) or time　確率は試行回数の n にも時間にも依存しない。

Examples: coin, die, roulette wheel.　例：コイン、サイコロ、ルーレット。
Counter-examples: lottery (depends on number of other bettors), cards (depends on previous hands), bingo (depends on previously drawn balls).　反例：ロータリー（他の人のはりかたに依存する）、トランプ（前の手に依存する）、ビンゴ（前に出た番号に依存する）。

$$S = \begin{pmatrix} a_1 & a_2 & \ldots & a_M \\ p_1 & p_2 & \ldots & p_M \end{pmatrix}$$

$$p_i = P(X_n = a_i)$$

Purpose; 目的

復元できるという条件の下での 平均符号長最小の符号 の
数学的な特徴付け
⇓
エントロピー

Domain	Analog	Digital
numbers	real	integer
property	continuous smooth	discrete quantized, granular
physics	classical (Newton, Einstein)	quantum (Planck, Bohr, Heisenberg)
resolution	infinite	finite
examples	audio or VHS cassettes vinyl records	internet media disk drives CDs, DVDs, & BD (Blu-Ray) discs DNA
jumping	broad (long) jump; はばとび	high jump; たかとび
clock		
metaphor	slope; スロープ	ladder; 梯子

表 2.1: Analog and Digital

The shortest mean encoding is the entropy. The goal of source coding is to make an encoding as short as possible, by removing redundancy in the message, to economize time (for transmission) and/or space (for storage).

最小単位でのエンコーディングはエントロピーである。コーディングの目標はできるだけ短く、無駄を省き、通信時間を短縮することにある。

2.2 (Information) Source Coding; 情報源符号

2.2.1 Bits (binary digits) and Binary States; ビット と 二値法

A bit is the smallest unit of information processed by a digital computer

ビットはデジタルコンピュータで処理される情報の最小単位である。

Digital computers use binary states. These opposing states can have any names, including those in Table 2.2. The names are arbitrary, but serve to distinguish the two states.

ディジタルコンピュータは 2 進法を使用し、この二つの値は区別できるのならば表 2 のようなものを含む名称は任意である

A byte (a.k.a. an octet) is 8 bits. 4 bits is a nibble (or nybble). Modern desktop computers have 64-bit, or 8-byte, words.

１バイト（オクテト）は８ビットで、４ビットはニブルと呼ぶ。最新の PC は６４ビット（８バイト）の文字を使う。

2.2.2 Binary Information Source Coding; ２元情報源符号化

Code words consist only of 0s and 1s.

符号語は０と１で構成される。

Contrast unsigned ternary (0..2), octal (0..7), hexadecimal (0..F).

対照的に、ternary 3 進法 (0..2)、octal 8 進法 (0..7)、hexadecimal 16 進法 (0..F)。

−	+
↓	↑
下	上
negative	positive
0	1
X	O
disable	enable
false (F)	true (T)
inhibit	assert
low (LO)	high (HI)
no; いいえ	yes; はい、ええ
off	on
reset	set
clear	load
not	so
contradictory	affirmative
'taint; ない	'tis; ある

表 2.2: Binary (2-state) Name Pairs; 一組の状態を表す言葉の例

2.2.3 Code, Coding, Code Word; 符号(ふごう)、符号化(ふごうか)、符号語(ふごうご)

In information theory, the interpretation (meaning) doesn't matter, just the number (and frequency) of the source symbols.

情報理論において、記号に意味はなく、その記号に対する値のみが重要である。

Example of a code (like a dictionary):

コードの例（辞書に似ている）:

N	北	00
S	南	01
E	東	10
W	西	11

2.2.4 Code Length; 符号長(ふごうちょう)

2.2.5 Mean Code Length; 平均符号長(へいきんふごうちょう)

Expected value of the code length. (See the bottom row of Table 2.3.)

符号長の期待値(きたいち)。（表 2.3 の下）

2.2.6 Decoding; 復号(ふくごう)

Interpreting an encoding, for example by looking up a code word in a dictionary (code book).

エンコードして暗号化したものを復元するときには、交換表を用いて復元する。

2.2.7 Uniquely Decodability; 一意的に復号可能

For each code word, there is a unique interpretation (non-singular).

それぞれの複合文字において、それぞれの解釈方法が存在する。

2.2.8 (Examples of) Various Coding Techniques; 各種の符号の例

2.2.8.1 Variable-length Code; 可変長符号

Code length is different across code words, like C_2–C_8 (Table 2.3).

符号語によって符号長は異なる (表 2.3)

Examples of fixed-length codes: ASCII (1-byte), EUC, JIS, S-JIS, Unicode (all 2-byte), mobile phone numbers.
Example of variable length codes: Japanese land phone numbers.

固定長符合の例：ASCII（１バイト）、EUC, JIS, S-JIS、Unicode（2 バイト）、携帯電話番号。
可変長符合の例：日本の電話番号。

2.2.8.2 Comma Code; コンマ符号

Has separator between code words (except possibly for longest).

符号の切れ目を表す。

2.2.8.3 Instantaneous Code; 瞬時符号

When expressed as a code tree, the code words are at the leaves.

符号木で表現した時、符号語が葉に割り当てられる。

For example of non-instantaneous code, consider *keitai* ("thumb-typing") number → kana on a numberpad:

非瞬時符号の例としては、携帯での仮名のタイピングが挙げられる。

2	か
22	き
222	く

To enter "きむら", one can simply press "227779", but to enter "きく [ち]", one can't simply enter "22222", since that is ambiguous; one must wait for modal time-out between "22" and "222" or use an arrow key to explicitly move to the next letter.

携帯電話で「きむら」と入力するためには単純に「227779」と押せば良いが、「きく [ち]」と入力するために「22222」と入力する事はできない。この曖昧さの為、「22」と「222」の間でタイムアウトを待つか、右キーを押して次の文字に進まなければならない。

2.2.8.4 Singular Code; 特異符号

Different codes are assigned the same word.

異なる記号に同一の符号語が割り当てられる

表 2.3: Examples of Codes; 情報源符号の例 (例題 1: 符号 C_1–C_6 [大 93, p. 17], 例題 2: 符号 C_7, C_8 [大 93, p. 29])

symbol; 記号	probability; 確率	code; 符号							
		C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
A	0.4	00	1	1	1	0	0	1	1
B	0.3	01	01	01	00	1	0	10	10
C	0.2	10	001	001	010	10	10	100	100
D	0.1	11	0001	000	011	01	01	1000	000
mean code length; 平均符号長		2.	2.	1.9	1.9	1.3	1.3	2.	1.9

2.2.9 (Information) Source Coding; 情報源符号

C_1**:** fixed length, but fragile (recovery impossible if letters drop out); 固定長符合けど壊れやすい（文字が欠落した場合、復元不可能）

C_2**:** comma code— robust (recovery possible if letters drop out); コンマ符合 堅牢（文字が欠落した場合、復元可能）

variable-length code; 可変長符号 (非等長符号)

C_3**:** tighter comma code (assuming maximum length); より良いコンマ符合（最大長を仮定している）

Uses '1' as comma (not needed if length reaches 3 bits); 「1」をコンマとして使う（符合長が 3 になったら必要なし）

C_3**,** C_4**:** also (variable-length) instantaneous codes, decodable without look-ahead; （可変長の）瞬時符合

C_5**:** not singular, but decoding impossible; 一意的に復号化不可能

$0\ 1\ 0 \Rightarrow \overset{D}{0}1\ \overset{A}{0}$ or $\overset{A}{0}\ \overset{C}{1}0$

C_6**:** singular code (so decoding impossible immediately); 特異符号

$0 \Rightarrow \overset{A}{0}$ or $\overset{B}{0}$

C_7**:** C_2 backwards; C_2 の逆. Uniquely decodable, but not instantaneous (since some words are prefixes of others), so need "look ahead."; 一意復号可能ではあるが、ある語の先頭部分が他の語の先頭部分と同じであるため瞬時符号ではない。そのため、最後まで見て判断する必要がある。

C_8**:** C_3 backwards; C_3 の逆. Note that while C_7 needed only a single bit look-ahead, C_8 requires an arbitrary length buffer. Consider, for instance, a received bit stream consisting of a one followed by n zeros: $1\overbrace{0000000000}^{n}$. If $n \bmod 3 \equiv 0$, the decoding is AD^*; if $n \bmod 3 \equiv 1$, the decoding is AC^*; but if $n \bmod 3 \equiv 2$, the decoding is AB^*. Since the string of zeros can

be of any length, the decoder (of this uniquely decodable but not instantensous code) needs an infitinte look-ahead.

2.3 Instantaneous Codes; 瞬時符号

2.3.1 Instantaneous Codes (Expressed by Code Trees); 瞬時符号 (符号の木による表現)

Graph Definition　　グラフの定義

2.3.1.1 Tree; 木

Usually drawn inverted (upside-down), with root at top.　　一般的に root（根）が一番上に描かれる

2.3.1.2 Node, Vertex (plural: nodes, vertices); 節点

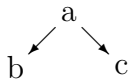
$$V = \{v_1, v_2, \ldots, v_s\} \tag{2.1}$$

2.3.1.3 Edge, Branch; 枝

$$\begin{aligned} e_n &= (v_{n_1}, v_{n_2}) \\ v_{n_i} &\in V, \quad i = 1, 2 \\ E &= \{e_1, e_2, \ldots, e_s\} \\ G &= [V, E] \end{aligned}$$

Example:

$$\begin{aligned} V &= \{a, b, c\} \\ E &= \{(a, b), (a, c)\} \\ G &= [\{a, b, c\}, \{(a, b), (a, c)\}] \end{aligned}$$



2.3.1.4 Path; パス

A path traverses (goes through) the nodes along the edges.　　枝でノード間の経路

2.3.1.5 Connected Graph; 連結(れんけつ)グラフ

A connected graph has a path through all the nodes.

全(すべ)のノード間にはパスが存在する

2.3.1.6 Loop; 閉路(へいろ)

A loop is a path with the same beginning and ending point.

始点(してん)と終点(しゅうてん)が同一(どういつ)であるパスが存在(そんざい)するグラフ

2.3.2 Representation as Coding Tree; 符号の木による表現(ひょうげん)

- coding tree; 符号(ふごう)の木
- binary tree; 2分木(にぶんぎ)In human families, two parents can spawn any number of children, but in binary trees, each (single) parent has 0, 1, or 2 children.
- root; 根(ね)
- leaf; 葉(は)
- level, length of branch; 次数(じすう), 枝(えだ)の長さ
- instantaneous codes; 瞬時符号(しゅんじふごう)

  All code words are assigned to leaves

  符号語はすべて葉に割り当てられる

  $$C_1, C_2, C_3, C_4$$

- non-instantaneous codes; 非瞬時(ひしゅんじ)符号

  Some code words are at non-leaf nodes.

  符合語が 葉ノードでないものもある

  $$C_5, C_6, C_7, C_8$$

- decoding tree; 復号木(ふくごうき)

  Decoding is possible using a decoding tree [大 93, p. 23].

  符号木を用いて復号化することが可能 [大 93, p. 23]。

Relation	Notation	Explanation	Example
sufficiency	A ⇒ B	if A then B (B or not A)	snow ⇒ cold (but cold ⇏ snow)
		A is sufficient (suffices) for B	snow suffices for cold
contrapositive	¬B ⇒ ¬A	if not B then not A (¬A ∥ ¬¬B)	if not cold, then no snow
necessity	A ⇐ B	if B then A (A or not B)	cold ⇐ snow
		A is necessary for B	cold is necessary for snow
equivalence,	A ≡ B	if A then B and if B then A	even ≡ not odd
mutual implication	A⇒B and B⇒A A⇔B	"if and only if" iff A then B (and vice versa)	

表 2.4: Implication; $A \xrightarrow[\text{"is necessary for"}]{\text{"suffices for"}} B$. For example, if snow implies cold, then cold is necessary for snow, and snow is sufficient for cold. 例えば、「雪」⇒「冷たい」という命題が与えられた時、「冷たい」は「雪」の必要条件であり、「雪」は「冷たい」の十分条件である。

表 2.5: $X + \overline{Y}$, $Y \Rightarrow X$ (necessity)

X\Y	0	1
0	1	0
1	1	1

表 2.6: $(X \oplus Y, X \neq Y)$: XOR— exclusive or (inequality, symmetric difference), same as CNOT (controlled not, used in quantum computing)

X\Y	0	1
0	0	1
1	1	0

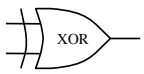


表 2.7: IMPLIES (sufficiency): $X \Rightarrow Y$ $(Y + \overline{X})$

X\Y	0	1
0	1	1
1	0	1

表 2.8: $X \equiv Y$: EQUIV, XNOR (equality, coincidence, mutual implication, iff [if and only if], necessity and sufficiency)

X\Y	0	1
0	1	0
1	0	1

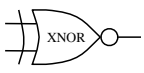


表 2.9: Dyadic binary functions; 2 変数の 2 進関数の組

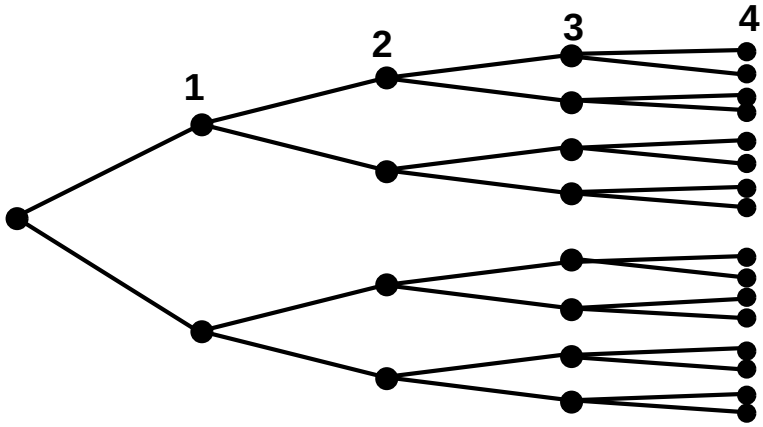


図 2.1: Completely Balanced Binary Tree

2.4 Kraft Inequality; クラフトの不等式;

Case of complete binary tree (l_n: level; N: 完全 2 分木 (l_n: 次数, N: 葉) の場合 leaves)

$$2^{-l_1} + 2^{-l_2} + \ldots + 2^{-l_N} = 1 \tag{2.2}$$

For Figure 2.1, 16 $2^{-4} = 1 \leq 1$ $\surd$.

Complete binary tree: completely balanced binary tree, possibly with some number of leaves at lowest level deleted.

完全 2 分木：次数の低い葉がいくつか削除されている 2 分木

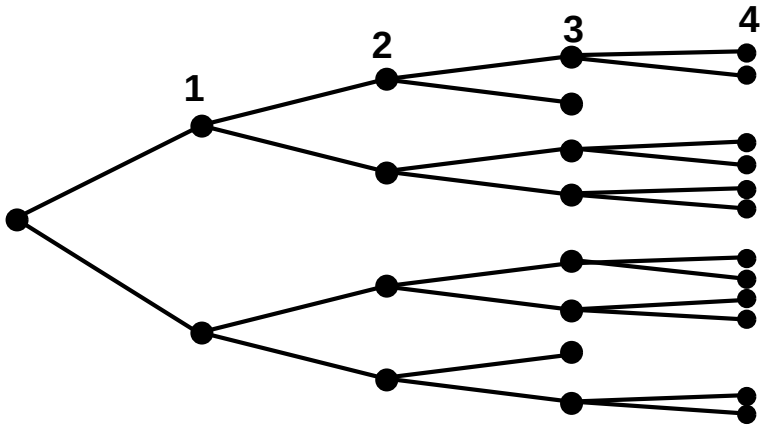


図 2.2: Complete Binary Subtree; 完全 2 分木の部分木

For Figure 2.2, 2 $2^{-3} + 12\ 2^{-4} = \frac{2}{8} + \frac{12}{16} = 1 \leq 1$ $\surd$.

Kraft Inequality; クラフトの不等式 (Theorem 2.5.1; 定理 2.5.1)

Satisfying the Kraft Inequality is necessary and sufficient condition for the existence of an instantaneous code with code words of lengths l_i.

クラフトの不等式は長さ l_i の符合語で成る瞬時符合が存在するための必要十分条件である。

$$\text{Kraft Sum K} \triangleq \sum_i 2^{-l_i} = 2^{-l_1} + 2^{-l_2} + \ldots + 2^{-l_M} \leq 1$$

Collection of integer lengths satisfying Kraft Inequality

正の整数 $l_1, l_2, \ldots, l_M$ がクラフトの不等式を満たす

⇳ (iff: if and only if [sufficient and necessary])

An instantaneous code of those lengths can be constructed.

瞬時符号が作れる。

In physical spaces (time, energy, volume), bigger is usually more costly...

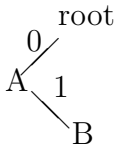
Long messages are cheap in code space; it's the short ones that eat up the code-space real-estate.

長いメッセージは変形する余地が少ない。短いものが浪費している。

Note that not every (uniquely decodable) code that satisfys the Kraft Inequality is instantaneous. A particular code's lengths might satisfy the Kraft Inequality but the code might still not be instantaneous. However, an instantaneous code of the given lengths can be found. For example,

クラフトの不等式を満たす全ての符号が瞬間的ではない。クラフトの不等式をを満たす符号でも瞬間的とは限らない。しかし、瞬間的に適応できるコードは以下の方法でみつけられる。

symbol	code	length	$(\frac{1}{2})^{\text{length}}$
A	0	1	$\frac{1}{2}$
B	01	2	$\frac{1}{4}$
Σ			$\frac{3}{4}$



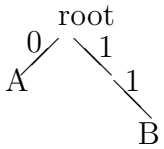
satisfies the Kraft inequality but is *not* instantaneous:

これはクラフトの不等式を満たすが、瞬時では**ない**。

0	0	$\mapsto$	A A
0	1	$\mapsto$	B

However, another code of the given code lengths

symbol	code	length	$(\frac{1}{2})^{\text{length}}$
A	0	1	$\frac{1}{2}$
B	11	2	$\frac{1}{4}$
Σ			$\frac{3}{4}$



is instantaneous.

これは瞬時で**ある**。

2.5 Code Space; 符号空間

Divide the area of a rectangle (as in Figure 2.4)

面積 1 の長方形の領域に符号語を割り振る (図2.4 参照)

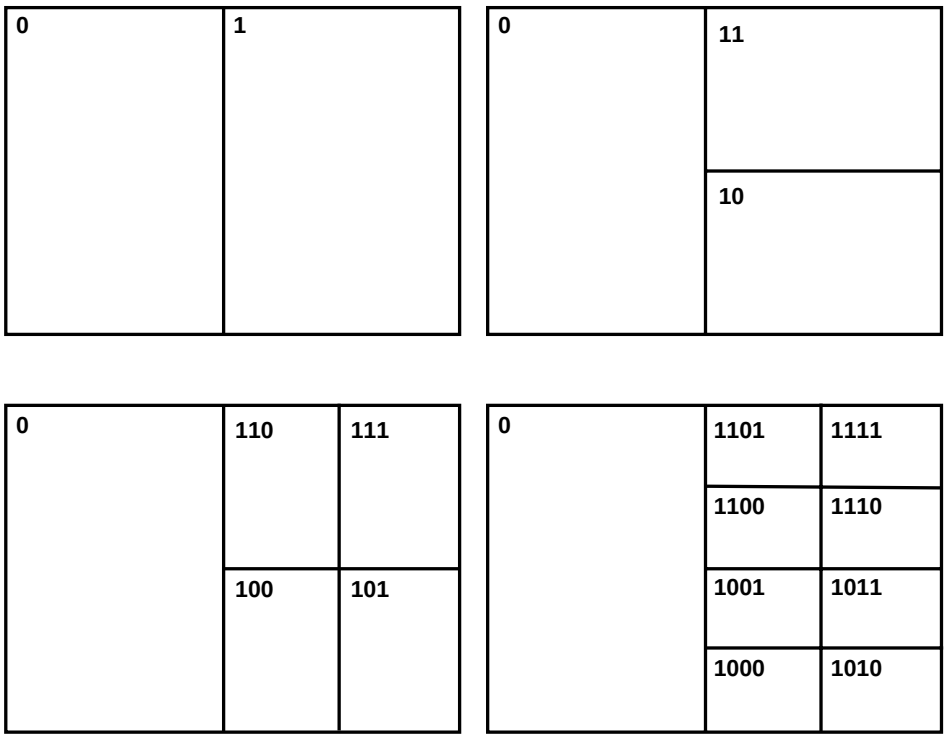


図 2.3: Partitioning Rectangle; 長方形分割

List the (integral) lengths in order.

正の整数 $l_1, l_2, \ldots, l_M$ を並べ代えて以下のようにする。

$$l_1 \leq l_2 \leq \ldots \leq l_M$$

ステップ 1: A0 版の紙を Al_1 版の紙に切る。この時、 2^{l_1} 枚得られる。これから 1 枚取り出す。$n = 1$ とする。

ステップ 2: $l_{n+1} = l_n$ を調べ、等しければ残りの紙から 1 枚取り出す。$l_{n+1} > l_n$ であれば、残された Al_n の紙を全て Al_{n+1} に切り、 1 枚取り出す。

ステップ 3: $n+1 = M$ であればアルゴリズムを終了、そうでなければ、$n = n+1$ としてステップ 2 に戻る。

The harmonic series $\sum_{k=1}^{\infty} \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \ldots$ diverges (goes to infinity), but the geometric series $\sum_{k=1}^{\infty} 2^{-k} = \sum(\frac{1}{2})^k = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \ldots$ converges (approaches a constant, in this case unity).

調和級数 $\sum_{k=1}^{\infty} \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \ldots$ は無限に発散するが、等比級数 $\sum_{k=1}^{\infty} 2^{-k} = \sum(\frac{1}{2})^k = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \ldots$ は一定の値に収束する。

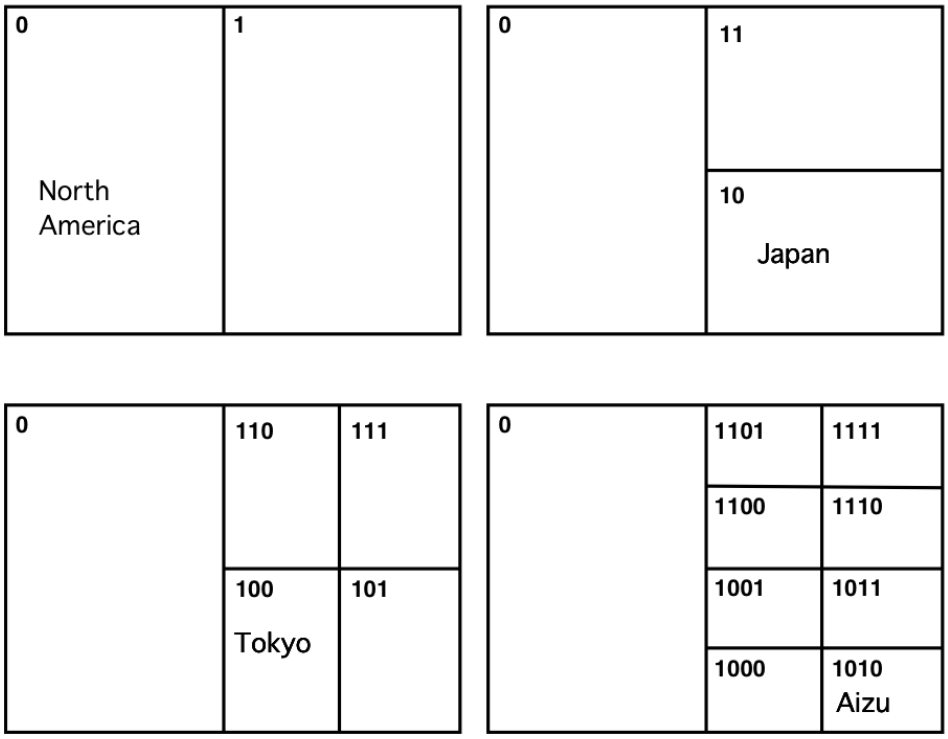


図 2.4: Variable Length Phone Number Prefixes

2.6 Uniquely Decodable; 一意的に復号可能な符号

C_7, C_8: non-instantaneous codes (reverses of C_2, C_3); 非瞬時符号 [大 93, p. 29]

A code is non-instantaneous if it has code words with codes assigned to words not at the leaves of its tree.

符号木において葉ではなく符号語がノードに割り当てられると非瞬時符号となる。

Problem; 問い

Make C_7, C_8 code trees, calculate average code length, and confirm the Kraft-McMillan Inequality. ([大 93, p. 39: 2.6])

C_7, C_8 の符号木を作成し、平均符号長を計算し、クラフトーマクミランの不等式を証明せよ。([大 93, p. 39: 2.6])

Uniquely decodable codes, even if non-instantaneous, satisfy the Kraft Inequality.

瞬時符号でない一意的に復号可能な符号においてもクラフトの不等式を満たしている。

S^2: 2^{nd} extension of information source

S^2: 2 次の拡大情報源

$$S^2 = \begin{pmatrix} a_1a_1 & a_1a_2 & \dots & a_Ma_M \\ p_1p_1 & p_1p_2 & \dots & p_Mp_M \end{pmatrix}$$

S^n: n^{th} extension of an information source

S^n: n 次の拡大情報源

Wordlength (bits)	Chunk	Extension	Range Size	Unsigned Range (unipolar)	Signed Range (bipolar)	Examples
1	bit	binary	2	[0,1]		Boolean
2		quadary	4	[0,3]	[-2,1]	
3		octal	8	[0,7]	[-4,3]	Unix file permissions r/w/x, 0...
4	nibble or nybble	hexadecimal	16	[0,15], [0,F]	[-8,7]	0x...
7			128	[0,127]	[-64,63]	ASCII
8	byte or octet		256	[0,255]	[-128,127]	extended ASCII, MIDI
10			1024 = 1 K	[0,1023]	[-512,511]	
16	2 bytes		65536 = 64 K	[0,65535]	[-32768,32767]	EUC, JIS, S-JIS, Unicode
20			1,048,576 = 1 M	[0,1048575]	[-524288,524287]	
32	quadlet: 4 bytes		address space: 4 GiB = 4,294,967,296			
64	8 bytes		address space: 16 EiB ≈ 18 EB (exabytes)			modern computer words
n		n-ary	2^n	$[0,2^n-1]$	$[-2^{n-1}, 2^{n-1}-1]$	

表 2.10: Binary Representations and Extensions; 2 進法表現 と 拡張

表 2.11: Second Extension of C_7; C_7 を用いた 2 次の拡大情報源 S^2 の符号

Symbol Stream	Probability	Encoding	Symbol Stream	Probability	Encoding
AA	$.4 \times .4$	11	CA	$.2 \times .4$	1001
AB	$.4 \times .3$	110	CB	$.2 \times .3$	10010
AC	$.4 \times .2$	1100	CC	$.2 \times .2$	100100
AD	$.4 \times .1$	11000	CD	$.2 \times .1$	1001000
BA	$.3 \times .4$	101	DA	$.1 \times .4$	10001
BB	$.3 \times .3$	1010	DB	$.1 \times .3$	100010
BC	$.3 \times .2$	10100	DC	$.1 \times .2$	1000100
BD	$.3 \times .1$	101000	DD	$.1 \times .1$	10001000

$$S^n = \begin{pmatrix} a_1 a_1 \dots a_1 & a_1 a_1 \dots a_1 a_2 & \dots & a_M a_M \dots a_M \\ p_1 p_1 \dots p_1 & p_1 p_1 \dots p_1 p_2 & \dots & p_M p_M \dots p_M \end{pmatrix}$$

For example, octal (0, 1, 2, 3, 4, 5, 6, 7) is represented as the third extension of binary, 2^3, and hexadecimal (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F) is the fourth extension, 2^4.

例えば 8 進の (0, 1, 2, 3, 4, 5, 6, 7) は 2^3 で、16 進の (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F) は 2^4。

Characteristic 1; 性質 1

If a code is uniquely decodable, its n$^{\text{th}}$ extension is also non-singular. That is, unique decodability (which instantaneous decodability implies) means that no two concatenations can be the same, even for different extensions.

ある符号が一意復号可能である時、任意の n 次の拡大符号は非特異である。つまり、一意復号可能であるということは、たとえ異なった次数であっても 2 つの連結が同じになることはない、ということである。

2.6.1 McMillan Inequality; マクミランの不等式

Given a uniquely decodable binary code S of lengths l_m: $l_1, l_2, \dots, l_M$, let l_m's maximum value be k. Then,

一意的に複合可能な S に与えられた 2 次符号の符号長を $l_1, l_2, \dots, l_M$ とし、l_M の最大値を k とする。従って、

$$1 \le l_m \le k \quad (\overset{\text{for all}}{\forall} \; m = 1, \dots, M) \tag{2.3}$$

Consider the n^{th} extension of the Kraft sum: クラフトの和の n 時の拡張を考える。

$$(2^{-l_1} + 2^{-l_2} + \dots + 2^{-l_M})^n = \sum_{i_1=1}^{M} \sum_{i_2=1}^{M} \dots \sum_{i_n=1}^{M} 2^{-(l_{i_1} + l_{i_2} + \dots + l_{i_n})} \tag{2.4}$$

As a confirmatory example, returning to the example towards the end of § 2.4, for M=2 words, of length $l_1 = 1, l_2 = 2$ (so maximum word length $k = 2$), and $n = 2$ (2^{nd} extension),

確認のための例として、2.4 章の終りのあたりの例に戻り、M=2, 符号語の長さ $l_1 = 1, l_2 = 2$ ($k = 2$ が最大値) で $n = 2$(2 次拡張) の時、

$$
\begin{aligned}
(2^{-1} + 2^{-2})^2 &\stackrel{?}{=} \sum_{i_1=1}^{2}\sum_{i_2=1}^{2} 2^{-(l_{i_1}+l_{i_2})} \\
&= (\overbrace{2^{-(l_1+l_1)}}^{AA} + \overbrace{2^{-(l_1+l_2)}}^{AB}) + (\overbrace{2^{-(l_2+l_1)}}^{BA} + \overbrace{2^{-(l_2+l_2)}}^{BB}) \\
(\frac{1}{2} + \frac{1}{4})^2 &= 2^{-(1+1)} + 2^{-(1+2)} + 2^{-(2+1)} + 2^{-(2+2)} \\
(\frac{1}{2} + \frac{1}{4})(\frac{1}{2} + \frac{1}{4}) &= 2^{-2} + 2^{-3} + 2^{-3} + 2^{-4} \\
\frac{1}{4} + \frac{1}{8} + \frac{1}{8} + \frac{1}{16} &= \frac{1}{4} + \frac{1}{8} + \frac{1}{8} + \frac{1}{16} \quad (= \frac{9}{16}) \surd
\end{aligned}
$$

The length l of the n^{th} extension is less than or equal to kn, n times the source code word's maximum length k (not to be confused with the Kraft sum 'K'). That is,

とすると、n 次拡張の長さ l は kn 以下になる。情報源符合語の最大長 k(クラフトの和「K」と混同しない様、注意してください。) の n 倍である。つまり、

$$n \leq l \leq kn$$

$$(2^{-l_1} + 2^{-l_2} + \ldots + 2^{-l_M})^n = \sum_{l=0}^{\infty} s_l 2^{-l} = \sum_{l=n}^{kn} s_l 2^{-l}$$

where s_l is the number of code words of length l.

と表すと s_l は符号長が l となる符号語の数であり

Using the example above, where lengths of the 2^{nd} extension are $\{2, 3, 3, 4\}$,

上記の例を使うと、2 次拡張の長さが {2,3,3,4} の場合、

$$
\begin{aligned}
(2^{-1} + 2^{-2})^2 &= \sum_{l=2}^{4} s_l 2^{-l} \\
&= 1 \cdot 2^{-2} + 2 \cdot 2^{-3} + 1 \cdot 2^{-4}
\end{aligned}
$$

S^n is uniquely decodable (and therefore non-singular), so the total number of words of each length must be no greater than the number of nodes of the binary coding tree at that level:

S^n は一意的に複合可能なので（同時に非特異でもある）合計単語数はそのレベルにおいての 2 進樹符号木のノード数を超えられない。

$$s_l \leq 2^l$$

である。

Therefore,

従って

$$
\begin{aligned}
(2^{-l_1} + 2^{-l_2} + \ldots + 2^{-l_M})^n &\leq \sum_{l=n}^{kn} 2^l 2^{-l} \\
&= \overbrace{\underbrace{1}_{l=n} + 1 + \cdots + \underbrace{1}_{l=kn}}^{kn-n+1} \\
&= kn - n + 1 \\
&\leq kn \\
2^{-l_1} + 2^{-l_2} + \ldots + 2^{-l_M} &\leq (kn)^{\frac{1}{n}}
\end{aligned}
$$

Since exponential functions grow faster than linear functions (as seen in Figures 2.5 and 2.6),

指数関数の方が一次関数より増加が早いので (図 2.5, 図 2.6)

$$\lim_{n\to\infty} (kn)^{1/n} = 1 \tag{2.5}$$

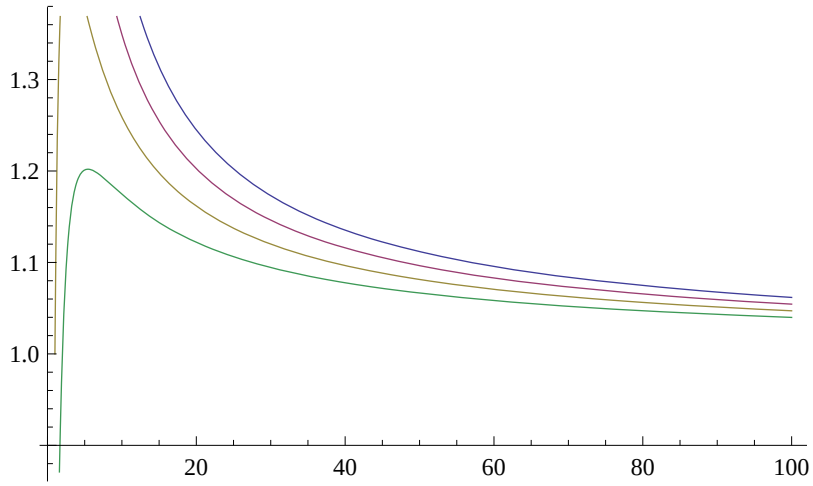


図 2.5: $(4n)^{\frac{1}{n}}$, $(2n)^{\frac{1}{n}}$, $n^{\frac{1}{n}}$, $\frac{n}{2}^{\frac{1}{n}}$

左辺は n に無関係な定数で、右辺は $\to 1 (n \to \infty)$ であるので、以下の不等式が成り立つ。

If K were > 1, an extension would explode beyond linear bounds, and since that boundary is not trespassed, K must therefore be ≤ 1.

K が 1 であるならば、拡張は線形境界の向こうで急増する。しかし、その境界線が交わらないので、$K \leq 1$ でなければならない。

McMillan Inequality; マクミランの不等式 (1965)

Uniquely decodable code lengths $l_1, l_2, \ldots, l_M$ also satisfy the Kraft Inequality

一意的に復号可能な符号の符号長を $l_1, l_2, \ldots, l_M$ は以下の不等式を満たす。

$$2^{-l_1} + 2^{-l_2} + \ldots + 2^{-l_M} \leq 1$$

Attention: We showed earlier that the Kraft Inequality is satisfied for instantaneous codes. The inequality also applies to uniquely decodable codes through the McMillan Inequality.

注意：瞬時符号であれば上の不等式を満たすことはクラフトの不等式として示した。マクミランの不等式は瞬時符号を含む一意的に復号可能な符号にクラフトに対して不等式が成り立つことを拡張したことになる。

complexity	function	example
constant; 定数	$y \sim c$	adding element to stack or queue
almost constant	$y \sim \overbrace{\log\log\ldots\log}^{n} x$	inverse Ackerman function
logarithmic; 対数	$y \sim \log x$	level [dB] $= 10 \log \frac{I}{I_0}$; extracting minimum from binary heap
sublinear	$y \sim x^{\frac{1}{2}}$	
linear; 線形	$y \sim cx$	Einstein: $E = \mathrm{m}\ c^2$ (c^2 is constant); traversing binary search tree
"linearithmic" (a.k.a. "pseudolinear")	$y \sim x \log x$	entropy: $H(P) = -\sum_i p_i \log p_i$; QuickSort; Dijkstra shortest path
polynomial; 多項式	$y \sim \sum_i c_i x^i$	a.k.a. **algebraic**; $ax^2 + bx + c = 0 \Rightarrow x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$
power; 乗数	$y \sim x^c$	$\log y = c \log x \Rightarrow$ linear curve on log-log plot
quadratic; 2 次の	$y \sim x^2$	parabola: $y = x^2$; adding two matrices
cubic; 3 次の	$y \sim x^3$	
subexponential	$y \sim c^{\sqrt{x}}$	
exponential; 指数	$y \sim c^x$	a.k.a. **geometric**; descendents = (original ancestors)(offspring$^{\text{generation}}$) Moore's Law: computing power $= 2^{\text{time}/1.5\,\text{years}}$
factorial; 階上	$y \sim x! \approx x^x e^{-x} \sqrt{2\pi x}$	a.k.a. combinatorial; $n! = n(n-1)!$
"superexponential"	$y \sim c_0^{\cdot^{c_i}{}^{\cdot^{c_{n-1}{}^x}}}$	Ackerman function
doubly exponential	$y \sim c_0^{c_1{}^x}$	
triply exponential	$y \sim c_0^{c_1{}^{c_2{}^x}}$	
quadruply exponential	$y \sim c_0^{c_1{}^{c_2{}^{c_3{}^x}}}$	
⋮	⋮	

表 2.12: Generic Function Complexity (Growth); 関数の増大

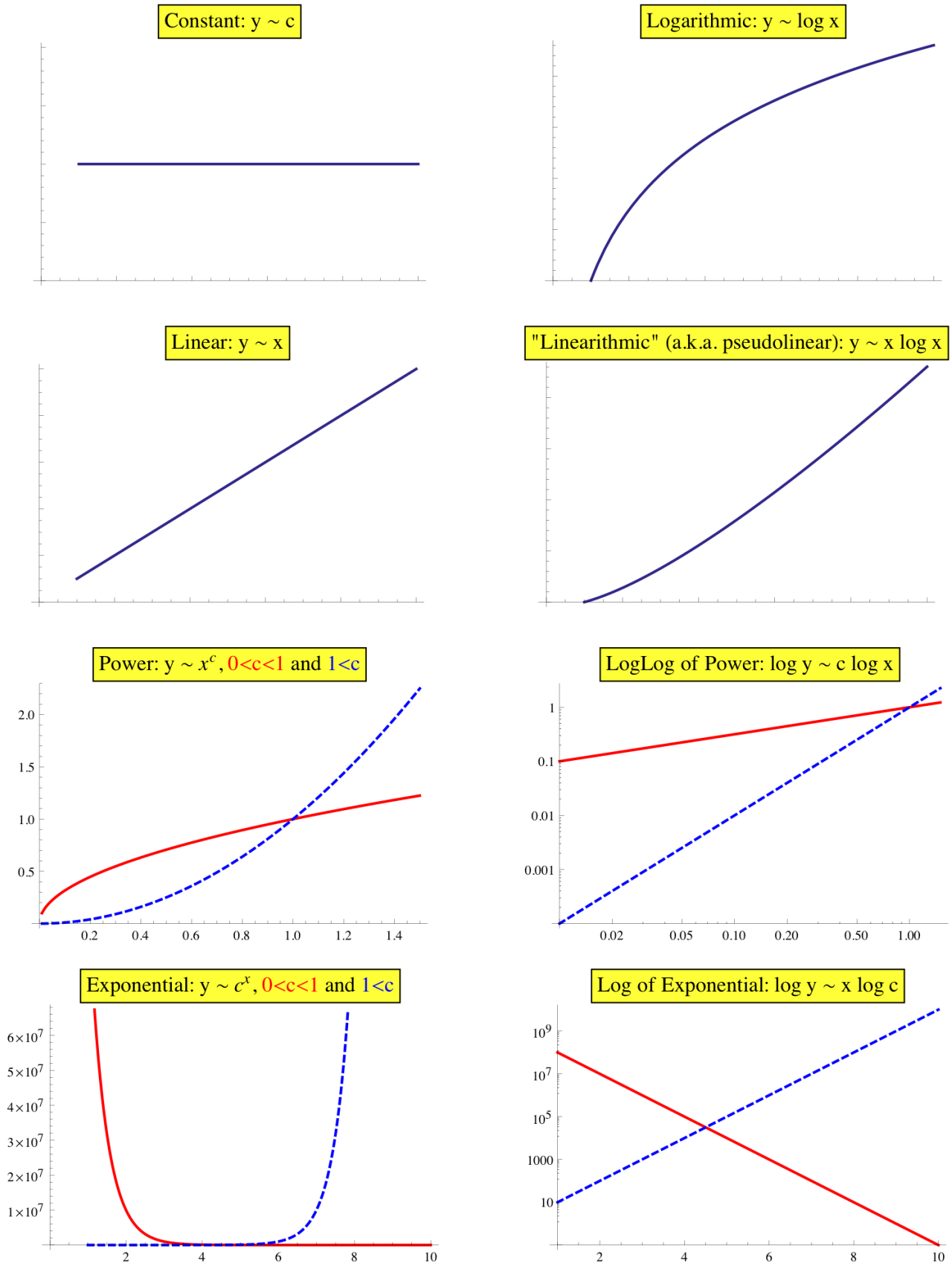


図 2.6: Generic Functions; 関数

2.7 Mean Code Length and Entropy; 平均符号長とエントロピー

S: Memoryless stationary discrete information source　　S: 無記憶定常離散的情報源

C: Uniquely decodable code　　C: 一意的に復号可能な符号

$$S = \begin{pmatrix} a_1 & a_2 & \dots & a_M \\ p_1 & p_2 & \dots & p_M \end{pmatrix}$$

L: mean code length　　L: 平均符号長

$$\begin{aligned} L &= p_1 l_1 + p_2 l_2 + \ldots + p_M l_M && \text{(2.6a)} \\ &= -\sum_{n=1}^{M} p_n \log_2 q_n && \text{(2.6b)} \end{aligned}$$

where

$$q_n = 2^{-l_n} = (\frac{1}{2})^{l_n} \quad n = 1, 2, \ldots, M \tag{2.7}$$

From the Kraft-McMillan Inequality,　　クラフトーマクミランの不等式より

$$q_1 + q_2 + \ldots + q_M \leq 1 \tag{2.8}$$

we can prove this inequality:　　このとき以下の不等式が成り立つ。

$$L \geq -\sum_{n=1}^{M} p_n \log_2 p_n. \tag{2.9}$$

2.7.1 Logarithms; 対数

Recall the Taylor series for a function $f(x)$ that is infinitely differentiable in the neighborhood of a number a:

a の近傍で無限回微分可能である関数 $f(x)$ におけるテイラー級数を思い出す。

$$\begin{aligned} f(x) &= f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f^{(3)}(a)}{3!}(x-a)^3 + \ldots && \text{(2.10a)} \\ &= \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n && \text{(2.10b)} \end{aligned}$$

This can be applied (as a Maclauren Series, in the neighborhood of 0) to the exponential function (since the derivative of e^x is also e^x and $e^0 = 1$):

e^x の導関数が e^x であり、かつ $e^0 = 1$ であることより、0 近傍の時、マクローリン級数として指数関数で応用される。

$$\begin{aligned} e^x &= e^0 + \frac{e^0}{1!}(x-0) + \frac{e^0}{2!}(x-0)^2 + \frac{e^0}{3!}(x-0)^3 + \dots && (2.11a)\\ &= 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots && (2.11b)\\ &= \sum_{n=0}^{\infty} \frac{x^n}{n!} && (2.11c) \end{aligned}$$

$$\begin{aligned} \frac{d}{dx}(e^x) &= 0 + 1 + \frac{2x}{2!} + \frac{3x^2}{3!} + \dots && (2.12a)\\ &= 1 + \frac{x}{1} + \frac{x^2}{2!} + \dots && (2.12b)\\ &= e^x && (2.12c) \end{aligned}$$

$$\begin{aligned} e^1 &= 1 + 1 + \frac{1}{2} + \frac{1}{6} + \frac{1}{24} + \dots && (2.13a)\\ &= 2.71828\dots && (2.13b) \end{aligned}$$

$$\begin{aligned} e &= \lim_{n\to\infty} \left(1 + \frac{1}{n}\right)^n && (2.14a)\\ &= \lim_{n\to 0} (1+n)^{\frac{1}{n}} && (2.14b)\\ &= 2 + \cfrac{1}{1 + \cfrac{1}{2 + \cfrac{2}{3 + \cfrac{3}{4 + \cfrac{4}{5 + \cfrac{5}{6 + \cfrac{6}{7 + \cfrac{7}{8 + \dots}}}}}}}} && (2.14c) \end{aligned}$$

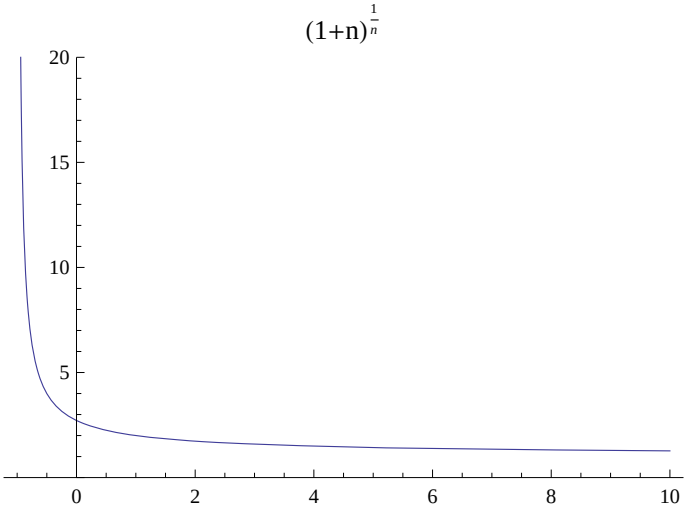


図 2.7: $e\left(= \lim_{n\to\infty}(1+\frac{1}{n})^n\right) = \lim_{n\to 0}(1+n)^{\frac{1}{n}}$

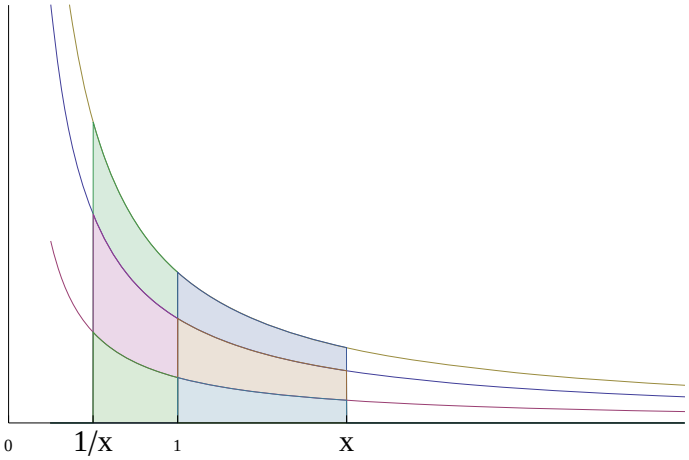


図 2.8: $\overset{\sim 1.4}{\lg_2} e\cdot(\frac{1}{t})$, $\frac{1}{t}$, $\overset{\sim .43}{\log_{10}} e\cdot(\frac{1}{t})$: $\log_e(x) \equiv \ln(x) \stackrel{\Delta}{=} \int_1^x(\frac{1}{t})dt; \log_{\{2,e,10\}}\frac{1}{x} = \log x^{-1} = -\log x; \int_1^{1/x}(\frac{1}{t})dt = -\int_{1/x}^1(\frac{1}{t})dt = -\int_1^x(\frac{1}{t})dt; \log(\frac{1}{2}) = -\log(2)$

system	base	units	notation	example
binary	2	bit	lb, lg, or lt	$H(X) = -\sum_i p_i \lg p_i$
natural	e	nat	ln	$\ln x = \int_1^x \frac{1}{t}dt$
common	10	ban, hartley (Hart), dit (**d**ecimal dig**it**)	log	level [dB] $= 10 \log_{10}\frac{I}{I_0}$

表 2.13: Logarithmic Bases; 対数の底

Table 2.13 shows conventions used when the logarithmic base isn't implicit. But since digital computers use two-state logic, unless otherwise stated, we will usually use binary logarithms, written just "log," with an implicit base 2.

表2.13 わかるようにデジタルコンピュータは 2 進数を使用しているので、対数の底が述べられていないときは、底が 2 の対数を使う。対数の底が明示されていない時の慣習を表2.8 で示す。しかし、デジタルコンピュータでは 2 進数を使用しているので、対数の底が述べられていない時、"log "とだけ書き底は 2 を用いる。

Lemma; 補題

For $x > 0$, $\ln x \leq x - 1$.

$x > 0$ に対して $\log x \leq x - 1$(底は自然対数 e) を示せ。

Natural log is convex, so always below a straight line tangent to it (except at the point x=1 where they touch).

自然対数のグラフは上に凸(とつめん)である、つまり常に接線よりも下にある。(それらが接する x=1 の点は除く)

Since the second derivative is negative, the curve bends away from its tangent:

$$\frac{d}{dx}(\ln x) = \frac{1}{x} \tag{2.15a}$$

$$\frac{d^2}{dx^2}(\ln x) = \frac{d}{dx}\left(\frac{1}{x}\right) = -x^{-2} = -\frac{1}{x^2} \tag{2.15b}$$

Slope of $\log_e x$ at $x = 1$:

x=1 のときの $\log_e x$ の傾きは

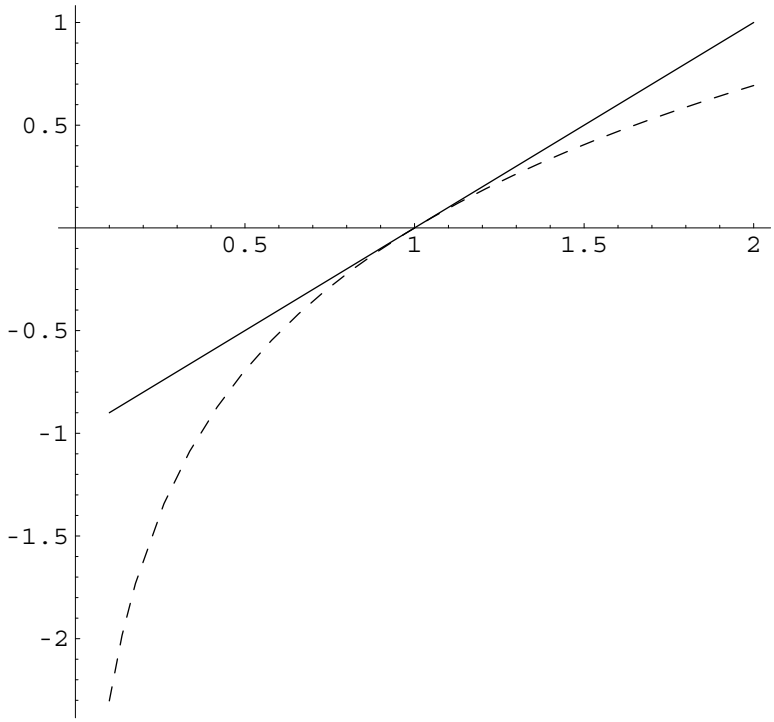


図 2.9: $\log_e = \ln$ (dashed) and $x - 1$ (solid)

$$\left.\frac{d(\log_e x)}{dx}\right|_{x=1} = \left.\frac{1}{x}\right|_{x=1} \tag{2.16a}$$

$$= 1 \tag{2.16b}$$

Gibbs' Theorem: The shortest mean length of a code is bounded by the entropy.

ギブス定理：符合の最小平均長はエントロピーに制限される。

$$L \geq -\sum_{n=1}^{M} p_n \log_2 p_n \tag{2.17}$$

Proof: Letting

証明:

$$x = \frac{q_n}{p_n} \geq 0$$

and applying above Lemma

として、補題の

$$\ln x \stackrel{\Delta}{=} \log_e x \leq x - 1 \tag{2.18}$$

yields

に代入すると、以下を得る。

$$\log_e \frac{q_n}{p_n} \leq \frac{q_n}{p_n} - 1$$

But the logarithmic base can be converted (by the chain rule for logarithms):

対数の底の変換公式を使うと左辺は

$$\log_{\mathbf{e}} \frac{q_n}{p_n} = (\log_{\mathbf{2}} \frac{q_n}{p_n}) / \log_2 e$$

e is greater than 2, and $\log_2 e \approx 1.4$, which is positive, so,

eは2より大きく、$\log_2 e \approx 1.4$で正であるので、

$$\begin{aligned} \log_2 \frac{q_n}{p_n} &\leq (\frac{q_n}{p_n} - 1) \log_2 e \\ p_n \log_2 \frac{q_n}{p_n} &\leq (q_n - p_n) \log_2 e \end{aligned}$$

従って和を求めると以下となる。

Summing over $n = 1..M$, where M is still the number of code words,

Mはコードの数であり、1からMまでを加算する。

$$\sum_{n=1}^{M} p_n \log_2 \frac{q_n}{p_n} \leq \sum_{n=1}^{M} (q_n - p_n) \log_2 e \tag{2.19}$$

But (by eqn. (2.6b)) the average code length is

しかし、(eqn. (2.6b))平均コード長は以下のとおりである。

$$L = \sum_n p_n l_n = -\sum_n p_n \log_2 q_n \tag{2.20}$$

where (following Eqn. 2.7)

$$q_n = 2^{-l_n}$$

and

$$\sum p_n = 1 \tag{2.21}$$

so, by eqn. (2.20),

$$\sum_n p_n \log_2 \frac{q_n}{p_n} = -L - \sum_n p_n \log_2 p_n$$

by eqn. (2.19)

$$\leq \left(\sum_n (q_n - p_n) \right) \log_2 e$$

$\sum_n p_n = 1$ (eqn. (2.21))

$$= (\sum_n q_n - 1) \log_2 e$$

uniquely decodable $\Rightarrow \sum_n q_n \leq 1$ (McMillan Inequality)

$$\leq 0$$

Therefore, since the code satisfys the Kraft-McMillan inequality,

クラフトーマクミランの式を満たすので

$$L \geq -\sum_{n=1}^{M} p_n \log_2 p_n$$

Entropy; エントロピー

The entropy of an information source is defined as

情報源 S のエントロピーは以下の式で定義される。

$$H(S) \triangleq -\sum_{n=1}^{M} p_n \log_2 p_n \tag{2.22}$$

Theorem; 定理

An arbitrary memoryless stationary discrete information source S that is uniquely decodable satisfys the following inequality:

任意(にんい)の無記憶定常離散的情報源 S の任意の一意的に復号可能な符号に対して、その平均符号長 L に関する以下の不等式が成り立つ。

$$L \geq H(S) \tag{2.23}$$

That is, L can't be shorter than $H(S)$: the mean code length is no shorter than the entropy.

すなわち、平均符号長Lはエントロピー $H(S)$ より短くできない。

2.7.2 Remarks; 注意

1. Since $0 \leq p_n \leq 1$,

1. $0 \leq p_n \leq 1$ であるので、

$$0 \leq H(S)$$

を得る。

2. $H(S)$ is at a maximum when $p_n = 1/M$. Therefore,

2. $H(S)$ が最大になるのは、$p_n = 1/M$ (一様分布) のときである。従って

$$0 \leq H(S) \leq \log_2 M$$

3. When $H(S) = 0$, there is exactly one n such that $p_n = 1$, with all other values of $p_n = 0$.

3. $H(S)$ が0のときには、あるひとつの n に対してのみ $p_n = 1$ となり、他の全ての値が $p_n = 0$ となる。

Thinking of a number line, it is possible to determine l_n, which the following inequality determines uniquely.

数直線を考えれば以下の不等式を満たす l_n を一意的に定めることが可能である。

$\exists$ positive integer l_n such that　　$\exists l_n$: 正の整数

$$0 \leq -\log_2 p_n \leq l_n < -\log_2 p_n + 1 \tag{2.24}$$

For example, for $p_n = \frac{1}{3}$, $\log_2 \frac{1}{3} \approx -1.6$ (for comparison, recall that $\log_2 e \approx 1.4$), so

$$l_n = \lceil 1.6 \rceil = 2 \leq 1.6 + 1 = 2.6$$
$$-l_n \leq \log_2 p_n$$

Exponentiating and summing over n　　指数をとって、n までの値を合計する

$$\sum_n 2^{-l_n} \leq \sum_n p_n = 1$$

Therefore (by Theorem 2.5.1), the Kraft Inequality is satisfied, and there exists an instantaneous code with code lengths l_n. Its average code length can be bounded using inequality eqn. (2.24) above.

従って クラフト 不等式が成り立つので、定理 2.5.1 より符号長を l_n とする瞬時符号が存在することになる。その平均符合長は上記の不等式で制限される。

$$\begin{aligned} p_n l_n &< -p_n \log_2 p_n + p_n \\ L = \sum_n p_n l_n &< \sum_n (-p_n \log_2 p_n + p_n) \\ &= H(S) + 1 \end{aligned}$$

So, to summarize these inequalities,　　以上をまとめると以下の不等式を得る。

There exists an instantaneous code which satisfies this inequality regarding the mean code length L:

S に対して平均符号長 L に関する以下の不等式を満たす瞬時符号が存在する (構成できる)。

$$H(S) \leq L < H(S) + 1$$

2.8 Source Coding Theorem; 情報源符号化定理

S^n: n$^{\text{th}}$ extension of information source S　　S^n: n 次の拡大情報源

$$S^n = \begin{pmatrix} a_1 a_1 \ldots a_1 & a_1 a_1 \ldots a_1 a_2 & \ldots & a_M a_M \ldots a_M \\ p_1 p_1 \ldots p_1 & p_1 p_1 \ldots p_1 p_2 & \ldots & p_M p_M \ldots p_M \end{pmatrix}$$

Block coding is a method that concatenates codes together.

ブロック符号化 (block coding): ブロックにまとめて符号化を行なう方法

According to previous analysis, there exist instantaneous codes that satisfy this inequality:

前の考察より S^n に対して平均符号長 L_n が以下の不等式を満たす瞬時符号が存在する。

$$L_n < H(S^n) + 1$$

L: mean code length for one symbol

L: 1 文字当たりの平均符号長

$$L = L_n/n$$

The entropy of an extension equals the order of the extension times the entropy of the first-order code.

ある拡張のエントロピーは拡張の次数かける 1 次符合のエントロピーである。

$$H(S^n) = nH(S) \tag{2.25}$$

Proof: Assuming S is memoryless (its events are independent), the product can be expressed as

証明: S は無記憶（独立）と仮定すると以下のように示せる

$$P(X = a_{i_1}a_{i_2}\ldots a_{i_n}) = p(a_{i_1})p(a_{i_2})\ldots p(a_{i_n}) \tag{2.26}$$

$$H(S^n) = -\sum P(X = a_{i_1}a_{i_2}\cdots a_{i_n}) \log P(X = a_{i_1}a_{i_2}\cdots a_{i_n})$$

so by independence of events (Eqns. 2.26)

$$= -\sum_{i_1=1}^{M}\sum_{i_2=1}^{M}\cdots\sum_{i_n=1}^{M} p(a_{i_1})p(a_{i_2})\cdots p(a_{i_n}) \log\left(p(a_{i_1})p(a_{i_2})\cdots p(a_{i_n})\right)$$

log of products is sum of logs

$$= -\sum_{i_1=1}^{M}\cdots\sum_{i_n=1}^{M} p(a_{i_1})p(a_{i_2})\cdots p(a_{i_n}) \log p(a_{i_1}) - \cdots - \sum_{i_1=1}^{M}\cdots\sum_{i_n=1}^{M} p(a_{i_1})p(a_{i_2})\cdots p(a_{i_n}) \log p(a_{i_n})$$

since respective probabilities sum to unity

$$= -\sum_{i_1=1}^{M} p(a_{i_1}) \log p(a_{i_1}) - \cdots - \sum_{i_n=1}^{M} p(a_{i_n}) \log p(a_{i_n})$$
$$= nH(S)$$

The length per symbol is the length of the extension divided by the order of the extension:

$$\begin{aligned} L &= \frac{L_n}{n} \\ &< \frac{H(S^n)+1}{n} \\ &= H(S) + \frac{1}{n} \end{aligned}$$

Theorem 2.8.1; 定理 2.8.1

There exists a binary information source code C which is uniquely decodable for a memoryless, stationary, discrete information source S. Its mean code length satisfies

無記憶定常離散的情報源 S に対して一意的に復号可能な２元情報源符号 C が存在して、その平均符号長 L が以下を満たすようにできる。

$$H(S) \leq L < H(S) + \epsilon \ \ (\forall \epsilon > 0)$$

Review; 確認

all codes
- non-uniquely decodable
  - singular
  - non-uniquely decodable but non-singular
- uniquely decodable
  - uniquely decodable but non-instantaneous
  - instantaneous (a.k.a. prefix)
    - comma

表 2.14: Containment Hierarchy of Codes

符号
- 一意的に復号可能な符号でない
  - 特異
  - 特異符号でなく一意的に復号可能な符号でない
- 一意的に復号可能な
  - 瞬時符号でなく一意的に復号可能
  - 瞬時 (接頭)
    - コンマ

表 2.15: 符合の階層

2.9 Practice Exercizes; 演習

2.9.1 Questions

1. Show the code trees of $C_1, C_2, C_3, C_4, C_5, C_6, C_7$ and C_8, as presented on [大 93, p. 17] and [大 93, p. 29]. Classify each C_i as one of the types listed below:
   $C_1, C_2, C_3, C_4, C_5, C_6$ ([大 93, p. 17]), C_7, C_8 ([大 93, p. 29]) の符号の木を示せ。さらに C_i を以下のように分類せよ。

   (a) comma code; コンマ符号

   (b) instantaneous (a.k.a. prefix) code; 瞬時符号

   (c) non-instantaneous but uniquely decodable; 瞬時符号でなく一意的に復号可能な符号

   (d) non-singular but non-uniquely decodable; 特異符号でなく一意的に復号可能な符号でない符号

   (e) singular code; 特異符号

2. Show the containment hierarchy of instantaneous, uniquely decodable, singular and comma codes. Show in which part each C_i is contained. 瞬時符号, 一意的に復号可能な符号, 特異符号, コンマ符号の包含関係を示せ。さらに上の C_i はどこに含まれるかを示せ。

2.9.2 Answers; 解答

1. If a symbol is lost, the parser can't recover the code word boundaries.

2. (Illustrated by following code trees)

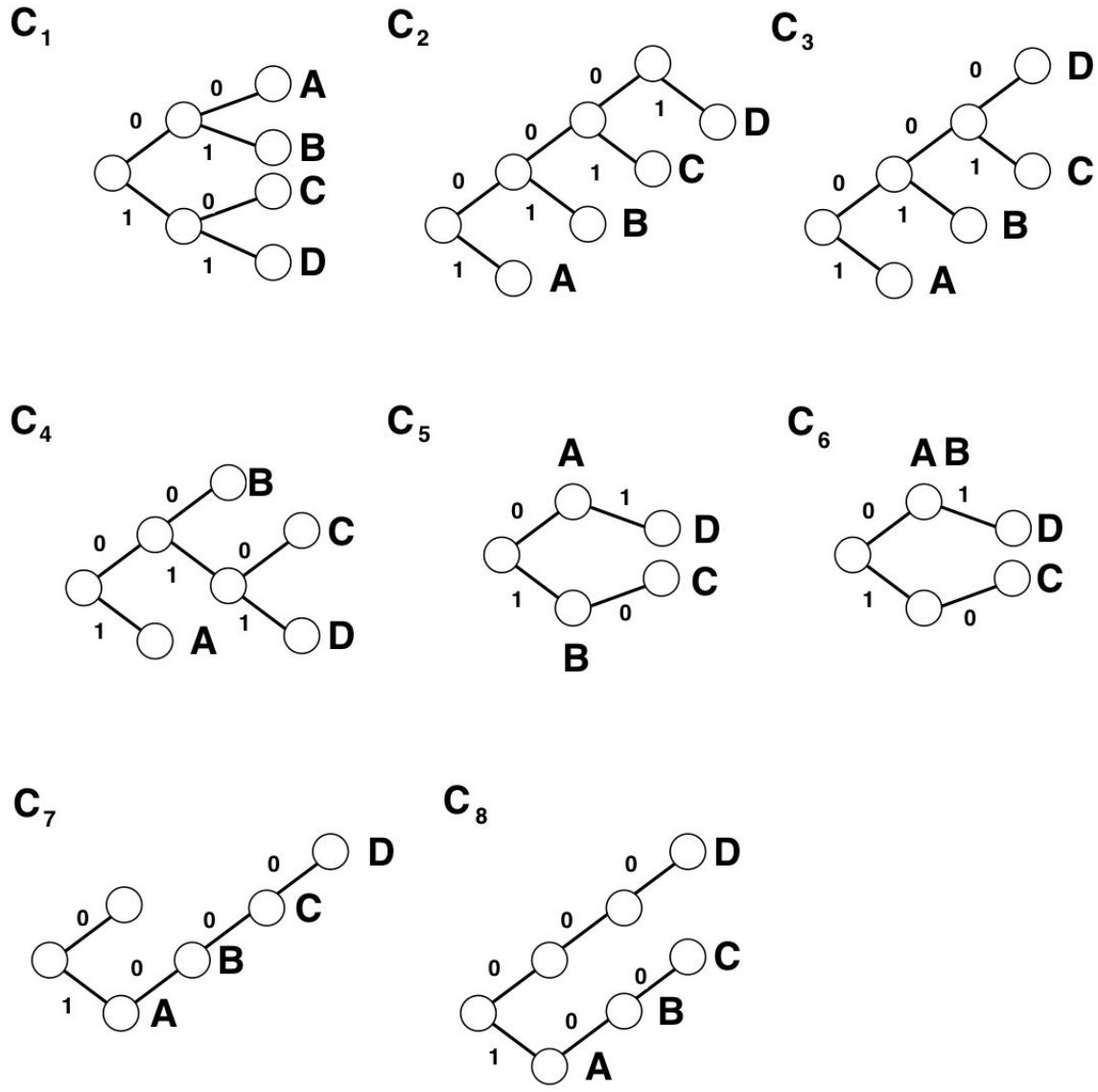


図 2.10: Code Trees; 符合木

3. (Summarized by following Venn diagram)

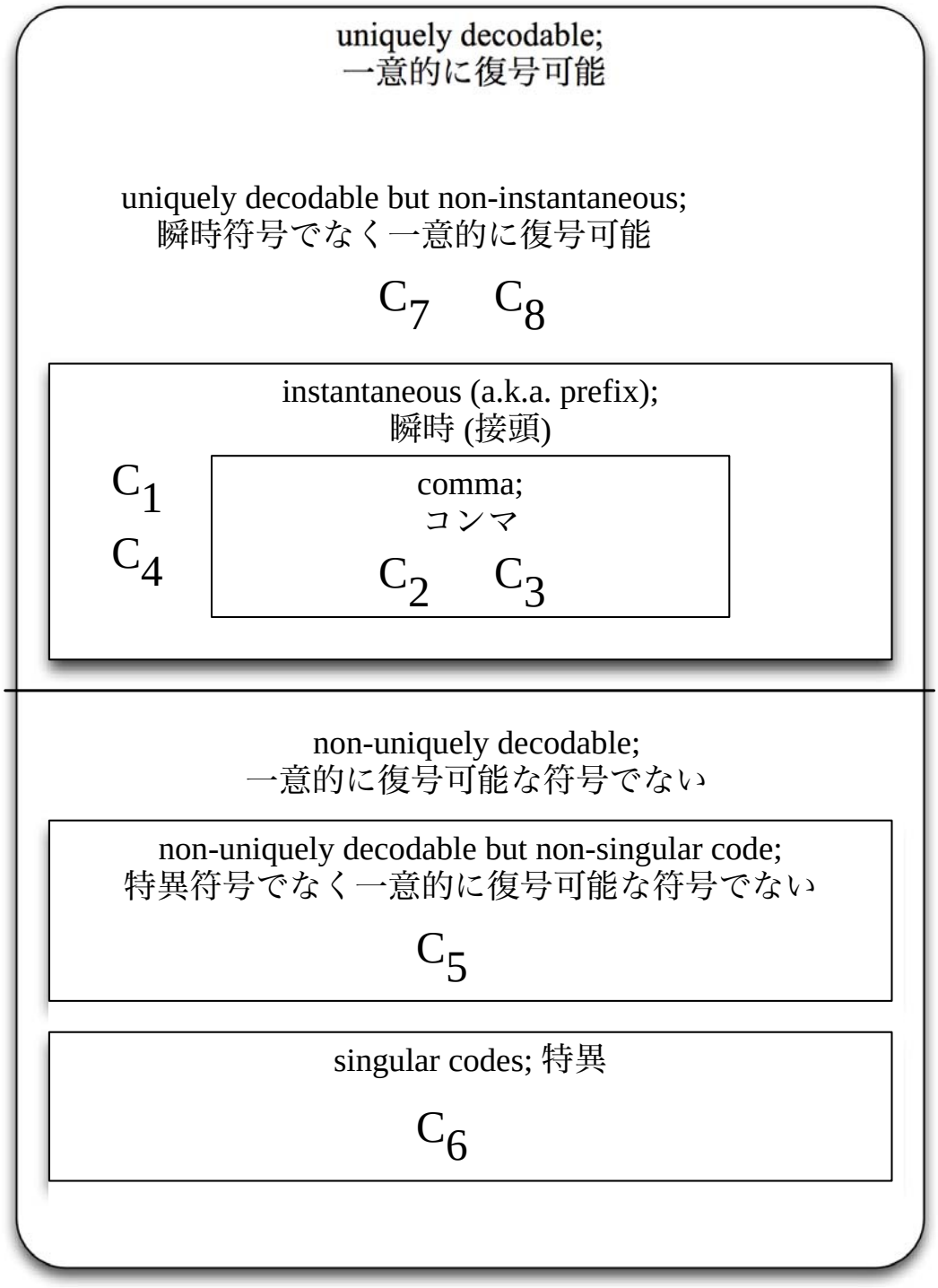


図 2.11: Code Containment (Taxonomy); 符合の包含図 (分類)

4. Check the Kraft-McMillan Inequality for C_7: C_7 の場合

$$K = 2^{-1} + 2^{-2} + 2^{-3} + 2^{-4}$$

Compare to infinite geometric series であり、

$$\begin{aligned} K_\infty &= 2^{-1} + 2^{-2} + \ldots + 2^{-n} + \ldots \\ &= \frac{1/2}{1 - 1/2} \\ &= 1 \end{aligned}$$

であるので、

$$K < K_\infty$$

第3章　3: Various Kinds of Source Codings; いろいろな情報源符号

3.1　Huffman Codes; ハフマン符号, 1962

reduction of information source　　情報源の縮退を次々と行なう

reduced information source; 縮退情報源

Pick the two smallest probability code words, and coalesce them into a single code.　　出現確率の小さい２つの情報源記号を選びだし２つの記号を１つの記号と考える。(出現確率は元の２つの記号の和)

S_r: reduced information source of S (number of codes $M-1$)　　S_r: S の縮退情報源 (記号数 $M-1$)

$$S \longrightarrow S_r$$

$$a_{M-1}, a_M \to a'_{M-1}$$

$$p'_{M-1} = p_{M-1} + p_M$$

$$S_r = \begin{pmatrix} a_1 & a_2 & \dots & a_{M-2} & a'_{M-1} \\ p_1 & p_2 & \dots & p_{M-2} & p'_{M-1} \end{pmatrix}$$

Huffman Coding Algorithm; ハフマン符号化アルゴリズム

Step 1: Each point becomes a node; 各点 $\longrightarrow$ 節点

Step 2: Combine the two smallest probability points to a new node, making a binary subtree; 出現確率 p が最小の２点に対して２分木になる節点を作る (枝)

Step 3: Repeat step 2 to each point of S_r until the number of codes becomes 1; S_r の各点に対して step 2 を繰り返す 記号数が 1 になると終了

Through this process, Huffman code is instantaneous.　　作成手順からハフマン符号は瞬時符号である。

3.2 Compact Code; コンパクト符号

- A compact code is a code which has the shortest mean code length among binary instantaneous codes.
- optimal tree; 最適木
- compact code tree; コンパクト符号の木

コンパクト符号: S に対する 2 元瞬時符号の中で最短の平均符号長を持つ符号

A Huffman code is a compact code.

ハフマン符号はコンパクト符号である

Proof:

証明:

$S_0 = S$ (M 個の記号), S_1 ($M-1$ 個), S_2($M-2$ 個), $\ldots, S_{M-1}$ (1 個)

$$S_0 = S \supset S_1 \supset S_2 \supset \ldots \supset S_{M-2} \supset S_{M-1}$$

T_i: code tree of S_i

T_i: S_i に対する符号の木

3.3 Markov Information Source (Process); マルコフ情報源

An m[th]-order Markov model "remembers" m previous states. Modeling a coding process, the probability of a symbol occurring in a message depends on finite number of proceeding symbols.

m 重マルコフモデルある記号が一つのメッセージに出現する確率は、有限の以前に出現した記号の数に依存する。

A first-order, or single-memory, Markov model models its transitions as $p(a_j|a_i)$, the conditional probability of a_j with previous state a_i. For simple example: "Tomorrow's weather will probably be like today's." Such a system S comprises states A and transitions T.

(１重) マルコフ モデル:
$p(a_j|a_i)$: 一つ前の出力が a_i の時の a_j の条件付き確率簡単な例:「明日の天気は、たぶん今日と似ている」このようなシステム S は状態 A と遷移 T からなる。

By using the structure of the informatoin source, we can reduce the size of its encoding. For instance, in English text, knowing that the previous letter is 'q' strongly suggests that the next letter will be 'u' (as in "queen," "quadrant," "question," "quick," "quiet," etc.).

$$S = (A, T) \tag{3.1a}$$

$$A = \{a_1, a_2, \ldots, a_m\} \tag{3.1b}$$

A transition matrix is square and has inputs coming in along the rows, and outputs totalled up for each column. This sum-of-products form appears frequently in math and engineering. The first subscript of each transition matrix element indexes the input, and the second indexes the output. The total for each row is unity.

遷移行列は、水平方向に入力、垂直方向に出力を持っている。この積和の形式は数学や工学において頻繁に現れる。それぞれの要素の一つ目の添え字は入力を、二つ目は出力をそれぞれ表す。各列は１となる。

Think of a matrix as a cross-bar that distributes flow.

$$T = \begin{pmatrix} t_{11} & t_{12} & \dots & t_{1m} \\ t_{21} & t_{22} & \dots & t_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ t_{m1} & t_{m2} & \dots & t_{mm} \end{pmatrix} \tag{3.2a}$$

$$t_{ij} = p(a_j|a_i) \tag{3.2b}$$

$$T = \begin{pmatrix} p(a_1|a_1) & p(a_2|a_1) & \dots & p(a_m|a_1) \\ p(a_1|a_2) & p(a_2|a_2) & \dots & p(a_m|a_2) \\ \vdots & \vdots & \ddots & \vdots \\ p(a_1|a_m) & p(a_2|a_m) & \dots & p(a_m|a_m) \end{pmatrix} \tag{3.2c}$$

3.4 Transition Diagram; 遷移図

Shannon state diagram: Each node represents the current information source output.

シャノン線図：節点(せってん)は状態（現在の情報源の出力）

Example: transition matrix corresponding to Figure 3.1.

例：図 3.1 に対応する遷移行列

$$T = \begin{pmatrix} \frac{1}{5} & \frac{1}{5} & \frac{3}{5} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{pmatrix}$$

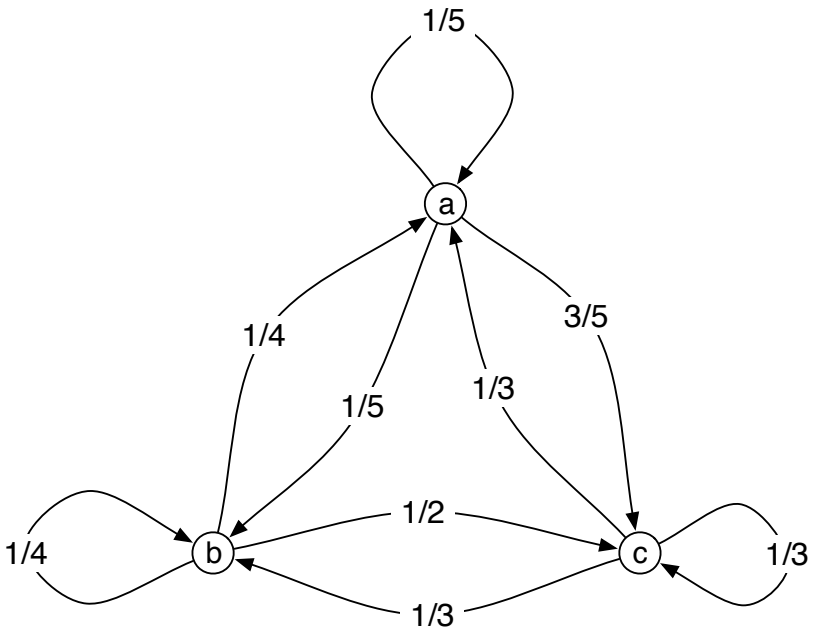


図 3.1: State Diagram: Ergodic (Hidden) Markov Chain/Model; マルコフ情報源 (repeated from §1.9)

3.4.1 Trellis Diagram: Markov Information Source (Process); マルコフ情報源 (2)

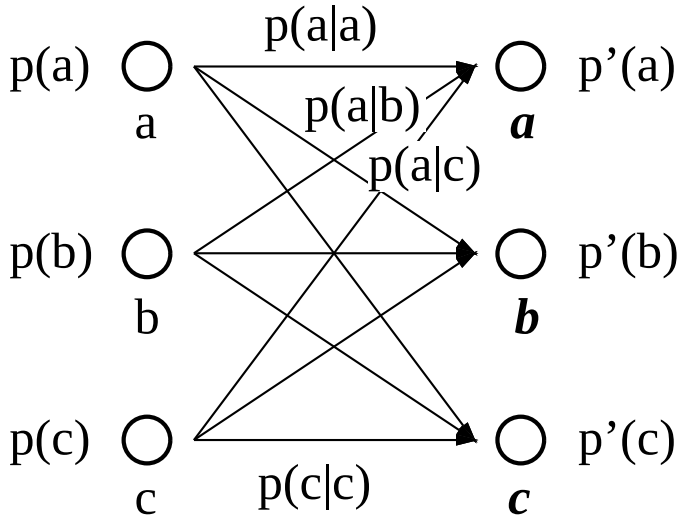


図 3.2: Trellis Diagram of 3-State Markov Process: $p'(a) = p(a)p(a|a) + p(b)p(a|b) + p(c)p(a|c); p'(b) = p(a)p(b|a) + p(b)p(b|b) + p(c)p(b|c); p'(c) = p(a)p(c|a) + p(b)p(c|b) + p(c)p(c|c)$

We can obtain the following equation, as illustrated by Figure 3.2.

図 3.2 から以下の関係式を得る。

$$\vec{p'} = \vec{p}\,T \tag{3.3a}$$

$$(p'_1, p'_2, \ldots, p'_m) = (p_1, p_2, \ldots, p_m)T \tag{3.3b}$$

Equations 3.3 are shorthand (abbreviations) for applications of the Markov chain rule, using square matrix T to represent all the transitions.

全ての遷移を示すために、正方行列 T を用いて eqn. (3.3) のように表す事ができる。

$$\begin{aligned}
p'_1 = &\quad p_1t_{11} + p_2t_{21} + \ldots + p_mt_{m1} = p_1p(a_1|a_1) + p_2p(a_1|a_2) + \ldots + p_mp(a_1|a_m) \\
p'_2 = &\quad p_1t_{12} + p_2t_{22} + \ldots + p_mt_{m2} = p_1p(a_2|a_1) + p_2p(a_2|a_2) + \ldots + p_mp(a_2|a_m) \\
&\qquad\qquad \vdots \\
p'_m = &\quad p_1t_{1m} + p_2t_{2m} + \ldots + p_mt_{mm} = p_1p(a_m|a_1) + p_2p(a_m|a_2) + \ldots + p_mp(a_m|a_m)
\end{aligned}$$

In a 3-dimensional case, point p' is mapped (calculated) from point p in a triangle embedded in 3-space, as illustrated by Figure 3.3.

3 次元の場合は以下の図 3.3 のような 3 次元空間の三角形の領域内の点 p から p' が次々と計算されることになる。

For repeated applications of the transition,

と定義すると、以下のように書ける。

$$\vec{p}_n = \vec{p}_0 T^n \tag{3.4}$$

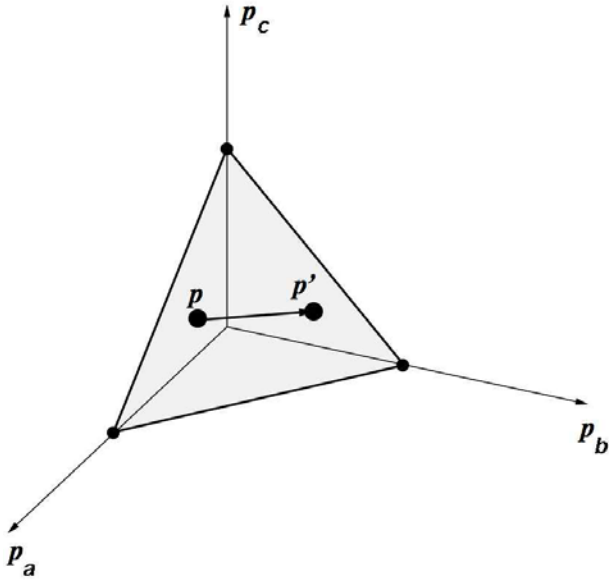


図 3.3: 3-Dimensional Probability Transition; 状態遷移の 3 次元表現 $(p_a + p_b + p_c = 1)$—$(p'_a, p'_b, p'_c) = (p_a, p_b, p_c)T$

That is, the n^{th} transition can be represented as T^n, the transition matrix T raised to power n (equivalent to successive matrix dot products).

従って、n 次の遷移は T^n で表される。

The outer product of two matrices yields another matrix (of necesarily the same size):

2 つの行列の外積はもうひとつの行列を生じさせる（いつも同じ大きさ）:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} e & f \\ g & h \end{pmatrix} = \begin{pmatrix} ae & bf \\ cg & dh \end{pmatrix} \tag{3.5}$$

but the inner or "dot" product (scalar product for vectors) yields a different matrix:

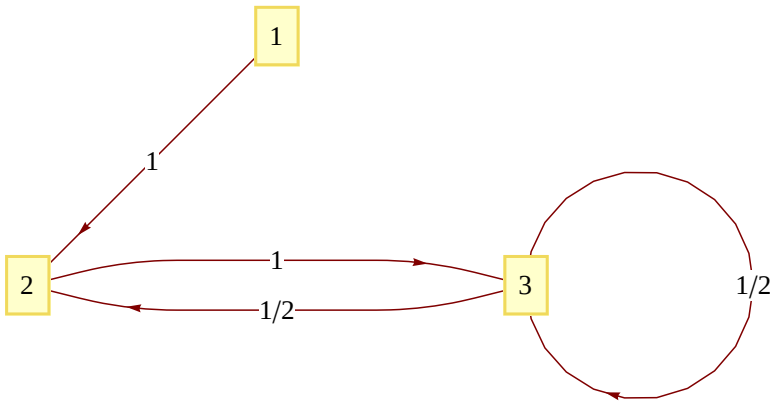
しかし、内積 (スカラー積) は異なる行列を生じさせる:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} e & f \\ g & h \end{pmatrix} = \begin{pmatrix} ae+bg & af+bh \\ ce+dg & cf+dh \end{pmatrix} \tag{3.6}$$

3.5 Ergodic Markov Process; エルゴード的 マルコフ情報源

It grows impossible to predict the future (after a few generations), except for general steady-state probability distribution. In an ergodic Markov process, any state can be reached from any other state, and in time the system settles down to a limited description, independent of the initial state.

エルゴート的マルコフでは、どの状態もほかの状態から辿り着くことができる。そして、システムはそのうち初期状態から独立している限られた状態へと落ち着く。



(a) Transition Matrix

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

(b) Transition Graph

図 3.4: Reducible Graph; “Leaky Flip-FLop”

irreducible Excepting itself, there is no closed member (“transitive closure” of the set is the same) of power set. Every point can eventually (in finite time) be reached from any other point.

recurrence time the minimum time to return to an original state. For example, if auto-transition is possible, $\tau = 1$. For two-state system (like even/odd) without such reflection, $\tau = 2$.

periodicity When the recurrence time is divisible by integer k and k is the largest divisor, it is periodic with period k.

non-periodic when $k = 1$;

Ergodic process irreducible and non-periodic

normal possibility of reaching all nodes: there is some n such that $\vec{p}T^n > \vec{0}$.

既約 自分自身以外に閉じた部分集合を持たない ([大 93, 図 3.3 (a),(b)]); $\Longrightarrow$ どの節点からの他の任意の節点に行ける [大 93, 図 3.3 (c)]$\{1, 2, 3\}$ に収束 ($\{1, 2, 3\}$ は既約)

再帰時間 状態から出発して始めてその状態に帰ることのできる最小の推移時間

周期的 任意の状態の再帰時間がある整数 k で割り切れそれより大きな正整数で割り切れない時、周期 k で周期的であるという

非周期的 $k = 1$ の時

エルゴード性 既約かつ非周期的

正規 : $\exists n$; T^n の全ての要素 $> \vec{0}$ (到達可能性)

Theorem; 定理

A sufficient condition for an Markov process being ergodic is that there exists an integer r satisfying the condition below, where $t_{ij}^{(r)}$ are the (i, j) elements of T^r

有限状態マルコフ情報源がエルゴード的であるための必要十分条件は、ある正整数 r が存在して以下を満たす。ここで $t_{ij}^{(r)}$ は T^r の i, j 要素である。

$$t_{ij}^{(r)} > 0 \quad (\forall i, j = 1, \ldots, m)$$

Each state is generated by its immediate ancestor (predecessor).

それぞれの状態はその直近の先祖(前のもの)により生成される。

$$\vec{p_n} = \vec{p}_{n-1} T \tag{3.7}$$

An ergodic Markov process will eventually settle into a steady state, independent of initial conditions.

であるので、両辺を極限をとって以下の式を得る。

$$\vec{p_k} = \vec{p}_{k-1}, \quad k \geq N \tag{3.8a}$$
$$\vec{p} = \vec{p}\, T \tag{3.8b}$$

For $\vec{p_0} = (1,0)$ and transition matrix $M = \begin{pmatrix} 0 & 1 \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix}$,

$$\vec{p_1} = (1,0)\begin{pmatrix} 0 & 1 \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix} = (0,1) = \vec{p_0}\ M \tag{3.9a}$$
$$\vec{p_2} = (0,1)\begin{pmatrix} 0 & 1 \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix} = (\frac{1}{2},\frac{1}{2}) = \vec{p_0}\ M^2 = (1,0)\begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{4} & \frac{3}{4} \end{pmatrix} \tag{3.9b}$$
$$\vec{p_3} = (\frac{1}{2},\frac{1}{2})\begin{pmatrix} 0 & 1 \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix} = (\frac{1}{4},\frac{3}{4}) = \vec{p_0}\ M^3 = (1,0)\begin{pmatrix} \frac{1}{4} & \frac{3}{4} \\ \frac{3}{8} & \frac{5}{8} \end{pmatrix} \tag{3.9c}$$
$$\vec{p_4} = (\frac{1}{4},\frac{3}{4})\begin{pmatrix} 0 & 1 \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix} = (\frac{3}{8},\frac{1}{4}+\frac{3}{8}) = (\frac{3}{8},\frac{5}{8}) = \vec{p_0}\ M^4 = (1,0)\begin{pmatrix} \frac{3}{8} & \frac{5}{8} \\ \frac{5}{16} & \frac{11}{16} \end{pmatrix} \tag{3.9d}$$
$$\vec{p_5} = (\frac{5}{16},\frac{6}{16}+\frac{5}{16}) = (\frac{5}{16},\frac{11}{16}) = \vec{p_0}\ M^5 = (1,0)\begin{pmatrix} \frac{5}{16} & \frac{11}{16} \\ \frac{11}{32} & \frac{21}{32} \end{pmatrix} \tag{3.9e}$$
$$\vec{p_6} = (\frac{11}{32},\frac{5}{16}+\frac{11}{32}) = (\frac{11}{32},\frac{21}{32}) = \vec{p_0}\ M^6 = (1,0)\begin{pmatrix} \frac{11}{32} & \frac{21}{32} \\ \frac{21}{64} & \frac{43}{64} \end{pmatrix} \tag{3.9f}$$
$$\vdots$$
$$\vec{p_\infty} = \vec{p_0}\ M^\infty = (1,0)\begin{pmatrix} \frac{1}{3} & \frac{2}{3} \\ \frac{1}{3} & \frac{2}{3} \end{pmatrix} \tag{3.9g}$$

This sequence converges, to limits which can be determined analytically as well as numerically.

Recalling that the identity matrix is (sparsely) populated by unity along its main diagonal,

斜め方向に 1 が並ぶ行列、即ち恒等行列を用いて、

$$I = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix}, \tag{3.10}$$

and solving the eigenequation,

以下の様に固有方程式を解くことができる。

$$\vec{p}\,T = \lambda\vec{p} \tag{3.11a}$$
$$\vec{p}\,T - \lambda\vec{p} = \vec{0} \tag{3.11b}$$
$$\vec{p}(T - \lambda I) = \vec{0} \tag{3.11c}$$

where λ is the eigenvalue (which can be unity), I is the identity matrix, and $\vec{p}$ is the eigenvector.

λ は固有値、I は単位行列、$\vec{p}$ は固有ベクトル

This homogeneous equation (right-hand side is zero vector) has non-trivial solution only if the matrix is singular. A singular matrix is one whose determinant is zero, which means that there is a linear dependency of the rows and columns.

この同次方程式は行列が特異行列の場合のみ解を持つ。特異行列は行列式が 0 である行列。つまり行と列に線形従属がある。

For example, a non-singular matrix yields only a degnerate solution to the homogeneous equations:

例えば、非特異行列により同次方程式の縮退解を得ることができる。

$$\begin{aligned}
(p_1, p_2)\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} &= \vec{0} \\
p_1 + 3p_2 &= 0 \\
p_1 &= -3p_2 \\
2p_1 + 4p_2 &= 0 \\
p_1 &= -2p_2
\end{aligned}$$

Equations	linearly dependent	independent
Singularity	singular	non-singular
Determinant; $\det M \equiv \|M\|$	$= 0$	$\neq 0$
Solution	non-unique	unique

表 3.1: Algebraic Equation System Comparison

The degenerate $\vec{p} = \vec{0}$ (corresponding to a vector that is identically zero) is a solution, but an eigenvector must be non-zero. Such a system has a unique solution only if its determinant $\det(T - \lambda I)$ is non-zero, so the existence of both a zero solution and an eigenpair (eigenvector and eigenvalue) means that the solution is non-unique, the system is singular, and the determinant vanishes (is zero).

縮退 $\vec{p} = \vec{0}$ が解であるが、固有ベクトルは 0 であってはならない。系は、$\det(T - \lambda I)$ が 0 でない時にのみ解が定まるので、解が存在せず固有対 (固有ベクトル及び固有値) が 0 である場合には、解は一つに定まらず、系は単数であり、そしてデターミナントは 0 になる。

For the original 2D example above,

$$\vec{p}T = \lambda\vec{p} \tag{3.12a}$$
$$\vec{p}(T - I\lambda) = \vec{0} \tag{3.12b}$$
$$T - I = \begin{pmatrix} 0-1 & 1 \\ \frac{1}{2} & \frac{1}{2}-1 \end{pmatrix} \tag{3.12c}$$

(Of course, this matrix no longer has the property that the rows total unity.) Note that $T - I$ has linear dependency of rows and columns.

$$\vec{p}\begin{pmatrix} -1 & 1 \\ \frac{1}{2} & -\frac{1}{2} \end{pmatrix} = \vec{0} \tag{3.12d}$$
$$-p_1 + \frac{1}{2}p_2 = 0 \text{ (and } p_1 - \frac{1}{2}p_2 = 0) \Rightarrow p_1 = \frac{p_2}{2} \tag{3.12e}$$
$$p_1 + p_2 = 1 \tag{3.12f}$$
$$\frac{3p_2}{2} = 1 \tag{3.12g}$$
$$p_2 = \frac{2}{3} \tag{3.12h}$$
$$p_1 = \frac{1}{3} \tag{3.12i}$$
$$\vec{p} = (\frac{1}{3}, \frac{2}{3}) \tag{3.12j}$$

Confirming...

$$\vec{p}T = (\frac{1}{3}, \frac{2}{3}) \ \surd \tag{3.12k}$$

For the more complicated 3D example further above, eqn. (3.11a) is a compact way of expressing these equations:

上記の例として、eqn. (3.11a) はこれらの方程式を表す簡潔な方法である:

$$(p_a, p_b, p_c)T = \lambda(p_a, p_b, p_c) \tag{3.13a}$$
$$(p_a, p_b, p_c)\begin{pmatrix} p(a|a) & p(b|a) & p(c|a) \\ p(a|b) & p(b|b) & p(c|b) \\ p(a|c) & p(b|c) & p(c|c) \end{pmatrix} = \lambda(p_a, p_b, p_c) \tag{3.13b}$$
$$(p_a, p_b, p_c)\begin{pmatrix} \frac{1}{5} & \frac{1}{5} & \frac{3}{5} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{pmatrix} = \lambda(p_a, p_b, p_c) \tag{3.13c}$$

So, for the example,

$$\begin{cases} \frac{p_a}{5} + \frac{p_b}{4} + \frac{p_c}{3} & = \lambda p_a \\ \frac{p_a}{5} + \frac{p_b}{4} + \frac{p_c}{3} & = \lambda p_b \\ \frac{3p_a}{5} + \frac{p_b}{2} + \frac{p_c}{3} & = \lambda p_c \end{cases}$$

$$\vec{p}(T-\lambda I)=\vec{0}$$

$$\begin{aligned} p_a(\frac{1}{5}-\lambda)+\frac{p_b}{4}+\frac{p_c}{3} &= 0 \\ \frac{p_a}{5}+p_b(\frac{1}{4}-\lambda)+\frac{p_c}{3} &= 0 \\ \frac{3p_a}{5}+\frac{p_b}{2}+p_c(\frac{1}{3}-\lambda) &= 0 \end{aligned}$$

Let $\lambda = 1$:

$$-\frac{4}{5}p_a+\frac{1}{4}p_b+\frac{1}{3}p_c=0 \tag{3.15a}$$
$$\frac{1}{5}p_a-\frac{3}{4}p_b+\frac{1}{3}p_c=0 \tag{3.15b}$$
$$\frac{3}{5}p_a+\frac{1}{2}p_b-\frac{2}{3}p_c=0 \tag{3.15c}$$

As foretold above, for this sytem, the determinant of the matrix does indeed vanish:

上記の説明の通り、この系における行列の行列式は 0 になる。

$$\det\begin{pmatrix} -\frac{4}{5} & \frac{1}{4} & \frac{1}{3} \\ \frac{1}{5} & -\frac{3}{4} & \frac{1}{3} \\ \frac{3}{5} & \frac{1}{2} & -\frac{2}{3} \end{pmatrix} = \begin{Vmatrix} -\frac{4}{5} & \frac{1}{4} & \frac{1}{3} \\ \frac{1}{5} & -\frac{3}{4} & \frac{1}{3} \\ \frac{3}{5} & \frac{1}{2} & -\frac{2}{3} \end{Vmatrix} = 0$$

Note that this $T-I$ matrix also has a linear dependence of its rows and columns. (The top row is the negative of the sum [additive inverse] of the second and third rows, etc.)

Since these equations model a Markov process, they are not independent. The $T-I$ matrix has a linear dependency of the rows and columns. (The top row is negative of the sum of the second and third rows.) Combining (subtracting) the first two of these yields

これらの方程式がマルコフ過程を表す事から、これらは独立では ない。これらのうち始めの二つを結合 (減算) する

$$-p_a+p_b=0 \tag{3.16}$$

but combining (twice) the second and third yields such a redundant equation:

しかし、2 つ目と 3 つ目を結合すると、不要な方程式が生じてしまう。

$$p_a-p_b=0$$

Likewise, twice the first added to the third also yields the redundant equation.

つ目を 2 倍したものに 3 つ目を加えることでまた、不要な方程式が生じる。

However, solving any two of them, combined with side-condition $\sum_i p_i = 1$, yields the unique steady-state solution.

うち 2 つを解き、$\sum_i p_i = 1$ と組み合わせることで固有の解を得ることができる。

$$p_a+p_b+p_c=1 \tag{3.17}$$

Continuing to solve the system (thrice the first equation minus the side-condition),

それから、解答を続ける (一つ目の方程式を 3 倍して引く)

$$(-\frac{12}{5}-\frac{5}{5})p_a+(\frac{3}{4}-1)p_b=-1.$$

Also, since, as in eqn. (3.16), $p_a=p_b$,

$$\begin{aligned}-\frac{17}{5}p_a-\frac{1}{4}p_b&=-1\\ \frac{-68-5}{20}p_a&=-1\\ p_a=p_b=\frac{20}{73};\ p_c&=\frac{33}{73}\end{aligned}$$

$$\begin{aligned}\vec{p}&=(p_a,p_b,p_c)\\ &=(\frac{20}{73},\frac{20}{73},\frac{33}{73})\approx(0.27,0.27,0.45)\end{aligned}$$

Confirming...

$$\vec{p}T\stackrel{?}{=}\vec{p} \tag{3.18}$$

$$(\frac{20}{73},\frac{20}{73},\frac{33}{73})\begin{pmatrix}\frac{1}{5}&\frac{1}{5}&\frac{3}{5}\\ \frac{1}{4}&\frac{1}{4}&\frac{1}{2}\\ \frac{1}{3}&\frac{1}{3}&\frac{1}{3}\end{pmatrix}=(\frac{20}{73},\frac{20}{73},\frac{33}{73})\ \surd \tag{3.19}$$

3.5.1 Formal Consideration

Theorem; 定理

If a finite state Markov process is ergodic, the steady-state probability distribution $\vec{p}=(p_i)$ that satisfies the characteristic equation is unique and

有限状態マルコフ情報源がエルゴード的であれば、固有方程式を満足する定常確率分布 $\vec{p}=(p_i)$ は一意的であり、しかも

$$p_i>0\quad(i=1,\ldots,m)$$

- $\forall$ initial states, there is a constant probability distribution limit. The initial state doesn't matter (since there is no recurrence period).

$\forall$ 初期状態 $\Longrightarrow$ 一定の確率分布関数。初期常態は関係ない。(周期が無いから)。

Theorem; 定理

If a finite state Markov process is ergodic, an arbitrary initial probability distribution $\vec{q}=(q_i)$ converges on a final (stable stationary steady-state) probability distribution after the transient behavior dies away.

有限状態マルコフ情報源がエルゴード的であれば、任意の初期確率分布 $\vec{q}=(q_i)$ に対して

$$\lim_{n\to\infty}q_i^{(n)}=p_i\quad(\forall i=1,\ldots,m)$$

Markov Rule; マルコフの古典極限定理

A transition matrix T of a Markov process has unity as eigenvalue, and

正規なマルコフ情報源において遷移行列 T は 1 を固有値としてもつ。さらに、

$$p_n = \frac{|q_n|}{\sum |q_k|}$$

$\vec{p} = (p_1, p_2, \ldots, p_m)$ is an eigenvector of T with eigenvalue unity, and there is only one that satisfies $\sum p_n = 1, p_n \geq 0$.

となる $\vec{p} = (p_1, p_2, \ldots, p_m)$ が T の固有値 1 の固有ベクトルであり、$\sum p_n = 1, p_n \geq 0$ を満たすものはただ一つである。

Proof: The sum of each of the rows of transition matrix T equals unity. Therefore, $\det(T - I) = 0$, and $T - I$ is singular. This implies that unity is an eigenvalue of T, and a vector $\vec{q} \neq \vec{0}$ satisfies the following equation:

証明: 遷移行列 T の各行を加えると 1 になる。従って、$\det(T - I) = 0$ であるので $T - I$ は正則ではない。これは 1 が T の固有値であることを示している。これより、以下を満たすベクトル $\vec{q} \neq \vec{0}$ が存在する。

$$\vec{q}(T - I) = \vec{0} \tag{3.20}$$

To write each element of the matrix, note that for $t_{nk} = p(a_k|a_n)$, and each output state k has probability q_k.

成分毎に書くと以下の式を得る。ただし、$t_{nk} = p(a_k|a_n)$ である。

$$\sum_{n=1}^{m} q_n t_{nk} = q_k$$

The sum of absolute products is no less than the absolute sum of products. (The absolute SoP is bounded by the absolute series.)

積の絶対値の和は積の和の絶対値に等しい。(積の和の絶対値は絶対値の級数に隣接する。)

$$\sum_{n=1}^{m} |q_n t_{nk}| \geq |\sum_{n=1}^{m} q_n t_{nk}| = |q_k|$$

For instance, $q_n = (1, 1), T = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} \end{pmatrix}$ yields $|1| + |-1| \geq |1 - 1|$.

例えば、

Sum up, with k as an index, m outputs (columns):

k に関して和を取ると、

$$\sum_{k=1}^{m} \sum_{n=1}^{m} |q_n||t_{nk}| \geq \sum_{k=1}^{m} |q_k|$$

Since T is a transition matrix, the sum of every row is unity. Since for any input (row) $\sum_{k=1}^{m} |t_{nk}| = 1$, equality is achieved, and

$\sum_{k=1}^{m} |t_{nk}| = 1$ であるので、左辺と右辺は一致するので、以下を得る。

$$\sum_{n=1}^{m} |q_n||t_{nk}| = |q_k|$$

Therefore, $\vec{q} = (|q_n|)$ is a eigenvector of T. It is still a eigenvector if uniformly scaled (multiplied) to normalize,

従って、$\vec{q} = (|q_n|)$ が T の固有ベクトルとなる。これを定数倍しても固有ベクトルであるので（正規化する）、

$$p_n = \frac{|q_n|}{\sum_{k=1}^{m} |q_k|},$$

and $\vec{p} = (p_n)$ is a unique eigenvector corresponding to eigenvalue unity of transition matrix T.

とすると、$\vec{p} = (p_n)$ は遷移行列 T の固有値 1 に対応する固有ベクトルとなる。一意性については省略。

3.6 Source Coding of (First-Order) Markov Process (Information Source); マルコフ情報源に対する情報源符号化

Let a_i be the current output. Make Huffman coding C_i while taking the probability of next symbol $(p(a_1|a_i), p(a_2|a_i), \ldots, p(a_m|a_i))$ into account.

現在、記号 a_i を出力している。このとき、次の記号の出現確率 $(p(a_1|a_i), p(a_2|a_i), \ldots, p(a_m|a_i))$ を考慮して、ハフマン符号化 C_i を行なう。

L_i: mean code length of Huffman coding C_i:

L_i: ハフマン符号 C_i の平均符号長

Mean code length of Markov process is:

マルコフ情報源に対する平均符号長は以下で与えられる。

$$\tilde{L} = p(a_1)L_1 + p(a_2)L_2 + \ldots + p(a_m)L_m$$

L: mean code length when Huffman coding with $(p(a_1), p(a_2), \ldots, p(a_m))$

L: $(p(a_1), p(a_2), \ldots, p(a_m))$ でハフマン符号化する時の平均符号長

$$\tilde{L} \leq L$$

3.7 Run-length encoding (RLE); ランレングス符号

RLE is especially useful if the data is naturally sparse (mostly zeros) or has been filtered by a predictive filter.

あるデータが元々わずかである場合やフィルタリングされている場合に RLE は特に有用である。

$$p + q = 1 \Leftrightarrow q = 1 - p$$

$$S = \begin{pmatrix} 0 & 1 \\ p & q \end{pmatrix}$$

memoryless information source

無記憶定常情報源

If $p \approx 1$, we can expect long runs of zeros, $01\overbrace{00\ldots0}^{n}1\ldots$, using n to count the number of zeros in each run.

$$p(n) = p^n q, \quad \sum_{n=0}^{\infty} p(n) = 1$$

Confirm that the sum of the probabilities is 100%:

確率の和が 100

$$\begin{aligned} S &= \sum_{n=0}^{\infty} p^n q \\ &= q \sum p^n \\ &= q(p^0 + p^1 + p^2 + \ldots) \\ pS &= q(\quad p^1 + p^2 + \ldots) \\ S(1-p) &= qp^0 \\ S &= \frac{q}{1-p} \\ &= 1 \ \surd \end{aligned}$$

Arbitrarily long run length is impossible to handle practically, so block it into finite chunks no longer than length M.

以前の議論は任意の長さの記号には適用できない ⇒ 有限個の run length M を考える (M 以上は不明)

$$p(n) = \begin{cases} p^n q & \text{if } 0 \le n \le M-1, \\ p^M & \text{if } n = M. \end{cases} \tag{3.21}$$

(This is similar to the optimization of comma codes such that a codeword is terminated by a designated symbol or upon reaching a maximum length.)

$$\begin{aligned} S &= \sum_{n=0}^{M-1} p^n q \\ &= q \sum_{n=0}^{M-1} p^n \\ &= q(p^0 + p^1 + p^2 + \ldots + p^{M-1}) \\ pS &= q(\quad p^1 + p^2 + \ldots + p^{M-1} + p^M) \\ S(1-p) &= q(p^0 - p^M) \\ S &= q\frac{1-p^M}{1-p} \\ &= 1 - p^M \ \surd \end{aligned}$$

Example: '0010100010000000100000011' would be parsed (counting zeros) as 2,1,3,4,2,4,2,0, which is distributed as p(0)=$\frac{1}{8}$, p(1)=$\frac{1}{8}$, p(2)=$\frac{3}{8}$, p(3)=$\frac{1}{8}$, p(4)=$\frac{2}{8}$. So the information would be sent as 00—010—011—10—00—10—00—11 (separators shown just for clarity; they're not transmitted). In this simple example, the coding factor is only 24/18, but a larger example would show a higher compression ratio.

Then Huffman code the M-piece blocks.

M 個の記号に関してハフマン符号化: ランレングスハフマン符号化

3.8 Source Coding and Algorithm; 情報源符号化とアルゴリズム

2-pass Huffman coding　２ーパスハフマン符号化

unknown occurrence probability　出現確率は未知

need to find occurrence probability　出現確率を得る必要がある

$$S = \begin{pmatrix} a_1 & a_2 & \dots & a_M \\ & & ? & \end{pmatrix}$$

First pass measure the occurrence possibility　**パス１** 出現確率を調べる

Second pass make a Huffman code　**パス２** ハフマン符号化を行なう

[大 93, p. 53] を参照。

symbol	fraction	prob.	comma codes	
記号	出現回数	確率	コンマ符号	
A	$\frac{16}{32}$	0.5	0	1
B	$\frac{8}{32}$	0.25	10	01
C	$\frac{4}{32}$	0.125	110	000
D	$\frac{4}{32}$	0.125	111	001

表 3.2: Sample 2-Pass Huffman Coding [大 93, p. 53]

Exercise: Make a Huffman code based on the observed probabilities.　演習: 上の確率に基づきハフマン符号符号化を行なう。

3.9 Ziv-Lempel Coding; ZL 符号

Basic strategy: create and reference dynamic dictionary by incrementally scanning the input source.　基本方針：入力をインクリメンタルスキャンにより動的辞書を作り参照する。

Parse source sequence into strings that have not yet appeared: After every break, look along the input sequence for the shortest phrase not previously observed. As it's the shortest such string, all of its prefixes must have occurred earlier. Encode as the location of its longest prefix and its last bit.　文字列の中でまだ出現していない情報の順序を分析する。その後、入力の順番に前に出てきてないような一番短い文字列を見つける。そのような文字列のプリフィックスはそれより前に出現していなければならない。プリフィックスの位置と最後のビットの値によってエンコードする。

Example:

0100010010111101000101000101101000

0|1|00|01|001|011|11|010|0010|10|00101|101|000

There being 13 phrases, $\lceil \log_2 13 \rceil = 4$ bits are needed to describe a dictionary index (location) without optimization.

13個のフレーズがあるので最適化しなければ辞書のインデックスをあらわすために4ビット必要である。

	0	**1**	**00**	**01**	**001**	**011**	**11**	**010**	**0010**	**10**	**00101**	**101**	**000**
w_0	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}	w_{11}	w_{12}	w_{13}
	(w_0;)0	**w_0;1**	**w_1;0**	**w_1;1**	**w_3;1**	**w_4;1**	**w_2;1**	**w_4;0**	**w_5;0**	**w_2;0**	**w_9;1**	**w_{10};1**	**w_3;0**
	0	**0;1**	**01;0**	**01;1**	**011;1**	**100;1**	**010;1**	**100;0**	**0101;0**	**0010;0**	**1001;1**	**1010;1**	**0011;0**

0|01|010|011|0111|1001|0101|1000|01010|00100|10011|10101|00110

0010100110111100101011000010100010010011101010 0110

In this short example, the output stream is longer than the input stream (negative compression!), but as the source stream lengthens, the mean code word length approaches the entropy.

十分長い系列を符号化する時、1記号あたりの平均符号長が情報源のエントロピーに収束する

3.10 Arithmetic Coding

(Paraphrased from http://www.faqs.org/faqs/compression-faq/part2.)

Arithmetic coding works by representing a message as an interval of real numbers between 0 and 1. As the message becomes longer, the interval needed to represent it becomes smaller and smaller, and the number of bits needed to specify that interval increases. Successive symbols in the message shrink this interval in accordance with the probability of that symbol. The more likely symbols reduce the range by less, and thus add fewer bits to the message.

符号は0から1の実数の間隔によって数字を表現することである。メッセージが長くなればなるほど、それを表現するのに必要な間隔は短くなり、その間隔を表すビット数は増える。メッセージの中の連続した記号はその記号の出現確率に一致してこの間隔を減らす。より適当な記号がこの範囲を小さくする。そのようにして、より少ないビットをメッセージに加える。

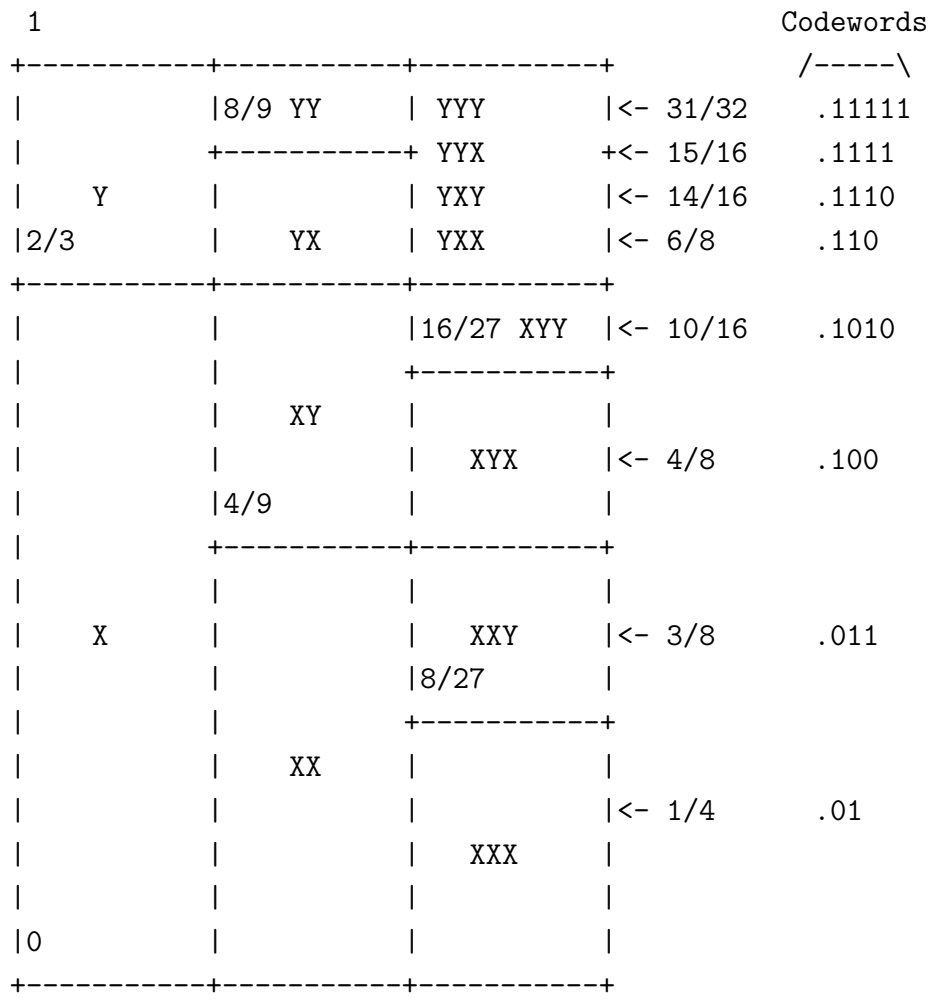


図 3.5: Arithmetic Coding Example; 算術符合の例

As an example of arithmetic coding, illustrated in Figure 3.5, consider an example of two symbols X and Y, with probabilities $\frac{2}{3}$ and $\frac{1}{3}$, respectively. To encode a message, examine the first symbol: If it is an X, choose the lower leftmost partition; if it is a Y, choose the upper. Continuing in this manner for three symbols, we get the codewords shown to the right of the diagram — they can be found by simply taking an appropriate location in the interval for that particular set of symbols and turning it into a binary fraction. (In practice, it is also necessary to add a special end-of-data symbol, not represented in this simple example.)

符号の例は図3.5 に描かれている。確率がそれぞれ $\frac{2}{3}$, $\frac{1}{3}$ である記号 X,Y を考える。このメッセージをエンコードするには、最初の記号を調べる。もし X なら、下段を選び、Y なら上段を選ぶ。3 つの記号に対して同じようにしていくと、上のダイアグラムの右側にあるような符合語が得られる。それらは単純に特定の記号の適切な間隔を取り、2 進分数にすることで得られます。実際には、特別なデータの終わりを示す記号も付け加える必要があります。

3.11 Gray Codes; グレイ符合

Gray Codes are lists of all binary numbers of a certain length such that any two adjacent numbers in the list differ by exactly one bit.[a] A natural instance of such a code is the binary reflected Gray code, since it can be generated in the following manner: Take the Gray code [0, 1]. Write it forwards, then backwards: [0, 1, 1, 0]. Then append 0s to the first half and 1s to the second half: [00, 01, 11, 10]. Continuing, write [00, 01, 11, 10, 10, 11, 01, 00] to obtain: [000, 001, 011, 010, 110, 111, 101, 100], etc.

[a] `http://mathworld.wolfram.com/GrayCode.html`

グレイ符合は二つの隣接した数字がちょうど 1 だけ違うようなある長さの全ての 2 進の数のリストのことです。そのような符合はそれが次のような方法で作られるので、binary reflected Gray Code と呼ばれています。[0,1] のグレイ符合。それを、順番に書き、次に逆の順番で書く。[0,1,1,0] そうしたら、始めの半分には 0 を、残り半分には 1 を付ける。[00,01,11,10] これを続ける。

3.11.1 Baguenaudier, Patience Puzzle, Chinese Rings, Devil's Needle

Known variously as the "Patience Puzzle," "Baguenaudier," or the "Chinese Rings," this commercially available puzzle is related to the "Towers of Hanoi" puzzle. In `emacs` or `mule`, type `M-x hanoi` (M-x is the same as Esc x). One can specify the number of rings with a numeric argument with `Ctrl-u` (Control+u). Example: `Ctrl+u 7 Esc x hanoi` (followed by RETurn).

Patience Puzzle,、Bagauenaudier または、Chinese Rings としてさまざまに知られているように、このパズルは商業的に活用されている。10 このパズルは「ハノイの塔」に関係がある。`emacs` それとも `mule` で `M-x hanoi` と打つ。`C-u` でリングの数を指定することができます。たとえば、`Ctrl+u 7 Esc x hanoi`

Describing a ring on the shuttle as '1' and off as '0' (and remembering the natural convention that the fastest-changing position is the LSB, the starting position for a 6-ring patience puzzle would be 111111_2 or 63_{10}. The corresponding decimal Gray Code sequence is: {**0**, 1, 3, 2, 6, 7, 5, 4, 12, 13, 15, 14, 10, 11, 9, 8, 24, 25, 27, 26, 30, 31, 29, 28, 20, 21, 23, 22, 18, 19, 17, 16, 48, 49, 51, 50, 54, 55, 53, 52, 60, 61, **63**, 62, 58, 59, 57, 56, 40, 41, 43, 42, 46, 47, 45, 44, 36, 37, 39, 38, 34, 35, 33, **32**}, where the starting point is the peak 63 (43/64 of the way through the sequence); the goal is 0 (at the beginning); and the dead end is 32 (at the end). This sequence can be visualized as a "fractal (self-similar) asymmetric wing," as in Figure 3.6.

シャトルの上にある状態を'1'、下にある状態を'0'と表現する。(習慣的に最も変化の多い位置をLSBと呼ぶことに注意、6つの輪の忍耐パズル(中国パズル)の初期位置は111111_2もしくは63_{10}である。対応する10進法グレイコード列は{**0**, 1, 3, 2, 6, 7, 5, 4, 12, 13, 15, 14, 10, 11, 9, 8, 24, 25, 27, 26, 30, 31, 29, 28, 20, 21, 23, 22, 18, 19, 17, 16, 48, 49, 51, 50, 54, 55, 53, 52, 60, 61, **63**, 62, 58, 59, 57, 56, 40, 41, 43, 42, 46, 47, 45, 44, 36, 37, 39, 38, 34, 35, 33, **32**}である。初期位置はピークである63(43/64)ゴールは0である。行き止まりは32である。この列は図3.6のようなフラクタルの左右非対称の羽のように視覚化される。

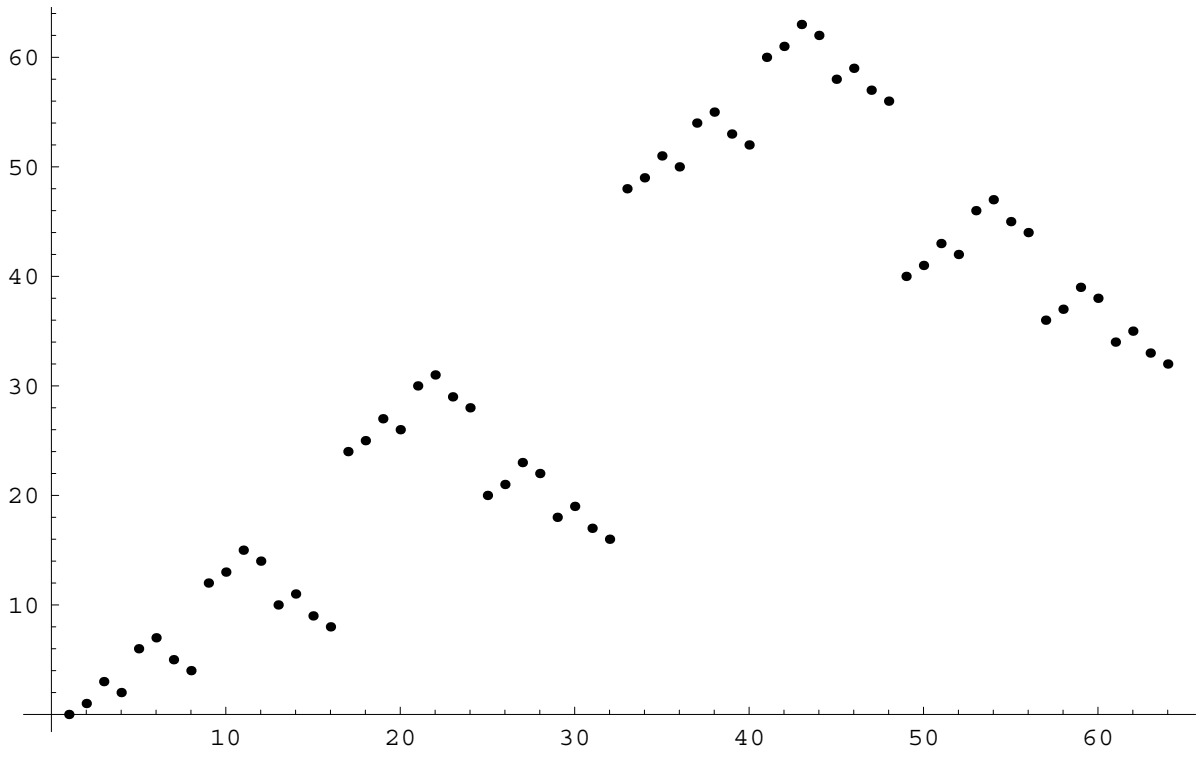


図 3.6: Gray Code Sequence

As seen in that figure (63→61, 31→30, 15→13, 7→6, 3→1), if the number of rings in the puzzle is odd (1, 3, 5, etc.), the correct first move is to take off the first ring, but *if the number of rings is even (2, 4, **6**, etc.), the correct first move is to take off the second ring*

図に見られるように（63→61, 31→30, 15→13, 7→6, 3→1)、パズルのリングの数が奇数ならば、初めに一番目のリングを外すのが正解だが、リングの数が偶数である場合、初めに二番目のリングを外すのが正しい。

第4章　4: Information and Entropy; 情報量とエントロピー

4.1　Information quantity; 情報量

4.1.1　Logarithms; 対数

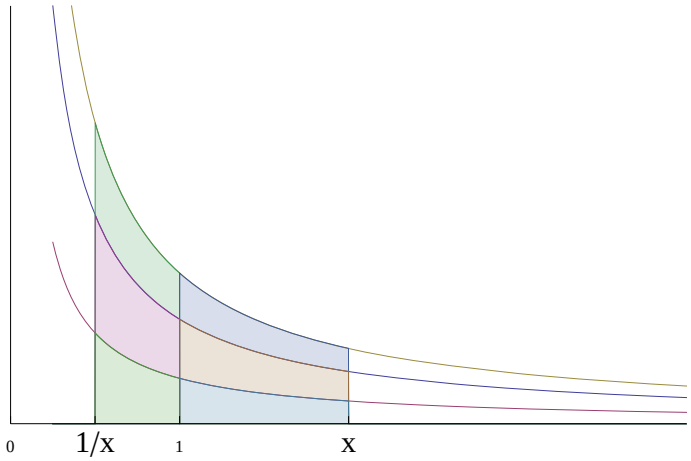


図 4.1: $\{\frac{\log_2(e)}{t}, \frac{1}{t}, \frac{\log_{10}(e)}{t}\}.\frac{d}{dx}\ln(x) = \frac{1}{x}; \ln(x) = \int_1^x(\frac{1}{t})dt \Rightarrow \ln(\frac{1}{2}) = -\ln(2); \frac{d}{dx}\log_{2,e,10}(x) = \frac{1}{x}\log_{2,e,10}(e)$

Table 2.13 reviewed the common logarithmic bases. Figure 4.1 illustrates the definition of the natural logarithm, juxtaposed against analogous scaled inverse hyperbolas for other bases.

図 4.1 は自然対数の定義を表しています。

Recall (§ 2.7.1) that "e," the base of the natural logarithm can be defined by the following equations:

§ 2.7.1 の自然対数の底「e」は以下の式で定義される。

$$e = \lim_{n\to\infty}(1+\frac{1}{n})^n \tag{4.1}$$

$$e^x = 1 + x + \frac{x^2}{2} + \ldots + \frac{x^n}{n!} + \ldots \tag{4.2a}$$

$$= \sum_{n=0}^{\infty}\frac{x^n}{n!} \tag{4.2b}$$

Table 2.13 showed conventions used when the logarithmic base isn't implicit. But since digital computers use two-state logic, unless otherwise stated, we usually will use binary logarithms, normally just written "log", with an implicit base 2.

(表 2.13) 前述の表は対数の底が実質的に述べられていないときの決まりを表わしていますが，デジタルコンピュータでは二進法を使っているので，特に述べられていなければ，大抵「log」で書かれる二の対数を使います.

4.2 Equation which follows information quantity; 情報量の従う方程式

Any measure of information must be invariant across its format. The information of a joint source should be equal to the sum of the information contents of the individual sources.

うに右辺と左辺は等しくなければいけません。結合した情報（左辺）は、それぞれ独立した情報の合計（右辺）と等しくなければいけません。

$$\overbrace{U(MN)}^{\text{parallel}} = \overbrace{U(M)+U(N)}^{\text{serial}} \tag{4.3}$$

The logarithmic function has this quality. For example, being informed of someone's birthday as a month and date together ("in parallel") should have the same amount of information as being notified of those separately ("serially").

対数関数もこの特性を持っています。例えば、誕生日という情報を月と日という形で一緒に持っている場合と、月と日という情報を別々に持っている場合も情報量は同じになるはずです。

$$\begin{aligned} \log ab &= \log a + \log b \\ \log 365 &\approx \log 12 + \log 30 \quad \text{(common logs, base 10)} \\ 2.6 &\approx 1.1 + 1.5 \quad \surd \end{aligned}$$

4.3 Complexity and program length; 系列の複雑さとプログラムの長さ

4.3.1 Kolmogorov Complexity; コルモゴロフ・コンプレキシティ

Kolmogorov, Chaitin, and Solomonoff developed the idea of complexity of a pattern (string, 2D image, ...) as the length of the shortest program for computing it. Kolmogorov complexity is the minimal descriptive length.

コルモゴロフ, チャイティン, と ソロモノフ らはストリングや 2D のイメージを計算するために，最も短いプログラムの長さに関するストリングの複雑さについての考え方を開発しました。コルモゴロフ・コンプレキシティは記述できる最小の長さを表します。

The computer, the most general form of data decompressor, will use such a description to exhibit desired object after finite computation.

もっとも一般的なデータ解凍器であるコンピュータは，有限個の計算の後要求されたオブジェクトを表示するのにこのような記述を用います。

Occam's Razor: the simplest explanation is best.

オッカムの剃刀： 単純な記述であればあるほど良い。

complexity; 複雑さ	computational; コンピュータによる計算	algorithmic; アルゴリズム
domain; 分野	time, space; 時間、空間	descriptive; 記述的
metric; 測定基準	program execution; プログラムの実行時間	program length; プログラムの長さ

表 4.1: Complexity; 複雑さ

The Mandelbrot set (Figure 4.2), even though it seems very complicated, has a very simple description and hence, a low Kolmogorov complexity.

マンデルブロー集合 (図 4.2)、複雑に見えますが，式は非常に単純です。
従って，低いコルモゴロフ・コンプレキシティです。

$$z \mapsto z^2 + C, z_\infty < \infty \tag{4.4}$$

For example,

$$C = 1 : 0, 1, 2, 5, 26, \ldots, \infty$$
$$C = i : 0, i, (-1+i), -i, (-1+i), -i, \ldots$$

4.4 Characteristics of Entropy; エントロピーの関数としての性質

Entropy is a measure of the $\left\{ \begin{array}{l} \text{uncertainty (disorder)} \\ \text{surprise} \end{array} \right.$ of a system. Information is reduction in uncertainty.

エントロピーは $\left\{ \begin{array}{l} \text{不確定} \\ \text{驚き} \end{array} \right.$ の尺度です。情報とは不確定の減少である。

Before learning the value of a random variable, like tomorrow's weather, we are uncertain about it. When the value is revealed to us — a forecast or prediction of rain, for instance — we are to some extent surprised. And we use this reduction in uncertainty as information, say by carrying an umbrella.

例えば明日の天気を知らないように，私達は変数の値を知る前はそれについての確信はありません。その値が現れた時に初めて私達は驚くのです。たとえば、降水確率で、私たちはそのデータを「傘を持っていきなさい」という情報として使います。

If X is a random variable with values x, each of probability $p(x)$,

もし X が各 $p(x)$ の値 x を持つ確率変数であるなら

$$H(X) \stackrel{\Delta}{=} -\sum_{x \in \mathcal{X}} p(x) \log p(x) \tag{4.5a}$$
$$= \sum_x p(x) \log \frac{1}{p(x)} \tag{4.5b}$$
$$= E_p[\log \frac{1}{p(x)}] \tag{4.5c}$$

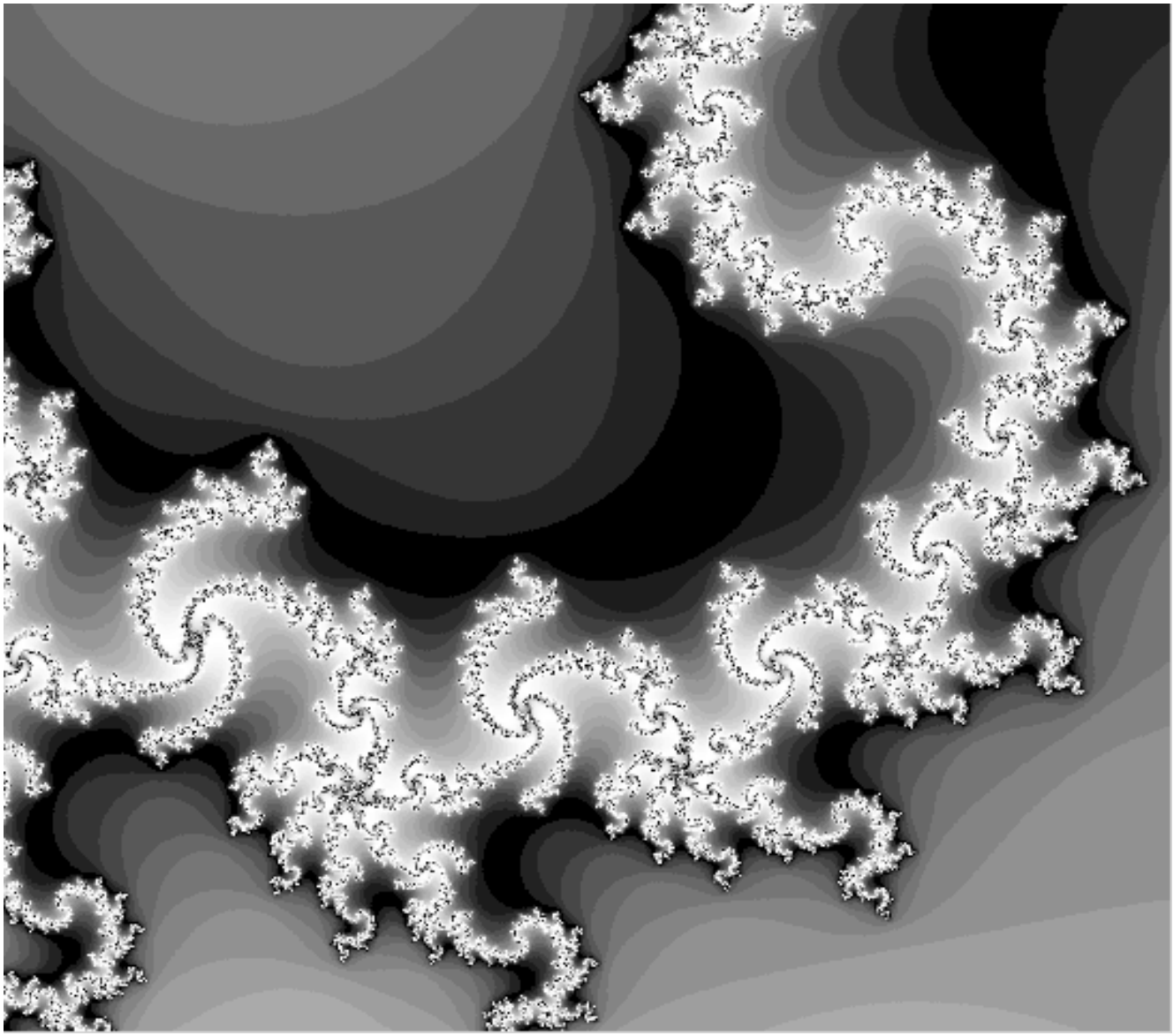


図 4.2: Part of the Mandelbrot Set, which has low algorithmic (Kolmogorov) complexity; 低いコルモゴロフ・コンプレキシティをもったマンデルブロー集合の一部

where $E_p[f(x)]$ means the expected (average) value of $f(x)$ with probability distribution p of x.

$E_p[f(x)]$ は $f(x)$ の期待値です。

Note that entropy is a function of the probabilities $p(x)$, not of the actual values or meanings of x.

注意！エントロピーとは，確率の関数 $p(x)$ で，実際の変数 x ではありません。

Also note that

次のことにも注意する。

$$\sum_x p(x) = 1 \tag{4.6}$$

since every trial yields a result (event), per §1.2.

どの試行も結果を生じるからである (§1.2)。

Unless otherwise stated, units of entropy are bits. Conversion between bases is straightforward, since (via chain rule for logarithms)

特に述べられていなければ，エントロピーの構成単位はビットです。底の変換は以下のような理由から簡単にできます。

$$\log_b p \;=\; \log_b a \;\log_a p \tag{4.7}$$
$$\tag{4.8}$$

For example, to convert from nats to bits,

例えば，ナット (対数の底が e) からビット (対数の底が 2) に変換するためには、

$$\overbrace{\log_2 p}^{\text{bits}} = \overbrace{\log_2 e}^{\text{bits/nats } \approx 1.4} \overbrace{\ln p}^{\text{nats}} \tag{4.9}$$

(This resembles the chain rule for joint probability: $p(x, y) = p(x)p(y|x)$.)

（これは同時確率のチェインルールと共通点がある。）

4.4.1 Equiprobable Distribution; 等率発生分布

A fair die can land in any of M states with equal probability $1/M$. Using eqn. (4.5a), the entropy of a toss of such a die can be calculated:

いかさまの賽でなければ，M の目は確率 $1/M$ で出ることになります。賽を投げることのエントロピーは，eqn. (4.5a) を使って計算できます。

$$\begin{aligned} H(\frac{1}{M}, \frac{1}{M}, \dots) &= -\sum_{i=1}^{M} \frac{1}{M} \log \frac{1}{M} && \text{(4.10a)} \\ &= -\frac{1}{M} \sum \log \frac{1}{M} && \text{(4.10b)} \\ &= -\frac{1}{M} M \log \frac{1}{M} && \text{(4.10c)} \\ &= -\log \frac{1}{M} && \text{(4.10d)} \\ &= \log M && \text{(4.10e)} \end{aligned}$$

4.4.1.1 Trivial Example: Computer Memory

$$H(1\,\text{GB}) = \log_2 \underbrace{2^{\overbrace{2^{30}2^3}^{1\,\text{GB}}}}_{\text{binary}} = 2^{33} = (2^3)^{11} \text{ bits (actually } 8\,,589\,,934\;\overbrace{,592}^{\text{thousand}}\;\overbrace{}^{\text{million}}\;\overbrace{}^{\text{billion}}\text{)}$$

4.4.1.2 Example: Fair Coin; 例： 公平な硬貨

A coin is like a 2-sided die:

硬貨は二つの目しかない賽と似ています。

Table Table 4.2 shows some equiprobable distribution entropies.

$$\begin{aligned} H\left(\frac{1}{2}, 1-\frac{1}{2}\right) &= -\frac{1}{2}\log\frac{1}{2} - (1-\frac{1}{2})\log(1-\frac{1}{2}) \\ &= -\log\frac{1}{2} \\ &= 1 \text{ bit} \end{aligned}$$

object 対象	equiprobable states 等率発生値	entropy エントロピー				
fair coin; 公平な硬貨	2	$H(\frac{1}{2},\frac{1}{2})$	$=$	$\log 2$	$=$	1 bit
fair cubic die; さいころ	6	$H(\frac{1}{6},\frac{1}{6},\frac{1}{6},\frac{1}{6},\frac{1}{6},\frac{1}{6})$	$=$	$\log 6$	$\approx$	2.6 bits
months; 月	12	$H(\frac{1}{12},\ldots)$	$=$	$\log 12 = \log 6 + \log 2$	$\approx$	3.6 bits
deck of cards; トランプ	52	$H(\frac{1}{52},\frac{1}{52},\ldots)$	$=$	$\log 52$	$\approx$	5.7 bits

表 4.2: Entropies of Some Common Equiprobable Distributions; 一般的な等率発生分布のエントロピー

4.4.2 Nonequiprobable Two-State Distribution: Unfair Coin; 不公平な硬貨

The toss of an unfair coin can be modeled as a random variable X such that

不公平な硬貨を投げることは，変数 x のモデルとして考えられます。

$$X = \begin{cases} \text{heads; 表} & \text{probability } p \\ \text{tails; 裏} & \text{probability } 1-p \end{cases}$$

Alternatively, using [大 93]'s notation,

別の方法として，教科書の手法も使えます,

$$X = \begin{pmatrix} 1 & 0 \\ p & 1-p \end{pmatrix} \tag{4.12}$$

$$H(X) = H(p, 1-p) = -p\log p - (1-p)\log(1-p) \triangleq H_2(p) \tag{4.13}$$

That is, "$H_2(p)$" is special notation for a 2-state system, since there is only a single degree of freedom.

1 自由度しかないため、「$H_2(p)$」は二値システムを考える為の特別な手法です。

4.4.3 Diversion: Zero Probabilities; 偏向： 確率ゼロ

When we write the probability distribution of a random variable, we don't bother with impossible states, i.e., those states with zero chance, like a coin landing on its edge. Extending eqn. (4.12) above,

私達が任意の変数の分布状態の確率を書く時，私達は不可能な状態については考えません。コインが縁で立つようなそのような状態の確率はゼロとなります。上の eqn. (4.12) を拡張して以下のように書くこととします。

$$X = \begin{pmatrix} 1\ (\text{heads}) & 0\ (\text{tails}) & ?\ (\text{edge}) \\ p & 1-p & 0 \end{pmatrix}$$

How would the inclusion of such zero probability states affect entropy calculation?

エントロピーの計算に 0 の確率を含めることはどのような影響があるのでしょうか？

$$H(p, 1-p, 0) \quad = -p\log p - (1-p)\log(1-p) - 0\log 0 \tag{4.14}$$

But what is $0 \log 0$ (since $\log 0 = -\infty$, as seen in Figure 2.9)?

$0 \log 0$ とは何でしょうか ($\log 0 = -\infty$ 参考, 図 2.9)？

By looking at the graph (Figure 4.3) of $-p \log p$, we can observe by continuity that $\lim_{p \to 0}(-p \log p) = 0$.

$-p \log p$ のグラフ (図 4.3) を見てもらうと, $\lim_{p \to 0}(-p \log p) = 0$ が観測できます。

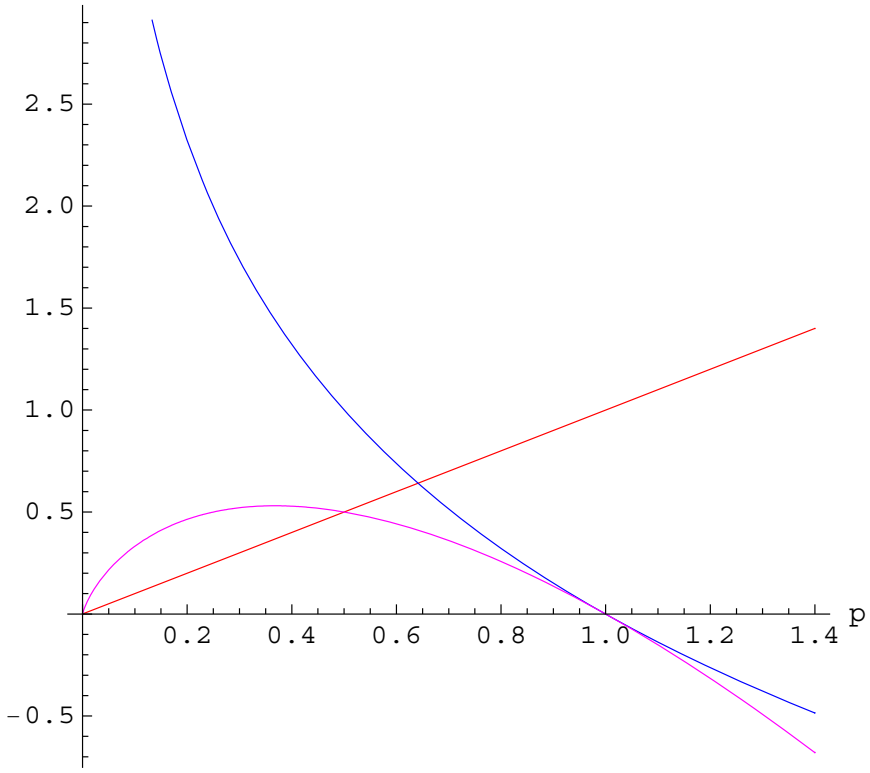


図 4.3: [p, $\log(1/p)$, and] $p \log(1/p) = -p \log p$. Note that $\lim_{p \to 0}(-p \log p) = 0$ as the linear function overpowers the logarithmic.

A more rigorous approach invokes l'Hôpital's rule, which confirms that linear functions 'overpower' logarithmic functions, as anticipated by Table 2.6.1.

表 2.6.1 から予想されるように、一次関数が対数関数より「勝っている」ことをより厳格な方法で確かめるためにロピタル（l'Hôpital）の定理を用います。

$$\lim_{t \to a} \frac{f(t)}{g(t)} = \lim_{t \to a} \frac{f'(t)}{g'(t)}, \tag{4.15}$$

$$\text{where } \circ'(t) = \tfrac{d}{dt} \circ (t).$$

Applying this rule to find $-p \log p$ at $p = 0$,

$p = 0$ の 時 の $-p \log p$ 値を見つけるためこの定理を適用して,

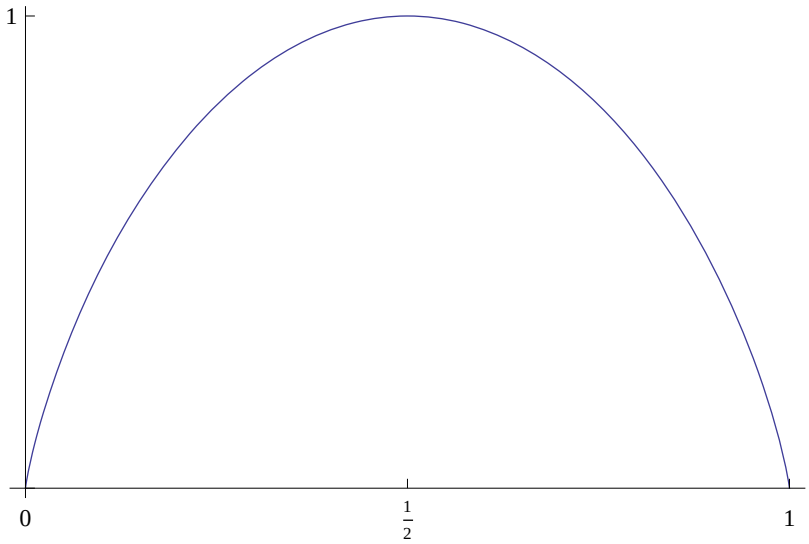


図 4.4: Dyadic entropy $H_2(p) = H(p, 1-p)$

$$
\begin{aligned}
\lim_{p\to 0} -p\log_2 p &= \lim_{p\to 0} \frac{-\log_2 p}{1/p} \\
&= \lim_{p\to 0} \frac{\frac{d}{dp}(-\log_2 p)}{\frac{d}{dp}(p^{-1})} \\
&= \lim_{p\to 0} \frac{\frac{d}{dp}((-\ln p)(\log_2 e))}{-p^{-2}} \\
&= \lim_{p\to 0} \frac{-p^{-1}\log_2 e}{-p^{-2}} \\
&= \lim_{p\to 0} (\frac{p^2}{p}(\log_2 e)) \\
&= 0 \quad \surd
\end{aligned}
$$

4.4.4 Properties of Entropy; エントロピーの特性

- Since p is within closed unit interval, $0 \le p(x) \le 1$, $\log(\frac{1}{p(x)}) \ge 0$, so $H_2(x) \ge 0$.
- $H_2(p)$ is convex.
- $H_2(0) = H_2(1) = 0$. When $p = 0$ or $p = 1$, the variable is not random, and there is no uncertainty. "The constant complainer is soon ignored."
- The dyadic entropy function is symmetric, with maximum in middle:

- $0 \le p(x) \le 1$, よって $\log(\frac{1}{p(x)}) \ge 0$, それで $H_2(x) \ge 0$。
- $H_2(p)$ は凹面(おうめん)です。
- $H_2(0) = H_2(1) = 0$。 $p = 0$ or $p = 1$ の時の値はランダムではなく不確かなものは存在しません。「いつも不平を言っている人は無視される。」
- 二値のエントロピー関数は最大値が真ん中にあり、対称である:

$$H_2(\frac{1}{2}) = H(\frac{1}{2}, \frac{1}{2}) = 1 \tag{4.17}$$

$$\begin{aligned}
\frac{d}{dp}(p\log_2 p + (1-p)\log_2(1-p) &= p\frac{1}{p} + \log_2 p + (1-p)(\frac{1}{1-p})(-1) + \log_2(1-p)(-1) \\
&= 1 + \log_2 p - 1 - \log_2(1-p) \\
&= \log_2 p - \log_2(1-p) \\
\log_2 p - \log_2(1-p) &= 0 \\
p &= 1-p \\
p &= \frac{1}{2}
\end{aligned}$$

4.4.5 Example: 4-state Random Variable; 例： 四つの状態のある任意の変数

$$S = \begin{cases} a & \text{with probability} 1/2 \\ b & \text{with probability} 1/4 \\ c & \text{with probability} 1/8 \\ d & \text{with probability} 1/8 \end{cases}$$

Alternatively, using [大 93]'s notation, 別の方法として，教科書の手法も使えます,

$$\begin{aligned}
S &= \begin{pmatrix} a & b & c & d \\ \frac{1}{2} & \frac{1}{4} & \frac{1}{8} & \frac{1}{8} \end{pmatrix} \\
H(S) &= -\frac{1}{2}\log\left(\frac{1}{2}\right) - \frac{1}{4}\log\left(\frac{1}{4}\right) - \frac{1}{8}\log\left(\frac{1}{8}\right) - \frac{1}{8}\log\left(\frac{1}{8}\right) \\
&= \frac{7}{4} \text{ bits}
\end{aligned}$$

The entropy (in bits) of a random variable corresponds to the minimum expected number of (binary) questions (diamonds [◊s] in the flowchart of Figure 4.5 below) required to resolve or determine its value. To calculate the expected number of yes/no questions needed to determine a random variable, sum the products of the respective probabilities for each value and the number of questions needed to determine that value.

任意の変数のエントロピーは，その値を決めるのに必要とされる質問の最小期待値に一致します (図 4.5 のフローチャート中の ◊ 参照)。Yes か No かの質問の期待値を計算する為に任意の値を決めなければいけません。その値を決めるのに必要な質問の数とその値をかけた，それぞれの結果を合計します。

$$
\begin{aligned}
\text{avg. number of questions to determine } s &= p(s=a)\ \text{(number of questions to reach a)} \\
&\quad + p(s=b)\ \text{(number of questions to reach b)} \\
&\quad + p(s=c)\ \text{(number of questions to reach c)} \\
&\quad + p(s=d)\ \text{(number of questions to reach d)} \\
&= \frac{1}{2}(1\ \text{question}) \\
&\quad + \frac{1}{4}(2\ \text{questions}) \\
&\quad + \frac{1}{8}(3\ \text{questions}) \\
&\quad + \frac{1}{8}(3\ \text{questions}) \\
&= \frac{7}{4}\ \text{questions}\ \surd
\end{aligned}
$$

For this particular example (in which the probabilities are powers of $\frac{1}{2}$), the entropy can be achived by Huffman coding, but other distributions generally require more complicated encoding. For instance, a first-order encoding of $(\frac{1}{3}, \frac{2}{3})$ would use a single bit to distinguish the codes, but higher-order extensions could use Huffman coding to approach the bits/symbol limit set by the entropy.

この特定の例（確率が 1/2 の乗数であるもの）においては、エントロピーはハフマン符号を用いて圧縮してまとめることができるが、他の分布は一般的に複雑な符号化を必要とする。たとえば、$(\frac{1}{3}, \frac{2}{3})$ の一次の符号化ではコードを識別するためにただ 1 ビットが使用されるが、高次に拡張されたものではエントロピーによる制限に近づけるためにハフマン符号を使用する。

4.5 Conditional Entropy; 条件付き エントロピー

4.5.1 Joint Entropy; 結合 エントロピー

Entropy can be generalized for systems of multiple variables or dimensions. Usually there will be some relationship (correlation) between the variables, as shown Table 4.4. (Note that the joint distribution is not a transition matrix.)

エントロピーは多項または多次元のシステムにおいて一般的に扱うことができる。通常、変数の間には何らかの関係 (相関関係) があります。(表 4.4) （同時分布は遷移行列ではないことに注意してください。）

4.5.1.1 Joint Distributions; 結合分布

Q. When does $p(x,y) = p(x)p(y)$? A. Only when $p(x)p(y|x) = p(y)p(x|y) = p(x)p(y)$: i.e., when X and Y are independent, like separate tosses of a coin or rolls of a die. Otherwise, each variable serves as a hint for the other, as in Table 4.4, and the joint probability is given by the chain rule for conditional probability.

Q. いつ $p(x,y) = p(x)p(y)$ となるのか？A. $p(x)p(y|x) = p(y)p(x|y) = p(x)p(y)$ の時になります，それは X と Y が別々に硬貨を投げるような時や二つの賽の裏表が違う時のように独立している時です。そうでなければ，変数の値がもう一方の値のヒントになります (表 4.4)。そして、同時確率は条件付き確率における連鎖法則によって与えられます。

Dimensionality	Space	Data Structure	Example	Measure
0D	point	scalar: primitive	magnitude, speed, duration	
1D	line	vector	velocity, $\vec{v} = (x, y, z)$	length; 長さ
2D	plane, surface, face, polygon	matrix	table, $M = \begin{pmatrix} m_{11} & m_{12} & \dots & m_{1k} \\ m_{21} & m_{22} & \dots & m_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ m_{j1} & m_{j2} & \dots & m_{jk} \end{pmatrix}$	area; 面積
3D	solid, polyhedron, volume			volume; 体積
⋮		array		

表 4.3: Dimensions and Arrays. (Note: "primitives" in modern programming languages are types like `int`, `float`, `enumerated`, ...); 次元と配列 (注：現代プログラミング言語における基本要素は `int`, `float`, `enumerated` といった型である。)

			marginal distribution; 周辺分布
	uncloudy	cloudy	
not rainy	.5	.25	.75
rainy	0	.25	.25
marginal distribution	.5	.5	1

表 4.4: Example of Dependent Variables (cloudiness is hint about rain); 例題 従属変数 (降水確率): rainy ⇒ cloudy and uncloudy ⇒ not rainy: $H(\text{rain}) = H(\frac{1}{4}, \frac{3}{4}) \approx 0.8$; $H(\text{Rain}|\text{Cloudiness}) = .5 \cdot 0 + .5 \cdot 1 = \frac{1}{2}$ bit; $H(\text{Cloudiness}|\text{Rain}) = \frac{3}{4}H(\frac{1}{3}, \frac{2}{3}) + \frac{1}{4}H(0, 1) \approx \frac{3}{4} \cdot 0.9 \approx 0.7$ bits

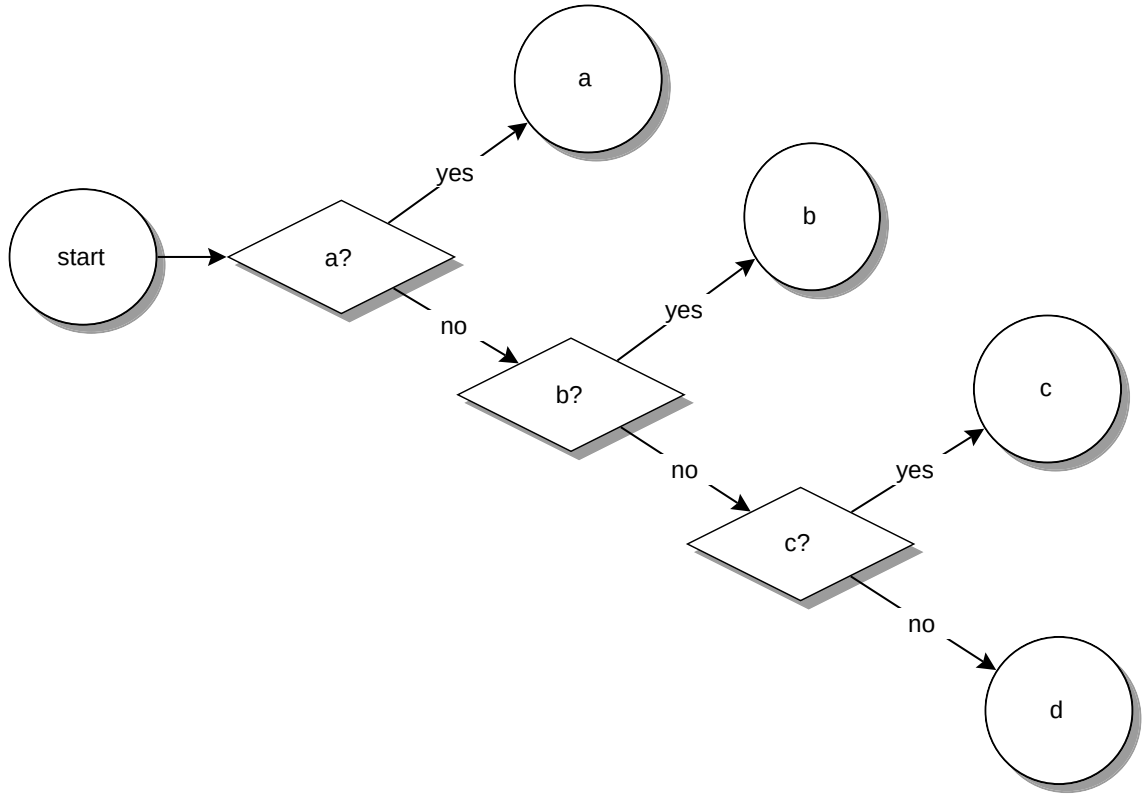


図 4.5: Flow Chart for Determining Value of Random 4-state Variable; 任意の 4 変数の値を決めるためのフローチャート

$$
\begin{aligned}
p(x,y) &= p(y)p(x|y) && \text{(4.18a)}\\
&= p(x)p(y|x) && \text{(4.18b)}
\end{aligned}
$$

4.5.1.2 Joint Entropy; 結合エントロピー

$$
\begin{aligned}
H(X,Y) &= -\sum_x \sum_y p(x,y)\log p(x,y) && \text{(4.19a)}\\
&= -E[\log p(x,y)] && \text{(4.19b)}\\
&= E[\frac{1}{\log p(x,y)}] && \text{(4.19c)}
\end{aligned}
$$

4.5.1.3 Example: Joint Distribution; 例：結合分布

Let X and Y both take on values drawn from $\{\clubsuit, \diamondsuit, \heartsuit, \spadesuit\}$, according to this probability distribution:

$\{\clubsuit, \diamondsuit, \heartsuit, \spadesuit\}$ と値を入れそれぞれ X と Y を確率順に沿って書き出します。

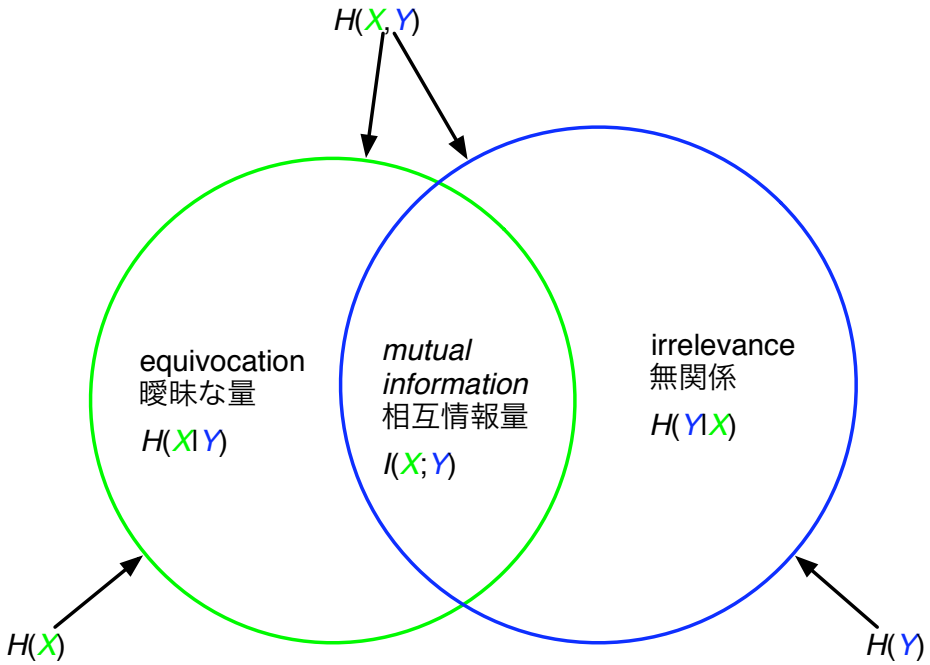


図 4.6: Venn Diagram: Joint Entropy and Mutual Information; ベン図: 結合エントロピーと相互情報量

	X				marginal distribution
Y		♣	♢	♡	♠	
	♣	1/8	1/16	1/32	1/32	1/4
	♢	1/16	1/8	1/32	1/32	1/4
	♡	1/16	1/16	1/16	1/16	1/4
	♠	1/4	0	0	0	1/4
marginal distribution		1/2	1/4	1/8	1/8	(1)

Since the two variables X and Y can be thought of as forming a composite variable, this is equivalent to:

二つの変数 X と Y は整えられた合成変数と考えることができるので，これらは以下に等しくなります。

♣,♣	♢,♣	♡,♣	♠,♣	♣,♢	♢,♢	♡,♢	♠,♢	♣,♡	♢,♡	♡,♡	♠,♡	♣,♠	♢,♠	♡,♠	♠,♠
1/8	1/16	1/32	1/32	1/16	1/8	1/32	1/32	1/16	1/16	1/16	1/16	1/4	0	0	0

The respective entropies for X, Y, and (X,Y) are easily calculated (from the marginal probabilities):

私達はそれぞれ X, Y, と (X,Y) についてのエントロピーを計算することができます。

$$\begin{aligned} H(X) &= H(1/2, 1/4, 1/8, 1/8) \quad \text{[using the marginal distribution of } X\text{]} \\ &= 7/4 \text{ bits [remembering 4-state example above in § 4.4.5]} \end{aligned}$$

$$
\begin{aligned}
H(Y) &= H(1/4, 1/4, 1/4, 1/4) \quad \text{[using the marginal distribution of } Y\text{]} \\
&= 2 \text{ bits [using eqn. (4.10e) above]}
\end{aligned}
$$

$$
\begin{aligned}
H(X,Y) &= H(1/8, 1/16, 1/32, 1/32, \\
&\quad 1/16, 1/8, 1/32, 1/32, \\
&\quad 1/16, 1/16, 1/16, 1/16, \\
&\quad 1/4, 0, 0, 0) \\
&= -1/8 \log(1/8) - 1/16 \log(1/16) - 1/32 \log(1/32) - 1/32 \log(1/32) \\
&\quad -1/16 \log(1/16) - 1/8 \log(1/8) - 1/32 \log(1/32) - 1/32 \log(1/32) \\
&\quad -1/16 \log(1/16) - 1/16 \log(1/16) - 1/16 \log(1/16) - 1/16 \log(1/16) \\
&\quad -1/4 \log(1/4) - 0 \log(0) - 0 \log(0) - 0 \log(0) \\
&= 3/8 + 4/16 + 5/32 + 5/32 \\
&\quad +4/16 + 3/8 + 5/32 + 5/32 \\
&\quad +4/16 + 4/16 + 4/16 + 4/16 \\
&\quad +2/4 + 0 + 0 + 0 \\
&= 27/8 \text{ bits}
\end{aligned}
$$

4.5.2 Conditional Entropy and the Chain Rule; 条件付きエントロピーと結合法則

A chain, such as a MIDI (computer ↔ keyboard ↔ sound module ↔ mixer ...) "daisy chain," makes a linked list by tying each element to its neighbor.

MIDI (computer ↔ keyboard ↔ sound module ↔ mixer ...) のように連結した鎖は、それぞれの要素をそれの隣に結合することでできるリンクトリストを作ります。

Conditional entropy is the entropy of a random variable, given another random variable:

条件付きエントロピーとは，他の任意変数が与えられた任意変数のエントロピーのことです。

$$
\begin{aligned}
H(\overset{\text{"given"}}{Y \mid X}) &\triangleq \sum_{x \in \mathcal{X}} p(x) H(Y|X=x) && \text{(4.20a)} \\
&= -\sum_{x \in \mathcal{X}} p(x) \sum_{y \in \mathcal{Y}} p(y|x) \log p(y|x) && \text{(4.20b)} \\
&= -\sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x,y) \log p(y|x) && \text{(4.20c)}
\end{aligned}
$$

The joint entropy of a pair of random variables *does not* generally equal the sum of the respective entropies.

一組の任意変数の結合エントロピーはそれぞれのエントロピーの合計では**ない**ことに注意。

$$
H(X,Y) \neq H(X) + H(Y) \tag{4.21}
$$

Chain rule for entropy: the entropy of a pair of random variables is the entropy of one, plus the conditional entropy of the other. (See Figure 4.6.)

エントロピーの結合法則：一組の任意変数によるエントロピーは，一方の変数のエントロピーと。もうひとつの変数の条件付きエントロピーの和である。（図 4.6）

$$H(X,Y) = -\sum_{x\in\mathcal{X},y\in\mathcal{Y}} p(x,y)\log p(x,y) \tag{4.22a}$$

Bayes' rule (eqn. (1.36)), chain rule for probability

$$= -\sum p(x,y)\log\left(p(y|x)p(x)\right) \tag{4.22b}$$

log of product is sum of logs

$$= -\sum p(x,y)\log p(y|x) - \sum p(x,y)\log p(x) \tag{4.22c}$$

definition of conditional entropy (eqn. (4.20c))

$$= H(Y|X) - \sum p(x,y)\log p(x) \tag{4.22d}$$

split summation: separate variables

$$= H(Y|X) - \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(x) \tag{4.22e}$$

$\log p(x)$ doesn't depend upon y

$$= H(Y|X) - \sum_{x\in\mathcal{X}} \log p(x)\sum_{y\in\mathcal{Y}} p(x,y) \tag{4.22f}$$

$\sum_y p(y) = 1$

$$= H(Y|X) - \sum_{x\in\mathcal{X}} \left(\log p(x)\right) p(x) \tag{4.22g}$$

definition of entropy

$$= H(Y|X) + H(X) \tag{4.22h}$$

commutative (symmetric)

$$= H(X|Y) + H(Y) \tag{4.22i}$$

4.5.2.1 Example: Conditional Entropy; 例： 条件付きエントロピー

(continued from § 4.5.1.3)

Normalize the rows (or columns) of the joint distribution to calculate the case-by-case entropies.

個別的のエントロピーを計算するために、結合分布の列（または行）を正規化する。

$$
\begin{aligned}
H(X|Y) &= \sum_{y\in\{\clubsuit,\diamondsuit,\heartsuit,\spadesuit\}} p(Y=y)H(X|Y=y) \\
&= \frac{1}{4}\overbrace{H\left(\frac{1/8,1/16,1/32,1/32}{1/8+1/16+1/32+1/32}\right)}^{H(X|y=\clubsuit)} + \frac{1}{4}\overbrace{H\left(\frac{1/16,1/8,1/32,1/32}{1/16+1/8+1/32+1/32}\right)}^{H(X|y=\diamondsuit)} \\
&\quad + \frac{1}{4}\overbrace{H\left(\frac{1/16,1/16,1/16,1/16}{1/16+1/16+1/16+1/16}\right)}^{H(X|y=\heartsuit)} + \frac{1}{4}\overbrace{H\left(\frac{1/4,0,0,0}{1/4+0+0+0}\right)}^{H(X|y=\spadesuit)} \\
&= \frac{1}{4}\overbrace{H(\frac{1}{2},\frac{1}{4},\frac{1}{8},\frac{1}{8})}^{H(X|y=\clubsuit)} + \frac{1}{4}\overbrace{H(\frac{1}{4},\frac{1}{2},\frac{1}{8},\frac{1}{8})}^{H(X|y=\diamondsuit)} + \frac{1}{4}\overbrace{H(\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4})}^{H(X|y=\heartsuit)} + \frac{1}{4}\overbrace{H(1,0,0,0)}^{H(X|y=\spadesuit)} \\
&= \frac{1}{4}\overbrace{(-1/2\log(1/2)-1/4\log(1/4)-1/8\log(1/8)-1/8\log(1/8))}^{\frac{7}{4}\text{ bits}} \\
&\quad + \frac{1}{4}\overbrace{(-1/4\log(1/4)-1/2\log(1/2)-1/8\log(1/8)-1/8\log(1/8))}^{\frac{7}{4}\text{ bits}} \\
&\quad + \frac{1}{4}\overbrace{(-1/4\log(1/4)-1/4\log(1/4)-1/4\log(1/4)-1/4\log(1/4))}^{2\text{ bits}} \\
&\quad + \frac{1}{4}\overbrace{(-1\log(1)-0\log(0)-0\log(0)-0\log(0))}^{0\text{ bits}} \\
&= 1/4[(-1/2)(-1)+(-1/4)(-2)+(-1/8)(-3)+(-1/8)(-3)] \\
&\quad +1/4\ [(-1/4)(-2)+(-1/2)(-1)+(-1/8)(-3)+(-1/8)(-3)] \\
&\quad +1/4\ [4(-1/4)(-2)] \\
&\quad +1/4\ [0-0-0-0] \\
&= 11/8 \text{ bits}
\end{aligned}
$$

This result is consistent with that obtained by the chain rule:

$$
\begin{aligned}
H(X|Y) &= H(X,Y) - H(Y) \\
&= 27/8 - 16/8 = 11/8 \quad \surd
\end{aligned}
$$

Exercise: calculate $H(Y|X)$ directly, and show that it confirms the chain rule, $H(X,Y) = H(X) + H(Y|X)$.

演習： $H(Y|X)$ を直接計算せよ。また結合法則の確認を示せ、$H(X,Y) = H(X) + H(Y|X)$。

4.5.2.2 Conditioning reduces entropy on the average; 平均的な条件付き減少エントロピー

$$
H(X|Y=y) \overset{>?}{\underset{<?}{=?}} H(X) \tag{4.23}
$$

but

しかし

$$H(X|Y) \leq H(X) \tag{4.24}$$

with equality iff (if and only if) X and Y are independent.

等号が成り立つのは，X と Y が 互いに独立している時のみです。

Knowing another random variable Y reduces the uncertainty in X *on the average*.

他の任意変数 Y が知ることによってのみ，**平均的に** X の不確定部分を減らすことができます。

For example, in a court case, specific new evidence might increase the uncertainty, but on the average evidence decreases uncertainty.

例えば裁判のとき，特別な新しい証拠によって不確定要素は増えるかもしれなせんが，平均的な証拠は不確定要素を減らしていきます。

4.5.2.3 Example: Joint Distribution and Conditional (Relative) Entropy; 例： 結合分配と条件付き（相対的）エントロピー

$p(x,y)$		X (precipitation)		marginal distribution
		rainy	dry	
Y	not cloudy	0	clear: $\frac{3}{4}$	$p(y = \text{not cloudy}) = \frac{3}{4}$
	cloudy	rainy: $\frac{1}{8}$	overcast: $\frac{1}{8}$	$p(y = \text{cloudy}) = \frac{1}{4}$
marginal distribution		$p(x = \text{rainy}) = \frac{1}{8}$	$p(x = \text{dry}) = \frac{7}{8}$	(1)

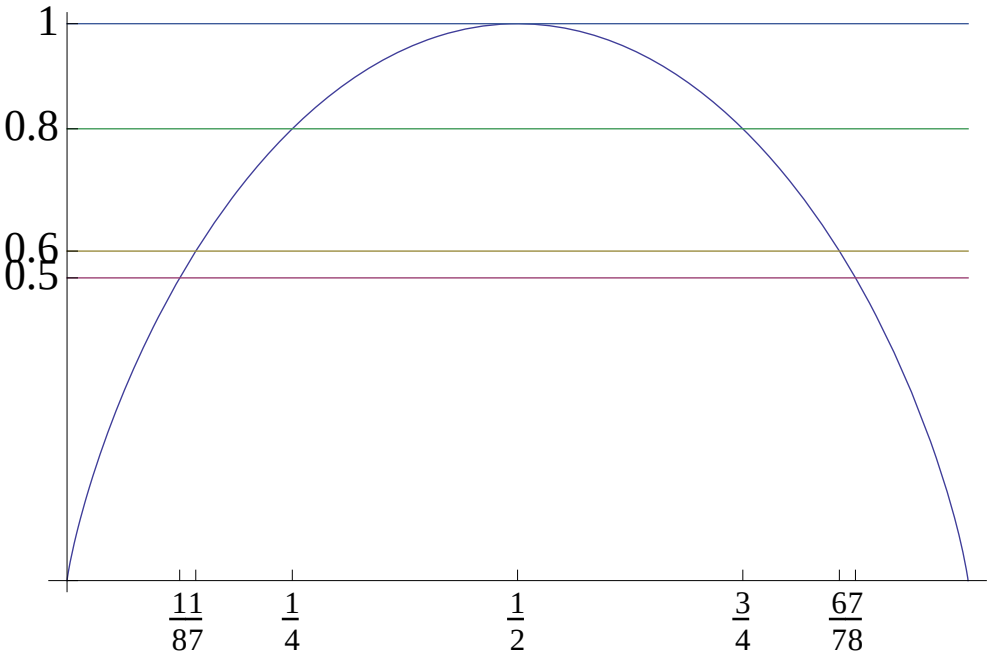


図 4.7: $H(X) = H(\frac{1}{8}, \frac{7}{8}) \approx .5$ bits; $H(X) = H(\frac{1}{7}, \frac{6}{7}) \approx .6$ bits; $H(Y) = H(\frac{1}{4}, \frac{3}{4}) \approx .8$ bits; $H(X,Y) = H(\frac{3}{4}, \frac{1}{8}, \frac{1}{8}) \approx 1.1$ bits;

$$\begin{aligned} H(X|y = \text{not cloudy}) &= H\left(\frac{0}{0+\frac{3}{4}}, \frac{\frac{3}{4}}{0+\frac{3}{4}}\right) = H(0,1) \\ &= 0 \text{ bits } [< H(X) : \text{entropy decreased; エントロピーが減少した}] \end{aligned}$$

$$H(X|y=\text{cloudy}) = H\left(\frac{\frac{1}{8}}{\frac{1}{8}+\frac{1}{8}}, \frac{\frac{1}{8}}{\frac{1}{8}+\frac{1}{8}}\right) = H\left(\frac{1}{2}, \frac{1}{2}\right)$$
$$= 1 \text{ bit } [> H(X) : \text{entropy increased!; エントロピーが増加した!}]$$

$$H(X|Y) = p(y=\text{not cloudy})\, H(X|y=\text{not cloudy}) + p(y=\text{cloudy})\, H(X|y=\text{cloudy})$$
$$= \frac{3}{4}(0) + \frac{1}{4}(1)$$
$$= .25 \text{ bits } [< H(X) : \text{overall entropy decrease; 全体的なエントロピーは増加する}] \ \surd$$

4.6 Mutual Information; 相互情報量

Recall that entropy is defined as

エントロピーはこのように定義されていたことを思い出して

$$H(X) \triangleq \sum_x p(x) \log \frac{1}{p(x)} \tag{4.25}$$

Mutual information $I(X;Y)$ is the reduction in the uncertainty of X due to knowledge of Y. (Note: [大 93] uses the equivalent notation "I(X,Y)" for "I(X;Y)".) If X and Y are independent, $p(x,y) = P(x)p(y)$ and $I(X,Y) = 0$.

相互情報量 $I(X;Y)$ は Y を知ることによって得られる X の不確定部分における減少のことです。(注：[大 93] では,「I(X,Y)」が「I(X;Y)」の同義として記述されています。) もし X と Y が独立している場合、$p(x,y) = P(x)p(y)$ そして $I(X,Y) = 0$ となります。

$$I(X;Y) = \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \tag{4.26a}$$

Bayes' rule (eqn. (1.36)); note that if X and Y are unrelated, $p(x,y) = p(x)p(y)$ and $I(X;Y) = 0$

$$= \sum p(x,y) \log \frac{p(x|y)}{p(x)} \tag{4.26b}$$

log of fraction is difference of logs

$$= -\sum_x p(x) \log p(x) - (-\sum_{x,y} p(x,y) \log p(x|y)) \tag{4.26c}$$

definition of entropy and conditional entropy

$$= H(X) - H(X|Y) \tag{4.26d}$$

Mutual information is the reduction in entropy due to conditioning. In general, one variable's value is a hint about another's. For particular instance, we are generally less unsure about the input of a channel if we have observed its output.

相互の情報は条件付けにより不確実性を減少させる。確率変数の値は、おおよそ他の値にも関係している。特にこの例においては、出力行う際のチャンネルの入力に関して一般的にそれほど不確実ではないということがわかる。

If Y is unrelated to X, $H(X|Y) = H(X)$, $H(Y|X) = H(Y)$, and $I(X;Y) = 0$. If Y unambiguously indicates X, $H(X|Y) = 0$ and $I(X;Y) = H(X)$.

もし、Y が X に関係がなかったら、$H(X|Y) = H(X)$、$H(Y|X) = H(Y)$ かつ $I(X;Y) = 0$ となる。もし、 Y が明確に X を示していたら、$H(X|Y) = 0$ かつ $I(X;Y) = H(X)$ となる。

4.6.1 Entropy and Mutual Information; エントロピーと相互情報量

Mutual information is symmetric in X and Y (by eqn. (4.26a)), and non-negative (by eqn. (4.24) and eqn. (4.26d)).

相互情報量は X と Y について対称であり（式 eqn. (4.26a) より）、また負ではない（式 eqn. (4.24)，eqn. (4.26d) より）。

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) && (4.26e)\\ &= H(Y) - H(Y|X) && (4.26f)\\ &= H(X) + H(Y) - H(X,Y) && (4.26g)\\ &= I(Y;X) && (4.26h) \end{aligned}$$

If X is thought of as the input, and Y the output, then the idea of channel-conveyed information is best captured by $H(X) - H(X|Y)$, the reduction in uncertainty by the conditioning. However, it is often easier to calculate the equivalent $H(Y) - H(Y|X)$ [Mac02, p. 149].

$$\begin{aligned} I(X;X) &= H(X) - \cancel{H(X|X)} && (4.27a)\\ &= H(X) && (4.27b) \end{aligned}$$

Therefore entropy can be called "self-information."

このため時々エントロピーは "自己情報" とよばれることがあります。

To maximize information flow, you should "say what you mean" (avoid 'economies of truth') and "mean what you say" (don't fib).

情報の流れを最大にするためには、「あなたの意味していることを言う」ことと「あなたの言っていることを意味する」

4.6.2 Example: Two Coins; 例： 2つのコイン

An experiment is performed with two coins: 1 unbiased (fair), 1 with two heads (unfair). One of these coins is chosen randomly (X) and tossed twice, recording the number of heads (Y), as shown in Figure 4.9.

2つのコインを使って実験をします。1つのコインは普通に表と裏があるもの，もう1つは，両面とも表のコインとします。2つのコインから1つを任意に選び，コイン投げを2回行ない，そして表が出た回数を記録します。図 4.9 にこれを示します。

Clearly the number of heads gives a hint (information) about the chosen coin: $|\text{heads}| < 2 \Rightarrow$ fair coin chosen; $|\text{heads}| = 2 \Rightarrow$ probably double-headed coin chosen, but maybe the fair coin.

表が出た回数から，どのコインが選ばれたかという情報が得られます。| 表が出た回数 | <2 の時選んだコインは普通のコイン。| 表が出た回数 | =2 の時，両面とも表のコインという可能性もありますが，普通のコインであるかもしれません。

$$x = \begin{cases} 0 & \text{unbiased coin chosen} \quad \text{一般の裏表のあるコインを選択} \\ 1 & \text{two-headed coin chosen} \quad \text{両面とも表のコインを選択} \end{cases}$$

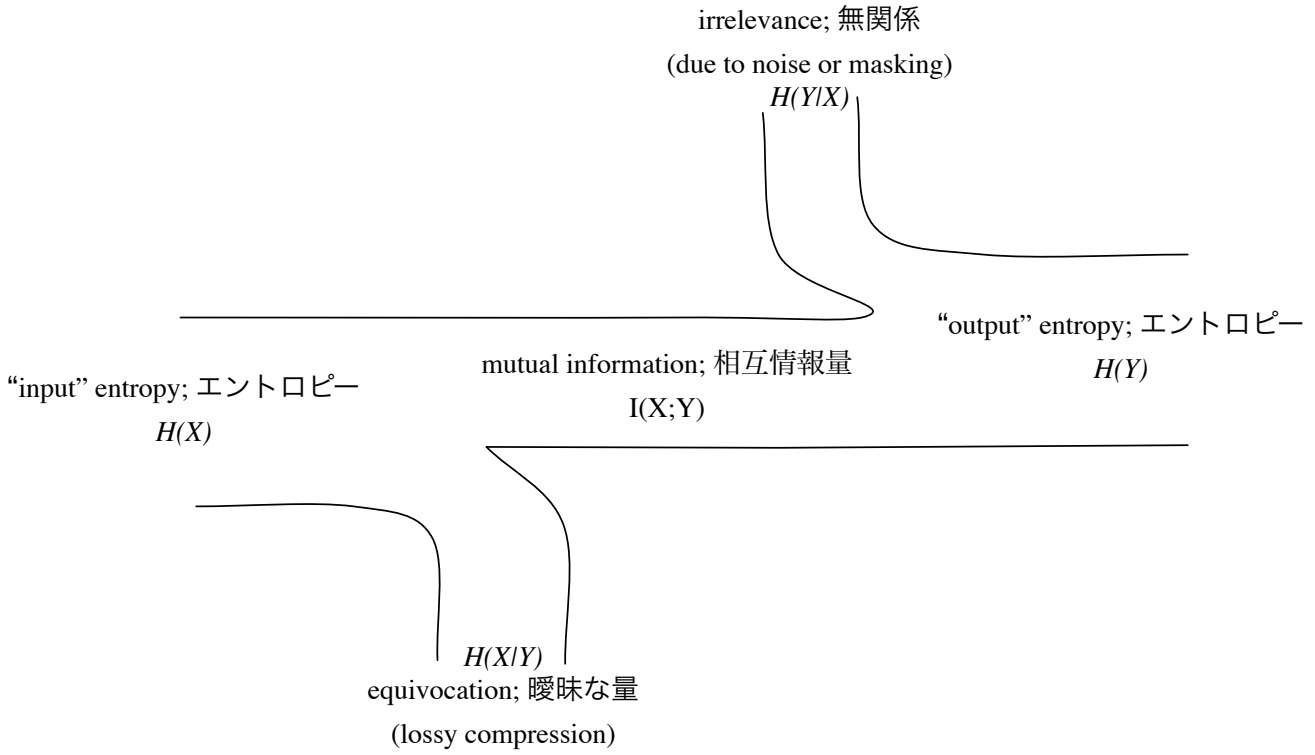


図 4.8: Information Flow; 情報の流れ: Irrelevance; 無関係 (ノイズや遮蔽物のため), Equivocation; 曖昧な量 (圧縮を失う)

$$y = \ |\text{heads}| = \ |\text{表}^{\text{おもて}}\text{が出た回数}^{\text{かいす}}|$$

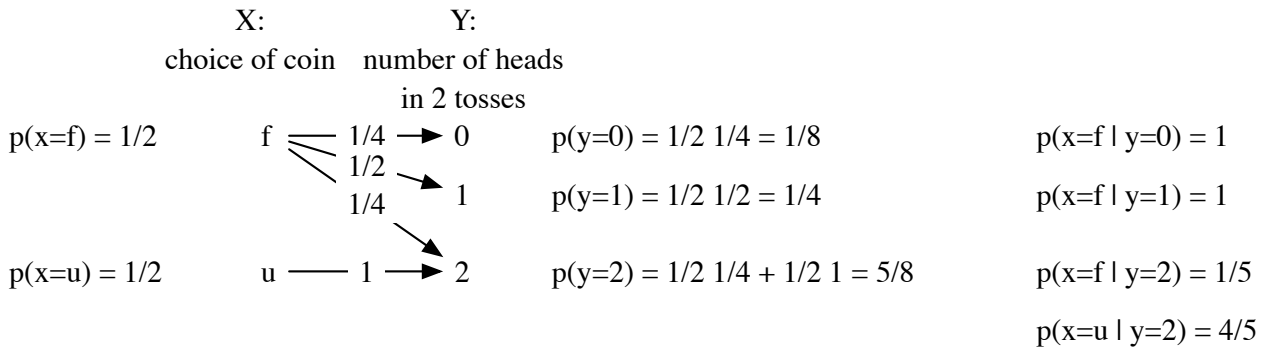
$$H(X) = H(\frac{1}{2}, \frac{1}{2}) = 1 \text{ bit}$$

$$\begin{aligned} H(X|Y) &= p(y=0)\ H(X|y=0) \\ &\quad + p(y=1)\ H(X|y=1) \\ &\quad + p(y=2)\ H(X|y=2) \\ &= \frac{1}{8}\cancel{H(1)} + \frac{1}{4}\cancel{H(1)} + \frac{5}{8}H(\frac{1}{5}, \frac{4}{5}) \\ &= \frac{1}{8}0 + \frac{1}{4}0 + -\frac{5}{8}(\frac{1}{5}\log\frac{1}{5} + \frac{4}{5}\log\frac{4}{5}) \\ &\approx 0.45 \text{ bits} \end{aligned}$$

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &\approx 0.55 \text{ bits} \end{aligned}$$

This is the maximum channel capacity, the amount of bidirectional information that each variable suggests about the other.

これはこの経路においての双方向情報の最大容量です。各変数が他方を意味しています。



(a) A coin-tossing experiment; コイン投げの実験— X: コインの選択 Y: 2 回のトスで表が何回でるか

$p(x,y)$		X		marginal distribution
		f	u	
Y	0	$\frac{1}{8}$	0	$p(y=0)=\frac{1}{8}$
	1	$\frac{1}{4}$	0	$p(y=1)=\frac{1}{4}$
	2	$\frac{1}{8}$	$\frac{1}{2}$	$p(y=2)=\frac{5}{8}$
marginal distribution		$p(x=f)=\frac{1}{2}$	$p(x=u)=\frac{1}{2}$	(1)

(b) Joint Probability Distribution

図 4.9: Example of Conditional Entropy

$$
\begin{aligned}
H(Y) &= H(\frac{1}{2}\cdot\frac{1}{4}, \frac{1}{2}\cdot\frac{1}{2}, \frac{1}{2}\cdot\frac{1}{4}+\frac{1}{2}\cdot 1) \\
&= H(\frac{1}{8}, \frac{1}{4}, \frac{1}{8}+\frac{1}{2}) \\
&= -\frac{1}{8}\log\frac{1}{8} - \frac{1}{4}\log\frac{1}{4} - \frac{5}{8}\log\frac{5}{8} \\
&\approx 1.3 \text{ bits}
\end{aligned}
$$

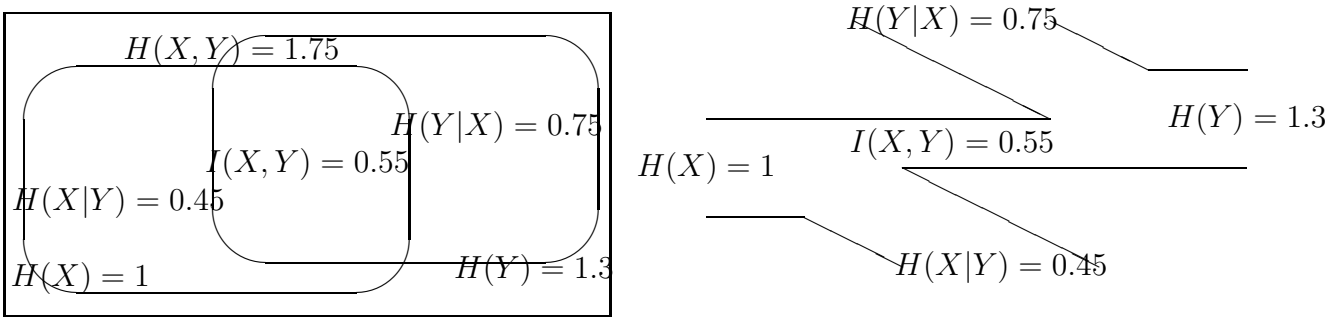
$$
\begin{aligned}
H(Y|X) &= p(X=f)\ H(Y|X=f) + p(X=u)\ H(Y|X=u) \\
&= \frac{1}{2}H(\frac{1}{4}, \frac{1}{2}, \frac{1}{4}) + \frac{1}{2}H(1) \\
&= \frac{1}{2}(-\frac{1}{4}\log\frac{1}{4} - \frac{1}{2}\log\frac{1}{2} - \frac{1}{4}\log\frac{1}{4}) + \cancel{\frac{1}{2}(-1\log 1)} \\
&= \frac{1}{2}[\frac{1}{4}(2) + \frac{1}{2}(1) + \frac{1}{4}(2)] \\
&= \frac{3}{4} \text{ bits}
\end{aligned}
$$

$$
\begin{aligned}
I(Y;X) &= H(Y) - H(Y|X) \\
&\approx 0.55 \text{ bits}
\end{aligned}
$$

confirming the value obtained previously. $\surd$ 先ので得られた値が確認されました。 $\surd$

The variation, or unpredictability, in the output (1.3 bits) is partly (0.55 bits) due to the input (choice of coin), but more due to randomness of the coin flips (0.75 bits).

出力中 (1.3 ビット) の変化、もしくは予測不能性は部分的に (0.55 ビット) 入力 (コインの選択) によるが、それ以上にコインのはじきかたの無作為性 (0.75 ビット) によるところが大きい。



(a) Information Containment

(b) Information Flow

図 4.10: Information in Coin-tossing Experimental Channel

第5章　5: Channel Coding; 通信路符号化の限界

5.1　Channels; 通信路

Model memoryless (stateless) channel as in Figure 5.1. The output depends only on the input at the time, independent of previous inputs and outputs.

メモリ（状態）のない通信路のモデルを、図5.1に示します。出力は、その瞬間の入力にのみ依存し、それ以前の入力、および出力には依存しません。

The channel is causal, so future events don't affect the present, but statelessness also prevents history from affecting the present.

通信路は原因なので、未来の出来事に影響を及ぼしませんが、ステートレスな状態もまた履歴が現在に影響を及ぼすのを妨げます。

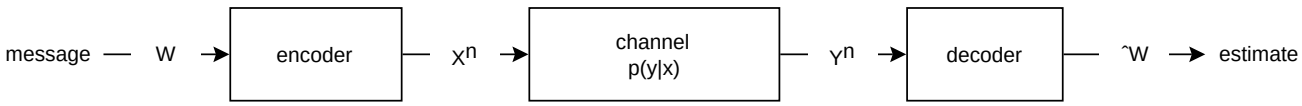


図 5.1: Channel Model: $x \in$ input alphabet $\mathcal{X}$, $y \in$ output alphabet $\mathcal{Y}$, $p(y|x)$ probability mass function; 通信路のモデル：$x \in$ 入力される確率変数 $\mathcal{X}$, $y \in$ 出力される確率変数 $\mathcal{Y}$, $p(y|x)$ 確率関数

Noise: snap, crackle, and pop (impulsive) and static and hiss (continuous) on audio line, "snow" in video signal.

ノイズ：音声回線での瞬間的なパチン、ピシッという音や絶え間ない「シー」というような雑音。ビデオ信号での砂嵐。

Channels communicate over space (transmission) and/or time (storage). Examples of media that can be modeled as channels include CDs, CD-ROMs, DVDs, BDs, phones, ethernet, paper, conversation, photographs, ... Noise in a channel is modeled by $p(y|x)$, the conditional probability of seeing output y given input x.

通信路は、空間（通信路）あるいは時間（記憶媒体）を越えて情報を伝えるものです。通信路のモデルとして、以下のような例が挙げられます。CD、CD-ROM、DVD、BD、電話、イーサネット、紙、会話、写真、...。通信路中の雑音は、$p(y|x)$ によってモデル化され、出力 y の条件付き確率は、入力 x によって得ることができます。

$$P = [p_{ij}] \tag{5.1a}$$

$$p_{ij} = p(y_j|x_i) \tag{5.1b}$$

$$p(y_j) = \sum_i p(x_i)p(y_j|x_i) \tag{5.1c}$$

These transition probabilities somewhat resemble Markov transition matrices, but express not a finite state automaton, but input/output correlation, so that it wouldn't make any sense to exponentiate (raise to a power) such matrices, since the model channel gets used only once/source symbol, and there is no feedback or recursion (cascading, serial composition). Since the ps are probabilities,

これらの推移確率はマルコフ推移行列と多少類似していますが、有限オートマトンではなく入出力の相関関係で表されます。つまり、モデルの通信路が一度だけソースシンボル使用し、フィードバックや再帰がない (直列、連続構成である) ので、当該の行列を累乗することに意味はありません。ps は確率なので

$$1 \geq p(y|x) \geq 0 \qquad \forall x, y. \tag{5.2a}$$

If something goes in (input is written), something comes out (output is read):

もし何かが書き込まれると、何かが読み出される。

$$\sum_y p(y|x) \;=\; 1 \qquad\qquad \forall x \tag{5.2b}$$

so the sum of each row of a transition matrix will be unity.

そのため、推移行列の各列の合計は一致します。

5.2 Channel Capacity; 通信路容量

The channel capacity C is the highest rate, in bits per channel use, at which information can be sent with arbitrarily low probability of error:

通信路容量 C とは、信号に対して通信路が与えられた時、どれだけ低いエラーの発生確率で信号を送れるかを表わし、その通信路において最も効率の良い状態を表わします:

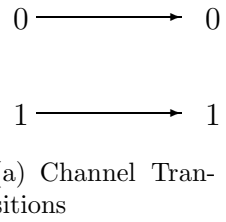
$$C \stackrel{\Delta}{=} \max_{\vec{p}} I(X;Y), \tag{5.3}$$

where $\vec{p}$ is the vector of input distributions, $p_i = p(x_i) \geq 0$ for all i, $\sum_{i=1}^{M} p_i = 1$, and the maximum is taken over all possible input distributions $\vec{p}$.

$\vec{p}$ は入力信号の出現確率ベクトルで、全ての i に対し、$p_i = p(x_i) \geq 0$、$\sum_{i=1}^{M} p_i = 1$ が成り立ち、最大値をとるのはすべての可能な入力信号の出現確率ベクトル $\vec{p}$ がこの 2 式を満たす時です。

There is a duality between problems of data compression and data transmission. During compression, redundancy is removed; during transmission, redundancy is added in controlled fashion to compensate for errors in channel.

データ圧縮の問題、データ伝送の問題という二元性があります。圧縮の過程で冗長度は除去されますが、伝送の際に制御の流れの中で通信路中のエラーを補うために冗長度が追加されます。



(a) Channel Transitions

$$P = [p(y|x)] = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

(b) Transition Matrix

図 5.2: Noiseless Binary Channel; ノイズのない二値通信路

5.3 Noiseless binary channel; ノイズのない二値通信路

Consider a noiseless (deterministic) two-state channel, as shown in Figure 5.2.

ノイズのない（確定的な）二値通信路について考えてみましょう、図5.2。

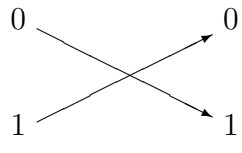
Recalling eqn. (5.3), $C \triangleq \max_{\vec{p}} I(X;Y)$, and the graph of entropy $H_2(p) \equiv H(p, 1-p)$, observe that one error-free bit transmitted per use of the channel implies a capacity of 1 bit, achieved when $\vec{p} = (p(x=0), p(x=1)) = (\frac{1}{2}, \frac{1}{2})$ (as shown in Figure 4.4).

もう一度、eqn. (5.3) $C \triangleq \max_{\vec{p}} I(X;Y)$ とエントロピーのグラフ $H_2(p) \equiv H(p, 1-p)$ を思い出して下さい。1ビットの容量をもつ通信路によって、ひとつのエラーのない信号が伝送されるのは、$\vec{p} = (p(x=0), p(x=1)) = (\frac{1}{2}, \frac{1}{2})$ の時であることが分かるでしょう (図4.4)。

$$\begin{aligned}
\vec{y} &= \vec{p}P \\
&= (\frac{1}{2}, \frac{1}{2}) \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \\
&= (\frac{1}{2}, \frac{1}{2}) \\
H(X) &= 1 \text{ bit} \\
\overset{\text{equivocation; 曖昧な量}}{H(X|Y)} &= 0 \text{ bits} \\
I(X;Y) &= H(X) - H(X|Y) \\
&= 1 - 0 \text{ bits} \\
&= 1 \text{ bit}
\end{aligned}$$

Note that an "always lying" channel, as seen in Figure 5.3,

では、「常に虚偽の」通信路ではどうなるのでしょう、図5.3



(a) Channel Transitions

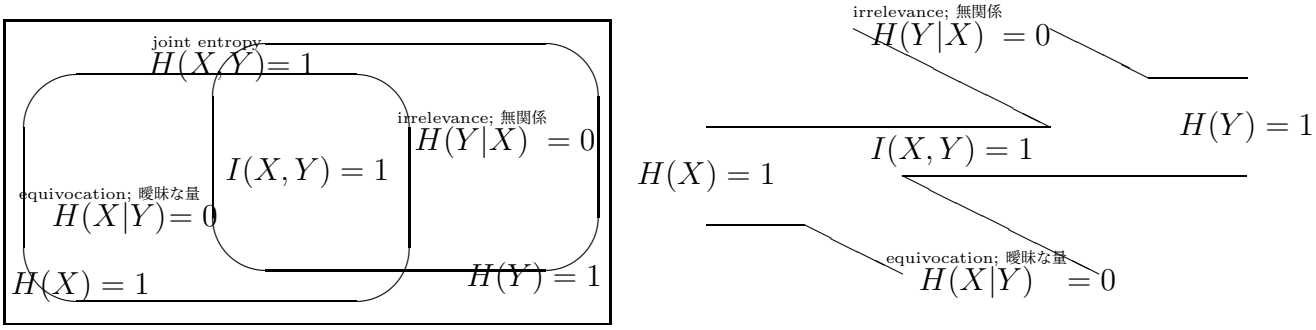
$$p(y|x) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

(b) Transition Matrix

図 5.3: Always Lying Channel; 「常に虚偽の」通信路

also has the same capacity, 1 bit per channel use, since the receiver, assumed to know the transition matrix, can simply invert the read data.

これもまた、同じ容量です。1ビットの容量を持つ通信路において、受信器が変遷行列を予期できれば、受けとった信号を常に反転させればいいだけです。

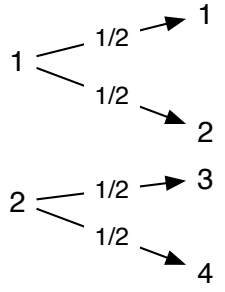


(a) Information Containment

(b) Information Flow

図 5.4: Information in Noiseless and "Always Lying" Binary Channels; ノイズのない「常に虚偽」の二値通信路の情報

5.4 Noisy channel with nonoverlapping outputs; 重複した出力がなくノイズのある通信路



(a) Channel Transitions

$$p(y|x) = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

(b) Transition Matrix

図 5.5: Noisy Channel with Nonoverlapping Outputs; 重複した出力がなくノイズのある通信路

If the input distribution is $\vec{x} = (1, 0)$, the joint distribution is $\begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$, and $H(X) = 0$, $H(Y) = 1$, $H(X,Y) = H(\frac{1}{2}, \frac{1}{2}) = 1$, $H(X|Y) = 0$, $H(Y|X) = 1$, and $I(X,Y) = 0$ bits.

The capacity of the channel in Figure 5.5 is also 1 bit/transmission, when $p(x = 0) = p(x = 1) = \frac{1}{2}$. Even though the outputs are nondeterministic, since they don't overlap the input can always be determined unambiguously.

If the input distribution is $\vec{x} = (\frac{1}{3}, \frac{2}{3})$, the joint distribution is $\begin{pmatrix} \frac{1}{6} & \frac{1}{6} & 0 & 0 \\ 0 & 0 & \frac{1}{3} & \frac{1}{3} \end{pmatrix}$, and $H(X) = H(\frac{1}{3}, \frac{2}{3}) \approx .9$, $H(Y) = H(\frac{1}{6}, \frac{1}{6}, \frac{1}{3}, \frac{1}{3}) \approx 1.9$, $H(X,Y) = H(\frac{1}{6}, \frac{1}{6}, \frac{1}{3}, \frac{1}{3}) \approx 1.9$, $H(X|Y) = 0$, $H(Y|X) = \frac{1}{3} \cdot 1 + \frac{2}{3} \cdot 1 = 1$, and $I(X,Y) \approx 0.9$ bits.

この通信路の場合も、$p(x = 0) = p(x = 1) = \frac{1}{2}$ のとき、容量は 1 ビットです。例え出力が確定されていないとしても、出力は重複しないので、入力はいつも明白に決まります。

						marginal distribution; 周辺分布
		1	2	3	4	
	1	$\frac{1}{4}$	$\frac{1}{4}$	0	0	$\frac{1}{2}$
	2	0	0	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{2}$
marginal distribution		$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	1

$$
\begin{aligned}
H(X) &= 1 \text{ bit} \\
\overset{\text{irrelevance; 無関係}}{H(Y|X)} &= \frac{1}{2}(1 \text{ bit}) + \frac{1}{2}(1 \text{ bit}) \\
&= 1 \text{ bit} \\
\overset{\text{joint entropy; 結合エントロピー}}{H(X,Y)} &= H(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}) \\
&= H(Y|X) + H(X) \\
&= 2 \text{ bits} \\
\overset{\text{equivocation; 曖昧な量}}{H(X|Y)} &= 0 \text{ bits} \\
H(Y) &= H(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}) \\
&= 2 \text{ bits} \\
I(X;Y) &= H(Y) - H(Y|X) \\
&= 2 - 1 \text{ bits} \\
&= 1 \text{ bit}
\end{aligned}
$$

Confirm via chain rule:

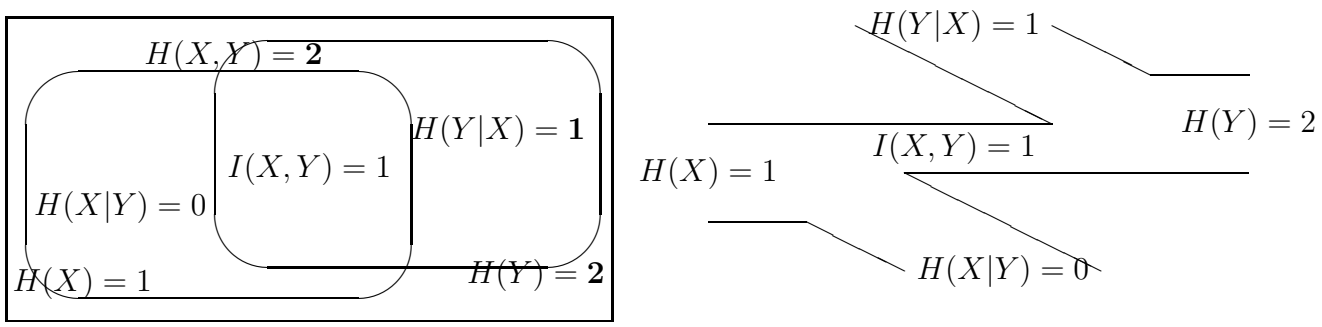
連鎖法則による確認:

$$
\begin{aligned}
H(X,Y) &= H(X) + H(Y|X) \\
&= H(Y) + H(X|Y) \\
H(Y) &= H(X,Y) - H(X|Y) \\
&= 2 \text{ bits } \surd \\
I(X;Y) &= H(Y) - H(Y|X) \\
&= H(X) - H(X|Y) \\
&= 1 - 0 \text{ bits} \\
&= 1 \text{ bit } \surd
\end{aligned}
$$

5.4.1 Noisy typewriter; ノイズのあるタイプライター

In the noisy typewriter (Figure 5.7(a)), the channel input is either unchanged, or transformed into the next letter.

ノイズのあるタイプライター（図5.7(a)）では、入力信号はそのまま出力されるか、次の文字に変換されて出力されるかのどちらかです。



(a) Information Containment

(b) Information Flow

図 5.6: Information in Noisy Channel with Nonoverlapping Outputs; 重複した出力がなくノイズのある通信路の情報

$$
\begin{aligned}
H(X,Y) &\neq H(X) + H(Y) \\
H(X,Y) &= H(X) + H(Y|X) \\
&= H(X) + H(Y) - I(X,Y) \\
&= 2\log 26 - \log 13 \\
&= 2(\log 2 + \log 13) - \log 13 \\
&= 2 + \log 13 \\
&\approx 6 \text{ bits}
\end{aligned}
$$

We can imagine disambiguating the channel by choosing a subset (alternate letters) of the inputs, so that the outputs do not overlap (Figure 5.7(b)).

そこで、入力群からサブセット（文字をひとつおきに扱う）を選ぶことによって通信路を明確にすることを考えます。これによって、出力の重複が無くなります（図5.7(b)）。

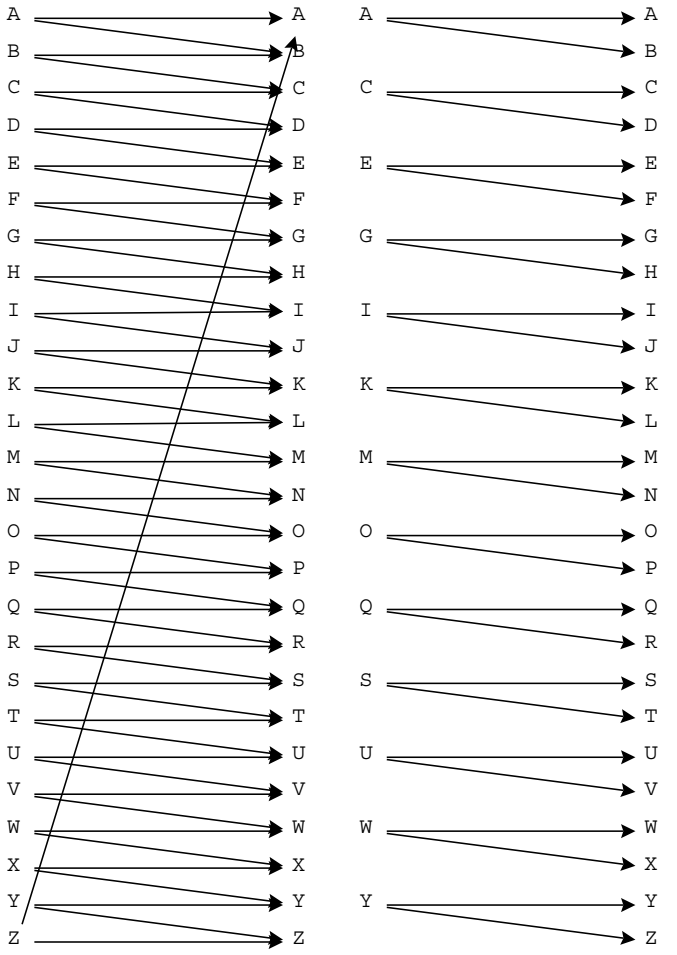
Since 13 symbols could be transmitted without error each transmission, the capacity of this channel can be calculated as $\log 13$ bits/transmission.

それぞれの伝送において、13 の文字がエラーなしで送れることから、この通信路の容量は $\log 13$ ビットと算出されます。

$$
\begin{aligned}
C &\stackrel{\Delta}{=} \max_{\vec{p}} I(X;Y) \\
&= \max[H(Y) - H(Y|X)] \\
&= \max H(Y) - 1 \\
&= \log 26 - 1 \\
&= \log 13
\end{aligned}
$$

This capacity can be achieved by choosing $\vec{p}$ uniformly distributed over all the odd inputs:

すべての奇数入力に対して、割り当てが一定になるように $\vec{p}$ を選べば、以下の式が成りたつことになります。



(a) Overlapping Outputs

(b) Non-overlapping Outputs

図 5.7: Noisy Typewriter; ノイズのあるタイプライター（出力が重複するものとしないもの）

$$p(\text{letter}) = \begin{cases} \frac{1}{13} & \text{letter} = \mathtt{A}, \mathtt{C}, \mathtt{E}, \ldots \\ 0 & \text{letter} = \mathtt{B}, \mathtt{D}, \mathtt{F}, \ldots \end{cases} \tag{5.4}$$

(Of course the information theoretic symbols such as H and I are completely different from typewriter letters H and I.)

（当然、H や I のような変数名は、タイプライターの文字 H、I とは全くの別なものです。）

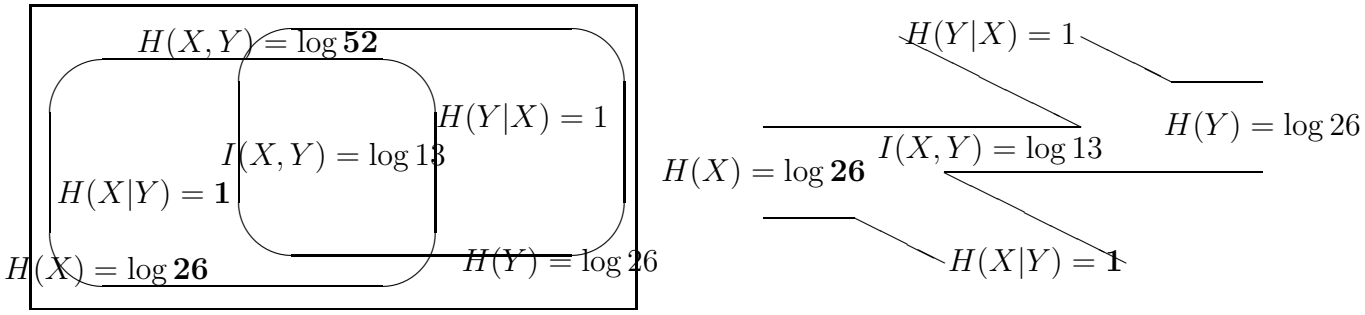
Note that

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &= \log 13 - 0 \\ &\approx 3.7 \text{ bits} \end{aligned}$$

5.5 Binary symmetric channel (BSC); 2元対称通信路

A BSC has the structure shown by Figure 5.10, where p is the probability of a "crossover" error.

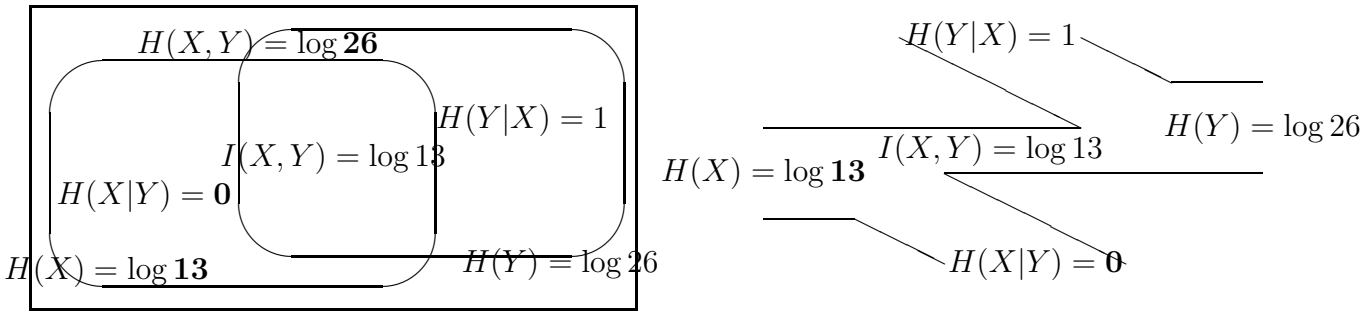
BSC (2要素対称通信路) は図5.10 のような構造をしています。p は、エラーの確率です。



(a) Information Containment

(b) Information Flow

図 5.8: Information in Noisy Typewriter with Overlapping Outputs; 重複した出力があるノイズのあるタイプライターの情報



(a) Information Containment

(b) Information Flow

図 5.9: Information in Noisy Typewriter with Non-Overlapping Outputs; 重複した出力がないノイズのあるタイプライターの情報: Equivocation eliminated by restricting input symbols; 入力記号の制限により曖昧な表現が除かれる

$$I(X;Y) = H(Y) - H(Y|X) \tag{5.5a}$$

definition of conditional entropy

$$= H(Y) - \sum_{x \in \mathcal{X}} p(x) H(Y|X = x) \tag{5.5b}$$

symmetric $\Rightarrow$ conditional probability independent of input

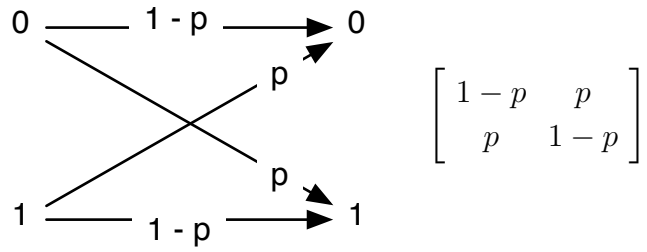
$$= H(Y) - \sum p(x) H(p, 1-p) \tag{5.5c}$$

notational convention

$$= H(Y) - \sum p(x) H_2(p) \tag{5.5d}$$

$\sum_x p(x) = 1$

$$= H(Y) - H_2(p) \tag{5.5e}$$



$$\begin{bmatrix} 1-p & p \\ p & 1-p \end{bmatrix}$$

図 5.10: **B**inary **S**ymmetric **C**hannel (BSC) with crossover probability p; 2 要素対称通信路

$$\max H(Y) = 1$$

$$\leq 1 - H_2(p) \tag{5.5f}$$

where the inequality at the end follows because Y is a binary random variable (with maximum entropy $H(Y) = 1$). Equality is achieved when the input distribution is uniform ($\vec{p} = (\frac{1}{2}, \frac{1}{2}) \Leftrightarrow p(x = 0) = p(x = 1) = \frac{1}{2}$).

Y は 2 進乱数 (エントロピーの最大値が $H(Y) = 1$) ですので、最後には不等式がつきます。等号成立は、入力分布が一定 ($\vec{p} = (\frac{1}{2}, \frac{1}{2}) \Leftrightarrow p(x = 0) = p(x = 1) = \frac{1}{2}$) な時です。

$$(1-\alpha, \alpha) \begin{bmatrix} 1-p & p \\ p & 1-p \end{bmatrix} = ((1-\alpha)(1-p) + \alpha p, (1-\alpha)p + \alpha(1-p)) \tag{5.6a}$$

$$1 - \alpha - p + \alpha p + \alpha p = \frac{1}{2} \tag{5.6b}$$

$$\alpha(1-2p) = 1 - \frac{1}{2} - p \tag{5.6c}$$

$$\alpha = \frac{\frac{1}{2} - p}{1-2p} = \frac{1}{2} \tag{5.6d}$$

Hence the information capacity of a BSC with parameter p is

ゆえに p の通信路容量は、

$$C = 1 - H_2(p) \text{ bits/transmisson} \tag{5.7}$$

If $H_2(p) = 0$ (i.e., at $\overbrace{p=0}^{\text{noiseless}}$ or $\overbrace{p=1}^{\text{“always lying”}}$), the capacity is 1 bit/transmission. If $H_2(p) = 1$ (at $p = 0.5$), the capacity is 0 bits/transmission, since the input can't be guessed from the output.

もし $H_2(p) = 0$ ならば ($\overbrace{p=0}^{\text{ノイズのない}}$ それとも $\overbrace{p=1}^{\text{「常に虚偽」}}$)、通信路容量は 1 度の送信につき 1bit になる。もし $H_2(p) = 1$ ならば ($p = 0.5$)、通信路容量は 1 度の送信につき 0bit になる、なぜならば入力は出力から推測することは出来ないからである。

5.6 Symmetric Channel and Maximum Liklihood Decoding; 対称通信路

Consider a channel with transmission matrix:

通信路行列を用いた通信路について考えると、

$$P = p(y|x) = \begin{bmatrix} 0.3 & 0.2 & 0.5 \\ 0.5 & 0.3 & 0.2 \\ 0.2 & 0.5 & 0.3 \end{bmatrix} \tag{5.8}$$

where an entry in the x^{th} row and y^{th} column denotes the conditional probability that y is received when x is sent. In this channel, all the rows of the probability transmission matrix are permutations of each other, and so are the columns. Such a channel is said to be *symmetric*.

x^{th} 行、y^{th} 列の要素は、x が転送された時に、y が受信される条件つき確率を表しています。この通信路では、確率通信路行列のすべての行、列が相互の順列です。そのような通信路を**対称的**といいます。

If $\vec{r}$ is a row of the matrix,

もし $\vec{r}$ が行列の行なら、

$$\begin{aligned} I(X;Y) &= H(Y) - H(Y|X) && (5.9a)\\ &= H(Y) - \sum_{x\in\mathcal{X}} p(x)H(Y|X=x) && (5.9b)\\ &= H(Y) - \sum_{x} p(x)H(\vec{r}) && (5.9c)\\ &= H(Y) - H(\vec{r}) && (5.9d)\\ &\leq \log|\mathcal{Y}| - H(\vec{r}) && (5.9e) \end{aligned}$$

with equality if the output distribution (across output alphabet $\mathcal{Y}$) is uniform. Choosing a uniform input distribution $p(x) = \frac{1}{|\mathcal{X}|}$ yields

出力分布 (出力されるアルファベット $\mathcal{Y}$) が一定なら等号が成立します。一定の入力分布 $p(x) = \frac{1}{|\mathcal{X}|}$ が与えられるように選べば、

$$\begin{aligned} p(y) &= \sum_{x\in\mathcal{X}} p(y|x)p(x) && (5.10a)\\ &= \frac{1}{|\mathcal{X}|}\sum p(y|x) && (5.10b)\\ &= k\frac{1}{|\mathcal{X}|} && (5.10c)\\ &= \frac{1}{|\mathcal{Y}|} && (5.10d) \end{aligned}$$

where k is the sum of the entries in any column of the probability transition matrix. Then, for the example above,

k は確率遷移行列の 1 つの列の要素の和です。それから、前述の例の中で、

$$\begin{aligned} C &\triangleq \max_{\vec{p}} I(X;Y)\\ &= \log 3 - H(0.2, 0.3, 0.5)\\ &\approx 1.6 - 1.5\\ &= 0.1 \text{ bits/transmission} \end{aligned}$$

In general, for a symmetric channel conditional probability transition matrix,

一般的に、対称通信路の条件付き確率遷移行列は

$$C = \log|\mathcal{Y}| - H(\text{any row of the transition matrix}) \tag{5.11}$$

5.6.1 Shannon-Hartley Capacity Theorem

The system capacity C of a channel perturbed by additive white Gaussian noise (AWGN) is a function of the average received signal power S, the average noise power N (so S/N is the mean-square signal-to-noise ratio), and the (one-sided) bandwidth B:

システム容量 C は受け取る信号電力の平均 S とノイズ電力の平均 N とバンド幅 B の関数で, その関係は次のように表現する.

$$C = B \log_2(1 + \frac{S}{N}) = B \log_2(\frac{S+N}{N}). \tag{5.12}$$

This equation represents a kind of analog↔digital conversion, since B has units of Hz (cycles/s) while C has units of bps (bit/s).

C が bps 単位を持つ一方、B は Hz 単位なので、この方程式は一種のアナログ ↔ デジタル変換を意味する。

For example, an ordinary analog phone line has a bandwidth of about 3,400 Hz, and a ratio of signal power to noise power of about 1000 (equivalent to 30 dB). So its capacity would be

たとえば, バンド幅が 3,400 Hz で, 信号とノイズの比が 1000 (30 dB 相当) である電話線があったとすると, その容量は

$$C \approx 3400 \log_2(1 + 1000) = 3400 \times 9.97 \approx 34,000 \text{ bps} = 34 \text{ kbps}.$$

Decibels corresponding to any ratio r of energy or power are calculated as $10 \log_{10} r$.

エネルギーまたは電力比率 r とでも一致するデシベルは $10 \log_{10} r$ として計算される。

$$\text{SNR/dB} \triangleq 10 \log_{10} \frac{S}{N} \tag{5.13a}$$

$$\frac{S}{N} = 10^{\frac{\text{SNR/dB}}{10}} \tag{5.13b}$$

For another example, if it is required to transmit at 50 kbps, using a bandwidth (on, for example, radio spectrum) of 1 MHz, then the minimum signal-to-noise power ratio required is given by

もう一つ例として, もし 1 MHz のバンド幅を使っていて, 50 kbps の伝送を必要とするなら, 要求される最小の信号とノイズの比は次のように得られる。

$$\begin{aligned} \frac{S}{N} &= 2^{\frac{C}{B}} - 1 \\ &= 2^{\frac{50,000}{1,000,000}} - 1 \\ &\approx 0.035 \\ &\approx -14.5 \text{ dB} \end{aligned}$$

This shows that it is possible to transmit using signals which are actually much weaker than the background noise level, as in spread-spectrum communications, such as CDMA used in mobile telephony.

これはスペクトラム拡散通信のようにバックグラウンドノイズレベルより弱い信号を使うと伝送が可能ということです (たとえば携帯電話の CDMA)。

Note that if there were no noise in a channel, then its capacity would be infinite even with low bandwidth. For instance, a real (infinite-precision floating point) number could represent an arbitrary sequence of code words.

もし, ノイズが全くなければ容量は無限であることに注意してください。例えば実数 (任意の精度の浮動小数点) は任意のコード配列を表現できます。

For large or small signal-to-noise ratios, eqn. (5.12) can be approximated.

大きいもしくは小さい信号対雑音比において eqn. (5.12) は近似できる。

5.6.1.1 Large S/N

For signal-to-noise ratios ("S/N" for power ratios or "SNR" for deciBels) $\gg 1$,

信号対雑音比 (「S/N」は出力比、「SNR」はデシベル)$\gg 1$ のとき,

$$\begin{aligned} C &\approx B \log_2 \frac{S}{N} && (5.14a) \\ &= B \log_2 10^{\frac{\text{SNR/dB}}{10}} && (5.14b) \\ &= B \ \text{SNR/dB} \log_2 10^{\frac{1}{10}} && (5.14c) \\ &\approx 0.332 \ B \ \text{SNR/dB} && (5.14d) \end{aligned}$$

To repeat the above telephone example,

上記の電話例を繰り返すと、

$$C \approx (0.332\,\text{bit/s/Hz})(3.4\,\text{kHz})(30) = 34\ \text{kbps} \ \surd$$

5.6.1.2 Small S/N

For signal-to-noise ratios $\ll 1$, if the (additive white Gaussian) one-sided noise power spectral density is $P(f) = \eta$ watts (or volts^2, normalized to 1 $\overset{\text{Ohm}}{\Omega}$) per Hz, assuming that noise power is proportional to bandwidth,

信号対雑音比 $\ll 1$ の時、一方向の (付加白色ガウス) 雑音電力のスペクトル密度が $P(f) = \eta$ ワット (もしくは 1Ω に正規化されたボルト^2) / Hz であった場合、雑音電力がバンド幅に比例すると仮定すると、

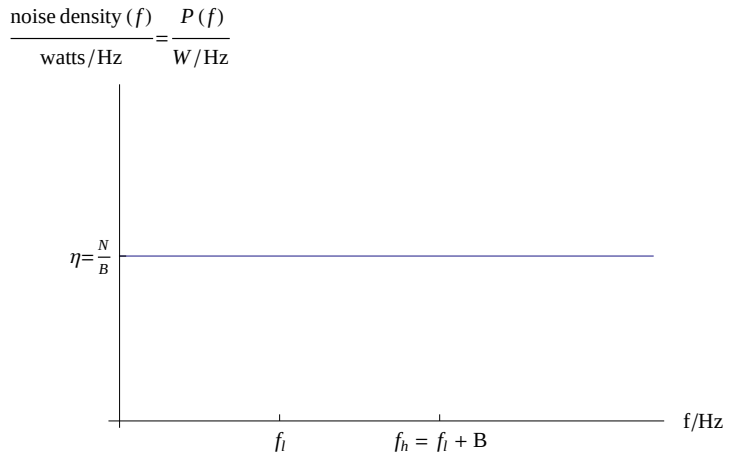


図 5.11: AWGN: power spectrum density of **A**dditive **W**hite **G**aussian **N**oise

$$
\begin{aligned}
N &= \int_{f_s}^{f_e=f_s+B} \overbrace{\text{noise density(f)}}^{\text{[W/Hz]}}\, df && \text{(5.15a)}\\
&= \eta B && \text{(5.15b)}\\
C &= B\log_2(1+\frac{S}{\eta B}) && \text{(5.15c)}\\
&= \frac{S}{\eta}\frac{\eta B}{S}\log_2(1+\frac{S}{\eta B}) && \text{(5.15d)}\\
&= \frac{S}{\eta}\log_2(1+\frac{S}{\eta B})^{\frac{\eta B}{S}} && \text{(5.15e)}\\
\lim_{x\to 0}(1+x)^{\frac{1}{x}} &= e && \text{(5.15f)}\\
\lim_{\frac{S}{N}\to 0} C = \lim_{\frac{S}{\eta B}\to 0} \frac{S}{\eta}\log_2(1+\frac{S}{\eta B})^{\frac{\eta B}{S}} &= \frac{S}{\eta}\log_2 e && \text{(5.15g)}\\
&\approx 1.44\frac{S}{\eta} && \text{(5.15h)}\\
&= 1.44\ B\ \frac{S}{N} && \text{(5.15i)}
\end{aligned}
$$

The $\log_2 e$ (≈ 1.44) is the "exchange rate" from nats to bits, calculable by the chain rule for logarithms:

$log_2 e$ (≈ 1.44) は nats から bits への交換比率で、対数の連鎖法則によって計算できる。

$$(\log_e x\ [\text{nats}])(\log_2 e) = \log_2 x\ [\text{bits}]. \tag{5.16}$$

The channel capacity C is the maximum number of bit/s. Since the average bit duration is $1/C$ seconds, the average signal power per bit [in W/bit] is $E_b = S/C$, and bandwidth efficiency [in bit/s/Hz] is

通信路容量 C は bit/s の最大値である。平均ビット期間が $1/C$ 秒であるので、1 ビットにおける平均の信号の強さは、$E_b = S/C$、帯域幅効率は以下のようになる。

$$\frac{C}{B} = \log_2(1 + \frac{\overbrace{E_b C}^{S}}{\underbrace{\eta B}_{N}}). \tag{5.17}$$

$\frac{E_b}{\eta}$ is normalized average power/bit, where the normalization is with respect to one-sided power spectral density of the noise, a sort of signal-to-noise ratio:

$\frac{E_b}{\eta}$ は正規化された平均の強さ/ビットで、正規化はノイズの一方的な力の分光濃度（一種の信号対ノイズの比率）に関係しており、

$$\frac{E_b}{\eta} = \frac{B}{C}(2^{\frac{C}{B}} - 1) \tag{5.18}$$

$$= \frac{2^{C/B} - 1}{C/B} \tag{5.19}$$

Note that

$$\frac{de^x}{dx} = e^x \tag{5.20}$$

$$\frac{de^{f(x)}}{dx} = e^{f(x)}\frac{df(x)}{dx} \tag{5.21}$$

$$\frac{d2^x}{dx} = \frac{d(e^{\ln 2})^x}{dx} = \frac{d(e^{(\ln 2)^x})}{dx} \tag{5.22a}$$

$$= \frac{d(e^{(\ln 2)x})}{dx} \tag{5.22b}$$

$$= e^{(\ln 2)x}\ln 2 \tag{5.22c}$$

$$= 2^x \ln 2 \tag{5.22d}$$

The limit of the ratio $\frac{E_b}{\eta}$ as bandwidth efficiency $\frac{C}{B} \to 0$ exists and can be found, for instance, by l'Hôpital's rule (as in §4.4.3) (with the above reminder):

帯域幅効率が、$\frac{C}{B} \to 0$ であるとき $\frac{E_b}{\eta}$ の極限は存在し、ロピタルの定理（§4.4.3）などを用いて求めることができる。

$$\lim_{\frac{C}{B}\to 0} \frac{E_b}{\eta} = \lim_{\frac{C}{B}\to 0} \frac{2^{C/B} - 1}{C/B} \tag{5.23a}$$

$$= \lim_{\frac{C}{B}\to 0} \frac{\frac{d}{d(C/B)}(2^{C/B} - 1)}{\frac{d}{d(C/B)}(C/B)} \tag{5.23b}$$

$$= \lim_{\frac{C}{B}\to 0} \frac{2^{C/B}\ln 2}{1} \tag{5.23c}$$

$$= \ln 2 \approx 0.69 \tag{5.23d}$$

Since this is a ratio of energies, it may meaningfully be expressed in decibels, $10\log 0.69 \approx -1.6$ dB. This is the so-called "Shannon-Hartley limit."

これはエネルギーの割合であるため、意味があるデシベルで示すことができる。$10\log 0.69 \approx -1.6$ dB。これは「シャノン・ハートレー限界」と呼ばれる。

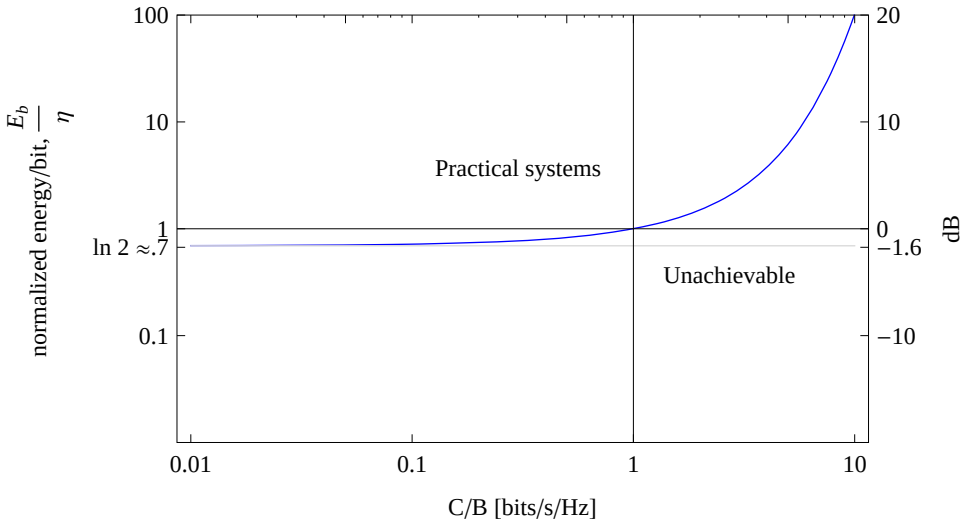


図 5.12: Shannon-Hartley Limit: normalized energy/bit vs. bandwidth efficiency (eqn. (5.19)); シャノン・ハートレー制限：正規化エネルギー/ビット対帯域幅効率 (eqn. (5.19))

This asymptotic convergence is graphically displayed if the normalized power per bit $\frac{E_b}{\eta}$ is plotted against bandwidth efficiency $\frac{C}{B}$, as in Figure 5.12. If the normalized bit energy is less than -1.6 dB, the Shannon-Hartley law can never be satisfied, however inefficient (in terms of bandwidth efficiency, bit rate/Hz) we are prepared to be. Above the curve (to the left) gives values of $\frac{C}{B}$ which satisy the law for a given bandwith efficiency: any bit error rate, however low, can be achieved in theory. Below the curve (to the right), the energy/bit is too low (in relation to η) for any given bandwidth efficiency: certain bit-error rates become unachievable.

図 5.12 の場合のように、ビット $\frac{E_b}{\eta}$ につき正規化された力が帯域幅効率 $\frac{C}{B}$ に対してのグラフを描けば、この漸近線の収束は視覚的に示される。正規化 されたビットエネルギーが-1.6dB 以下ならば、シャノン・ ハートレーの定理が満たされることはない。(帯域幅効率に関して) どんなに能率が悪くても、我々は準備する。図 5.10 の曲線の上方より、所定の帯域幅効率において定理を満たす,C-B. の値がわかる。：ビットがどんな率でも、どんなに低くても、理論的には実現可能である。図 5.12 の曲線の下方においてはエネルギー/ビットはどんな帯域幅効率に対しても低すぎる。特定のビットの率においては実現不可能である。

図 5.13: Quantized Gray Scale; 量子化グレースケール

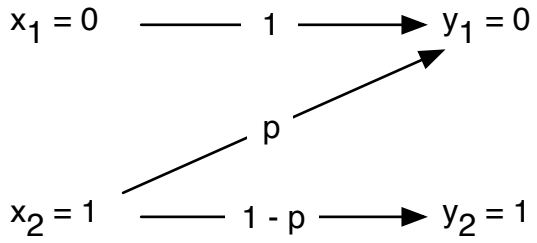
A final example: a black and white television can be considered as having a display of about 3×10^5 pixels. Assuming that each pixel's intensity is equiprobable among ten distinguishable brightness levels (as in Figure 5.13), that the refresh rate is 30 fps (frames/s), and that 30 dB SNR is required for satisfactory picture reproduction, what is the minimum channel bandwidth?

最終例：白黒のテレビはおよそ 3×10^5 ピクセルの表示をすることができる。各々のピクセルの強さが 10 の区別できる明るさレベルのそれぞれが同程度の確率で、リフレッシュレートが 30fps であり、その 30dB の SNR が満足な画質での再生のために必要である時、最小限の通信路帯域幅を求めよ。

$$\text{SNR} = 30\,\text{dB} \Rightarrow \text{S/N} = 10^{\frac{30}{10}} = 1000$$

$$\begin{aligned} C &= B\log_2(1+\text{S/N}) \\ (30\,\text{fps})(3\times10^5\,\text{pixels/frame})(\log_2 10\,\text{bits/pixel}) &= B\log_2(1+1000) \\ B &\approx 3\times10^6\,\text{Hz} \end{aligned}$$

5.6.2 Z-channel



$$(1-\alpha,\alpha)\begin{bmatrix} 1 & 0 \\ p & 1-p \end{bmatrix} = (1-\alpha+\alpha p, \alpha(1-p))$$

図 5.14: Z-Channel

$$p(X=x_2) = p(x_2) = \alpha \tag{5.24a}$$
$$p(X=x_1) = p(x_1) = 1-\alpha \tag{5.24b}$$
$$H(Y) = H(1-\alpha+\alpha p, \alpha(1-p)) \tag{5.24c}$$
$$= -(1-\alpha+\alpha p)\log(1-\alpha+\alpha p) - \alpha(1-p)\log\alpha(1-p) \tag{5.24d}$$
$$H(Y|X) = p(x_1)H(Y|X=x_1) + p(x_2)H(Y|X=x_2) \tag{5.24e}$$
$$= \cancel{-(1-\alpha)\cdot 1\log 1} - \cancel{(1-\alpha)\cdot 0\log 0} - \alpha p\log p - \alpha(1-p)\log(1-p) \tag{5.24f}$$
$$I(X;Y) = H(Y) - H(Y|X) \tag{5.24g}$$
$$= -(1-\alpha+\alpha p)\log(1-\alpha+\alpha p) - \alpha(1-p)\log\alpha(1-p) + \alpha p\log p + \alpha(1-p)\log(1-p) \tag{5.24h}$$
$$= -(1-\alpha+\alpha p)\log(1-\alpha+\alpha p) - \alpha(1-p)\log\alpha + \alpha p\log p \tag{5.24i}$$
$$C = \max_{\alpha} I(X;Y) \tag{5.24j}$$

By the chain rule for derivatives,

$$\frac{d}{dx}\ln f(x) = \frac{1}{f(x)}\frac{d}{dx}f(x) \tag{5.25a}$$

By the chain rule for logarithms,

$$\frac{d}{dx}\lg f(x) = \frac{d}{dx}\log_2 f(x) = \frac{d}{dx}\ln f(x)\log_2 e \tag{5.25b}$$

$$= (\lg e)\frac{1}{f(x)}\frac{d}{dx}f(x) \tag{5.25c}$$

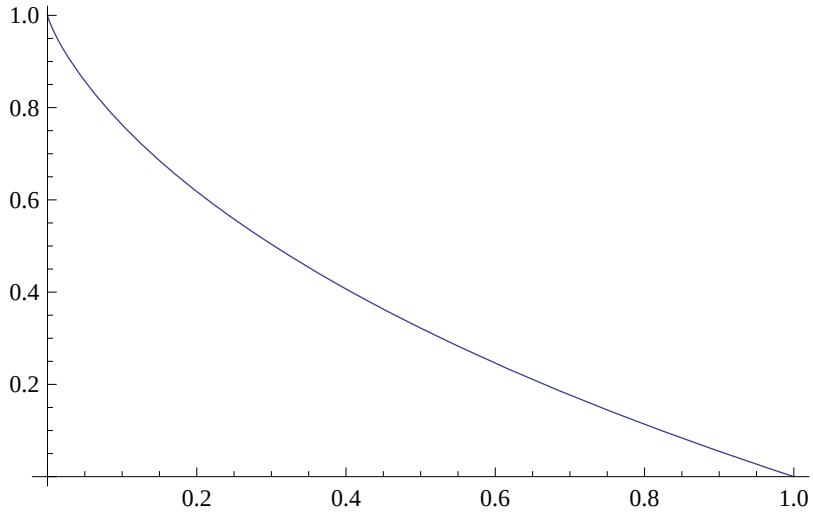


図 5.15: Z-Channel Capacity; throughput vs. error probability. Even when $p = \frac{1}{2}$, some channel capacity can be salvaged. However, when $p = 1$, output is constant ($y_1 = 0$), so capacity vanishes completely. This plot shows the capacity, which does what one would expect: full capacity when error probability is zero, dropping to zero capacity when error probablility is unity.

The maxima or minima of a function occur where its derivative vanishes.

$$\frac{dI(X;Y)}{d\alpha} = (1-p)\log(1-\alpha+\alpha p) - \cancel{(1-\alpha+\alpha p)}\frac{p-1}{\cancel{1-\alpha+\alpha p}}\log_2 e \tag{5.26a}$$

$$- (1-p)\log\alpha - \cancel{\alpha}(1-p)\frac{1}{\cancel{\alpha}}\log_2 e + p\log p$$

$$= (1-p)\log(1-\alpha+\alpha p) - (1-p)\log\alpha + p\log p \tag{5.26b}$$

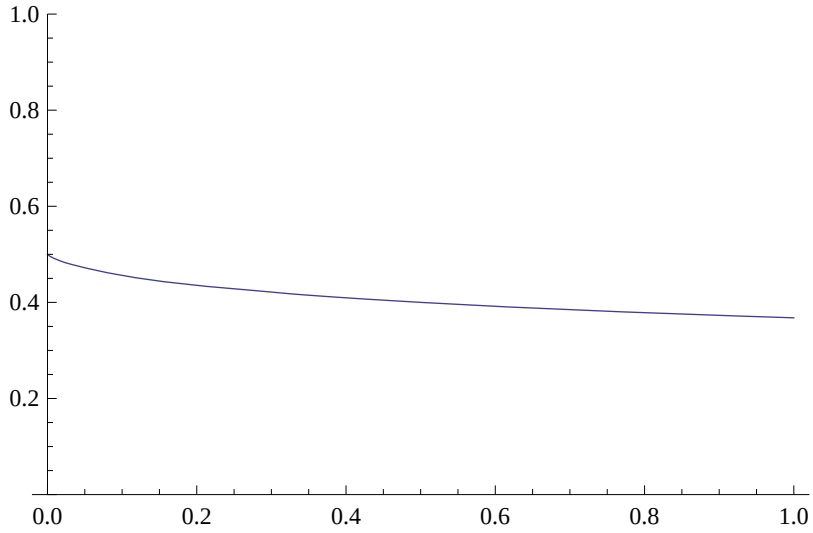


図 5.16: Z-Channel Optimization: optimal fraction of use of noisy side vs. error probability. This plot shows the optimal α to use for given p, the chance of cross-symbol error in a z-channel. Note that for zero error probability, highest throughput obtained when 50% use of each side, as for usual simple noiseless channel, but as error probability increases, use of the error-prone side is decreased for optimal throughput.

Let the point α at which the mutual information function maximizes be α_0.

$$\left.\frac{dI(X;Y)}{d\alpha}\right|_{\alpha=\alpha_0} = 0 \tag{5.27a}$$

$$\log(1-\alpha_0+\alpha_0 p) - log\alpha_0 + \frac{p\log p}{1-p} = 0 \tag{5.27b}$$

$$\log\frac{1-\alpha_0+\alpha_0 p}{\alpha_0} = -\frac{p}{1-p}\log p \tag{5.27c}$$

$$1-\alpha_0+\alpha_0 p = \alpha_0 p^{p/(p-1)} \tag{5.27d}$$

$$\alpha_0 = \frac{1}{1-p+p^{p/(p-1)}} \tag{5.27e}$$

$$C = -\log\alpha_0 - \frac{p}{p-1}\log p \tag{5.28a}$$

$$= \log(1-p+p^{p/(p-1)}) - \frac{p}{p-1}\log p \tag{5.28b}$$

$$= \log\left(\frac{1-p+p^{\frac{p}{p-1}}}{p^{\frac{p}{p-1}}}\right) \tag{5.28c}$$

$$= \log((1-p)p^{\frac{p}{1-p}}+1) \tag{5.28d}$$

5.7 (Average Error Rate)

5.8 (Channel Coding)

5.9 (Hamming Distance)

5.10 (Channel Coding Theorem)

5.11 (Lemma for Proof of Channel Coding Theorem)

第6章　6: Limits of Coding Theory; 符号理論

6.1　Parity Check; パリティチェック

6.1.1　Binary functions; 2進関数

Table 6.17 shows all 16 (2^{2^2}) dyadic (two argument) binary functions, organized as pairs of complementary functions. In binary arithmetic, multiplication corresponds to AND ('&' in C and Java); addition ('+') corresponds to (I)OR ('|' in C and Java); and addition mod 2 ('⊕') corresponds to XOR ('^' in C and Java).

表6.17 は、2変数の2進関数の組の16(2^{2^2}) 個すべてを表しています。2進演算では、かけ算が AND (C と Java の「&」)、足し算 (+) が OR (C と Java の「|」)、和 mod2, (⊕) が XOR (C と Java の「^」) に対応します。

Recalling §1.1.3.5, DeMorgan's theorem (for NAND and NOR):

ド・モルガンの定理 (NAND と NOR), §1.1.3.5:

$$\text{NAND: } \overline{A\,B} \equiv \overline{A} + \overline{B} \tag{6.1a}$$

$$\text{NOR: } \overline{A+B} = \overline{A}\,\overline{B} \tag{6.1b}$$

Example (for NOR): We might not want to go outside if it's raining or if it's cold. ⇔ We want to go out only if it's not raining and it's not cold.

例 (for NOR):雨が降っていたり寒いひは外出したくない。⇔ 雨が降っておらず、寒くない日に外出したい。

not(raining or cold) ≡ not raining and not cold

Any boolean function can be implemented by just a combination of NOR or NAND gates.

どのブーリアン関数も NAND それとも NOR の組合わせで実装できる。

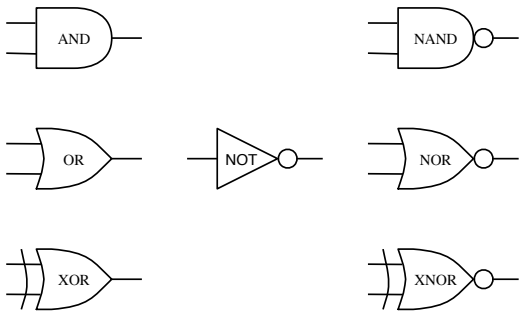


図 6.1: Symbols for boolean gates; ブーリアンゲートの記号

表 6.1: ZERO, RESET, INHIBIT, OFF (0)

X\Y	0	1
0	0	0
1	0	0

表 6.2: ONE, SET, ASSERT, ON (1)

X\Y	0	1
0	1	1
1	1	1

表 6.3: AND, conjunction (XY)

X\Y	0	1
0	0	0
1	0	1

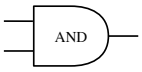


表 6.4: NAND ($\overline{XY} \equiv \overline{X} + \overline{Y}$)

X\Y	0	1
0	1	1
1	1	0

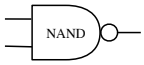


表 6.5: $X\overline{Y}$ (set difference $X - Y$)

X\Y	0	1
0	0	0
1	1	0

表 6.6: IMPLIES (sufficiency): $X \Rightarrow Y$, $Y + \overline{X}$

X\Y	0	1
0	1	1
1	0	1

表 6.7: X

X\Y	0	1
0	0	0
1	1	1

表 6.8: $\overline{X}$

X\Y	0	1
0	1	1
1	0	0

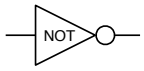


表 6.9: $\overline{X}Y$ (set difference $Y - X$)

X\Y	0	1
0	0	1
1	0	0

表 6.10: $X + \overline{Y}$, $Y \Rightarrow X$ (necessity)

X\Y	0	1
0	1	0
1	1	1

表 6.11: Y

X\Y	0	1
0	0	1
1	0	1

表 6.12: $\overline{Y}$

X\Y	0	1
0	1	0
1	1	0

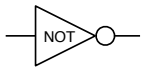


表 6.13: $(X \oplus Y, X \neq Y)$: XOR—e**x**clusive or (inequality, symmetric difference), same as CNOT (controlled not, used in quantum computing)

X\Y	0	1
0	0	1
1	1	0

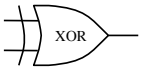


表 6.14: $X \equiv Y$, $X \Leftrightarrow Y$: EQUIV, XNOR (equality, coincidence, mutual implication, iff [if and only if], necessity and sufficiency)

X\Y	0	1
0	1	0
1	0	1

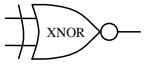


表 6.15: (**i**nclusive) OR, disjunction ($X + Y$)

X\Y	0	1
0	0	1
1	1	1

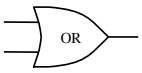


表 6.16: NOR ($\overline{X + Y} \equiv \overline{X}\ \overline{Y}$)

X\Y	0	1
0	1	0
1	0	0

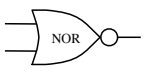


表 6.17: Dyadic binary functions; 2 変数の 2 進関数の組

A	B	AND: A&B	XOR: A^B	IOR: A|B
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	0	1

表 6.18: Bitwise Operations (and C and Java notation); ビット演算 (と C や Java での表記)

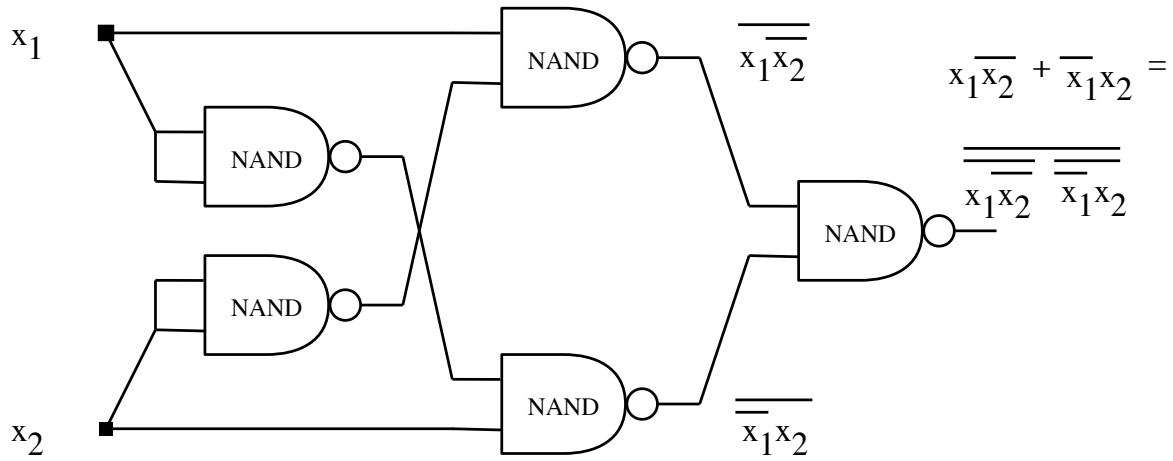


図 6.2: XOR can be implemented with just NAND gates. This circuit is also deployable as a half-adder, as it calculates the LSB of 1-bit addition. (The MSB is just an AND, or NANDed NAND.); XOR は NAND だけの組合わせで実装できる。これは 1 ビット加算の LSB を計算するのでこの回路は半加算器としても配置可能である。(MSB は AND や NAND された NAND である。)

6.1.2 Magic cards; マジックカード

The magic cards in Figure 6.3 work according to simple binary arithmetic. C(opy and c)ut out the six cards and hand them to someone. Have that friend silently choose an arbitrary number on any card, and then return to you all the cards which have that number on them. To discover your friend's secret number, add the first numbers on each of the cards returned to you.

図 6.3 のマジックカードは、単純な 2 進演算によっておこないます。6 枚のカードを切りとって友達に渡します。その友達は黙って適当にカードの中にある番号を選びます。そして、あなたはその番号が含まれているすべてのカードをもらいます。友達の選んだ番号を発見するには、単純にあなたがもらったカードの先頭 (左上) の番号を足し合わせればよいのです。

1	3	5	7	9	11	13	15
17	19	21	23	25	27	29	31
33	35	37	39	41	43	45	47
49	51	53	55	57	59	61	63

2	3	6	7	10	11	14	15
18	19	22	23	26	27	30	31
34	35	38	39	42	43	46	47
50	51	54	55	58	59	62	63

4	5	6	7	12	13	14	15
20	21	22	23	28	29	30	31
36	37	38	39	44	45	46	47
52	53	54	55	60	61	62	63

8	9	10	11	12	13	14	15
24	25	26	27	28	29	30	31
40	41	42	43	44	45	46	47
56	57	58	59	60	61	62	63

16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63

32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63

図 6.3: Magic Cards; マジックカード

6.1.3 Error detection; エラーの検知

In a perfect, noise-free world, checks would not be needed. But reality being imperfect, storage and communication systems need a way of verifying data.

For example, a checksum (like that calculated by the Unix `sum` command) is used in file transfer protocols (like those used by `ftp` and `rcp`) to verify the data. (Checksum is a running XOR of the data.) The last digit of an ISBN is a check against incorrect copying. Parity check bits are like insurance: they are redundant, adding no new information, but instead re-represent existent information, so that mistakes can be caught.

完全なノイズのない世界ではチェックは必要ありません。しかし、現実には記憶装置と通信システムにはデータ (が正しいかどうか) を確認する手段が必要です。

例えば、チェックサムは磁気ファイル転送プロトコルで、データを確認するのに使われます。ISBN の最後の数字は、写し間違えをチェックします。パリティチェックビットは、保険のようなもので、余分なものです。それは新しい情報を提供する訳ではありませんが、間違いが検出されるように代替現存する情報を再提供するのです。

6.2 Modulo Operation; 剰余演算

"Even parity" means that the *total* number, including the check bit, of asserted bits (= 1) is even. A(n even) parity check bit may be calculated by, equivalently,

- taking the sum mod 2 of the data bits,
- taking the LSB (**l**east **s**ignificant **bi**t) of the sum of bits, or
- taking the XOR (e**x**clusive **or**) of the data bits, "⊕"

偶数パリティの意味は、チェックビットも含め1であるビットの合計が偶数であることをいいます。偶数パリティのチェックビットは以下の方法で同様に求められます。

- データビットの和の mod 2 をとる。
- データビットの和の LSB (**l**east **s**ignificant **bi**t, 最下位ビット) をとる。
- または、続けてすべてのデータビットの XOR (e**x**clusive **or**, 排他的論理和) をとる、「⊕」。

6.3 Parity Check Codes; パリティチェック 符号

As seen in Table 2.6, Unicode and kanji codes such as EUC, JIS, and shift-JIS are two-byte codes. ASCII is a single-byte code for English, but it uses only 7 of the byte's 8 bits.

表2.6 の中に示されているように EUC、JIS、shift-JIS。ASCII は英語のための1バイトのコードです。しかし、それは、バイトの8ビットのうち7ビットだけを使用します。

Example: given a seven bit ASCII character $c_1 \ldots c_7$, a parity check bit c_8 can be calculated as

例: 7ビットの ASCII 文字 $c_1 \ldots c_7$ が与えられとします。パリティチェックビット c_8 は以下のように求められます。

$$\begin{aligned} c_8 &= (c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7) \bmod 2 && \text{(6.2a)} \\ &= \mathrm{LSB}(c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7) && \text{(6.2b)} \\ &= c_1 \oplus c_2 \oplus c_3 \oplus c_4 \oplus c_5 \oplus c_6 \oplus c_7 && \text{(6.2c)} \end{aligned}$$

Then

よって

$$\begin{aligned} (c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7 + c_8) \bmod 2 &= 0 && \text{(6.3a)} \\ \mathrm{LSB}(c_1 + c_2 + c_3 + c_4 + c_5 + c_6 + c_7 + c_8) &= 0 && \text{(6.3b)} \\ c_1 \oplus c_2 \oplus c_3 \oplus c_4 \oplus c_5 \oplus c_6 \oplus c_7 \oplus c_8 &= 0 && \text{(6.3c)} \end{aligned}$$

The parity check bits are calculated as the data is being written or transmitted, and then confirmed as it is read or received, where if inconsistent, some error-handling procedure is invoked.

パリティチェックビットはデータが書かれたり、転送される時に計算されます。そして再び読まれたり、受信される時に、もし、食い違っていればエラー処理をする手続きが呼ばれます。

Parity checks are the simplest kind of error detection. They don't work (they fail to detect errors) when there are an even number of errors, and can not correct detected errors.

パリティチェックはエラー検知の中でも一番簡単なものです。偶数個のエラーがあった場合には作用しませんし、エラーを訂正することもできません。

Forward error correction extends the notion of a parity check, using multiple check bits to determine the position of an error. The data bits can be blocked into groups, and then analyzed as a group.

前進型誤信号訂正はパリティチェックの考え方を拡張し、エラーの起こった箇所を調べるために、複数のチェックビットを使います。データビットはグループに分割され、そしてグループで分析されます。

6.4 Rectangular Codes; 長方形符合

Rectangular codes add up mod 2 the columns and rows of a rectangular block of data bits. For example, say the 10-bit vector 1011011101 is to be transmitted. The bits can be arranged as a 2×5 matrix

長方形符合は、データビットの四角形のブロックの行と列を足して mod2 をとったものです。例えば、10 ビットのベクトル、1011011101 が転送されたとします。それらのビットは 2×5 行列、

$$\begin{pmatrix} 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \end{pmatrix},$$

and the parity bits calculated for each row and column, and transmitted along with the data. (Of course the logical organization is independent of the actual transmission.) Then, upon retrieval, the parity bits are recalculated and compared to those originally sent. In the example below, the second-last data bit, 0, is corrupted, indicated by the strike-out and overwrite, $\not{0}\ 1$. The mismatches in the (*italicized*) parity bits indicate the position of the (single) error, which is then correctable (by toggling).

として表されます。パリティビットは各行と列ことに計算され、データと共に転送されます (もちろん、理論的な転送は物理的な転送に依存しません)。データの訂正についてですが、パリティビットは再計算され、送られてきた元のビットと比較されます。下の例では、$\not{0}\ 1$ と修正されている、後ろから 2 番目のデータビット 0 が間違っています。パリティビット (イタリック体) の中で違っているものが、(1 つの) 修正可能なエラーの箇所を示しています (トグルで)。

$$\begin{array}{ccccccc} 1 & 0 & 1 & 1 & 0 & \textit{1} & \\ 1 & 1 & 1 & \not{0}\,1 & 1 & \textit{0} \neq \textit{1} & \leftarrow \\ \textit{0} & \textit{1} & \textit{0} & \textit{1} \neq \textit{0} & \textit{1} & (\textit{1} \neq \textit{0}) & \\ & & & \uparrow & & & \end{array}$$

6.5 Triangular Codes; 三角形符合

Similarly, data can be organized triangularly, calculating parity bits for a row and column. This scheme uses fewer check bits, but can still locate a single error. For example, using the same data bits as above, each parity bit is calculated for its respective rows and columns together.

同様に、データを三角形に構成することもできます。パリティビットは対角線に沿って計算します。この方法はより少ないチェックビットでできますが、1 つのエラーしか特定できません。例えば、上の例と同じデータビットを使えば、各パリティビットはそれぞれの行と列を一緒に計算して求めます。

$$
\begin{array}{llll}
1 & 0 & 1 & 1 \quad \textit{1} \\
0 & 1 & 1 & \textit{1} \\
1 & \not{\textit{0}}\ 1 & \textit{1} \neq \textit{0} & \leftarrow \\
1 & \textit{0} \neq \textit{1} & & \\
\textit{1} & \uparrow & &
\end{array}
$$

6.5.1 Higher Dimension Parity Checks; 高次元パリティチェック

Besides rectangle and triangle, similar parity checks can be organized around differently-shaped or higher-dimension structures, like a cube, wedge, hypercube, pyramid, etc.

長方形と三角形以外に、同じようなパリティチェックは前述の 2 つと異なる形状のもの、もしくは立方体、楔型、超立方体、ピラミッドなどのような高次元構造のもので構成される。

6.6 Hamming Codes; ハミング符号

Hamming codes further generalize the idea of parity check bits, using a "syndrome" to point at the location of an error. If m check bits are used, they must be able to indicate an error in any of n locations (including both the data and check bits) as well as the "no error" condition:

ハミング符合はより一般化されたパリティチェックビットの考え方で、エラー箇所を示すため、シンドローム (チェックビットの並び) を使います。もし、m 個のチェックビットが使われたら、"エラーのない" 状態も同様に n 箇所 (データとチェックビット両方) にあるエラーを示すことができなければいけません。

$$2^m \geq \overset{\text{locations}}{n} + \overset{\text{no error}}{1} \tag{6.4}$$

For example, a Hamming (7,4) code has a total of 7 bits, of which 4 are data (implying that 3 are check bits). Such a code allows the calculation of such a syndrome that gives the position of any single error. The three check bits, which can be labeled c_1, c_2, and c_4 for convenience, can indicate an error in any of the 7 locations plus the "no error" condition, and are calculated as below and as illustrated in Figure 6.4:

たとえばハミング (7,4) 符合の意味は、合計 7 ビットのうち 4 ビットがデータ (3 ビットがチェックビット) であるということです。そのような符合は 1 個のエラーの位置を得られるシンドロームを計算できます。c_1, c_2, c_4 (便宜のため) の 3 つのチェックビットで、7 ビットのうちのどこかにあるエラーまたは、エラーのない状態を示すことができます。そして、以下のように計算できます。それを図式化したものが図 6.4 です。

$$c_1 = c_3 \oplus c_5 \oplus c_7 \tag{6.5a}$$
$$c_2 = c_3 \oplus c_6 \oplus c_7 \tag{6.5b}$$
$$c_4 = c_5 \oplus c_6 \oplus c_7 \tag{6.5c}$$

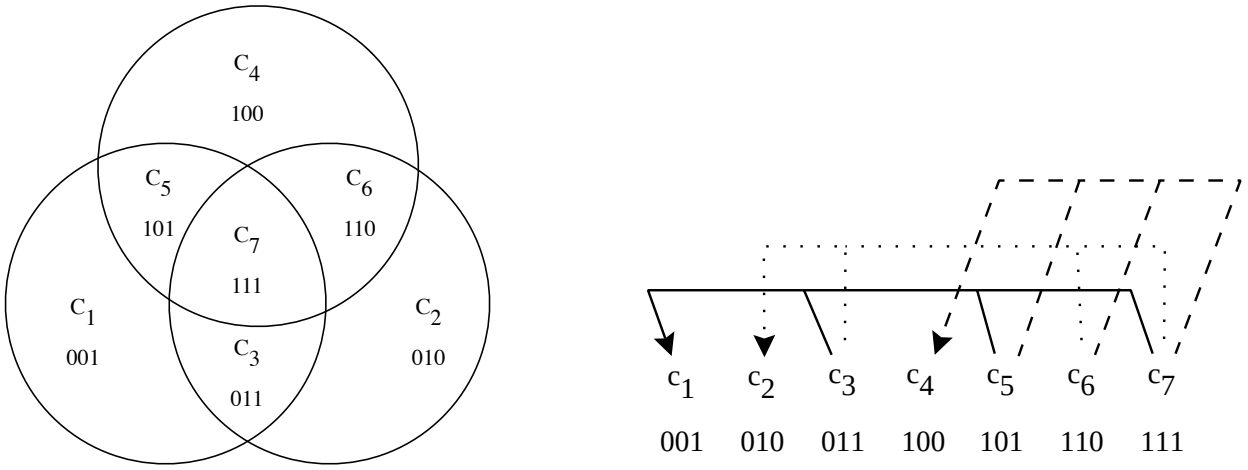
which are of course equivalent to

$$c_1 \oplus c_3 \oplus c_5 \oplus c_7 = 0$$
$$c_2 \oplus c_3 \oplus c_6 \oplus c_7 = 0$$
$$c_4 \oplus c_5 \oplus c_6 \oplus c_7 = 0$$

同様にして

$$c_1 = (c_3 + c_5 + c_7) \bmod 2$$
$$c_2 = (c_3 + c_6 + c_7) \bmod 2$$
$$c_4 = (c_5 + c_6 + c_7) \bmod 2$$

$$(c_1 + c_3 + c_5 + c_7) \bmod 2 = 0$$
$$(c_2 + c_3 + c_6 + c_7) \bmod 2 = 0$$
$$(c_4 + c_5 + c_6 + c_7) \bmod 2 = 0$$



(a) Venn Diagram; ベン図　　(b) Parity Bit Calculation; パリティビット

図 6.4: Hamming (7,4) Code; ハミング (7,4) 符合

The syndrome is calculated by verifying the check bits, effectively comparing the write-time and read-time bits. For instance, for data 1011, a hypothetical error in the first data bit is corrected as follows:

シンドロームはチェックビットの確認によって計算します。書き込み (転送) 時のビットと読み込み (受信) 時のビットを効率よく比較します。例えば、データ 1011 について、最初のデータビット中の仮想のエラーは以下のように修正されます。

		positions						
c_1	c_2	c_3	c_4	c_5	c_6	c_7		
—	—	1	—	0	1	1	data	(4 bits)
0	1	1	0	0	1	1	encode	(7 bits)
		/					error	
0	1	0	0	0	1	1	receive	

$$
\begin{aligned}
\text{syndrome}_1 &= c_1 \oplus c_3 \oplus c_5 \oplus c_7 \\
&= 0 \oplus 0 \oplus 0 \oplus 1 \\
&= 1 \\
\text{syndrome}_2 &= c_2 \oplus c_3 \oplus c_6 \oplus c_7 \\
&= 1 \oplus 0 \oplus 1 \oplus 1 \\
&= 1 \\
\text{syndrome}_4 &= c_4 \oplus c_5 \oplus c_6 \oplus c_7 \\
&= 0 \oplus 0 \oplus 1 \oplus 1 \\
&= 0
\end{aligned}
$$

So the syndrome, written with MSB leftmost as (syndrome_4, syndrome_2, syndrome_1), is (binary) 011, or (decimal) 3, implying an error in location c_3, which is then correctable by toggling.

左 (MSB) から (syndrome_4, syndrome_2, syndrome_1) と書かれたシンドロームは、011 (２進数) または 3 (１０進数) となり、c_3 にエラーがあることを示しています。それは、そのビットを反転することで修正できます。

6.7 Generator Matrix and Parity Check Matrix; 生成 (ジェネレーター) 行列とパリティチェック行列

Single error detection parity codes use a single parity check over all the positions. If we write a 1 for each position that is checked, and a 0 otherwise, we get the (in this case, trivial $m \times 1$ row) matrix

単一エラーの発見方法であるパリティーコードは全ての位置に対してパリティーチェックを行います。またチェックを行った位置に 1 を、それ以外の場所に 0 を書くと以下のような m 列 1 行の表を得ることが出来ます。

$$M = (1, 1, 1, \ldots, 1) = \vec{1}$$

Consider an encoded symbol $\vec{c}$, a row vector of n bits (including both data and check bits):

ここで符号化されたシンボル $\vec{c}$ を考えます。

$$\vec{c} = (0, 1, 0, \ldots, 1)$$

The transpose of $\vec{c}$, $(\vec{c})^T$, is the corresponding column vector:

$\vec{c}$ を対応する列ベクトル、$(\vec{c})^T$、に移行してみます。

$$(\vec{c})^T = \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 1 \end{pmatrix}.$$

Then

その時、

$$M(\vec{c})^T = 0 \tag{6.9}$$

for every encoded word to be sent.

送られる全ての符号化されたデータに対して上記が成り立ちます。

Suppose a single error is made in sending code symbol $\vec{c}$, changing a bit. Since the arithmetic is mod 2, the received word $\vec{r}$ is

単一のエラーが送信中に起こった場合は、シンボル $\vec{c}$ が 1 ビット変化します。その計算式は、2 で割った余りだから、受信されたデータ $\vec{r}$ は、

$$\vec{r} = \vec{c} \oplus \vec{e} \tag{6.10}$$

where $\vec{e}$ is the vector of all 0s except a 1 in the position of the error. Generally,

$\vec{e}$ はエラーが起きた位置が 1、それ以外の全てが 0 となるベクトルです。一般的に

$$\begin{aligned} M(\vec{r})^T &= M(\vec{c} \oplus \vec{e})^T && \text{(6.11a)} \\ &= \cancel{M(\vec{c})^T} \oplus M(\vec{e})^T && \text{(6.11b)} \\ &\neq \vec{0} \text{ (for a single error)} && \text{(6.11c)} \end{aligned}$$

The syndrome of the erroneous message (the result of applying the parity checks) in this case is a 1 whenever there is an odd number of errors, and a 0 whenever there is an even number of errors, including no errors.

この場合のエラーメッセージ（パリティチェックの結果）のシンドロームは奇数のエラー番号がある場合に 1 それ以外の場合は 0 となります。

A Hamming (7,4) code uses the parity check matrix

ハミング (7,4) コードは行列を使います

$$M = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}. \tag{6.12}$$

Note that columns of the check matrix are simply ordinal numbers (because of the choice of generating matrix, explained in the following subsection), so that the syndrome becomes simply the position of the error, simplifying the correction. Again letting $\vec{c}$ be a code word, we have for all code words

一群が単純にエラーの位置になるようにして修正を単純化できるように、チェック行列の各列は単純な序数になっている。(生成行列の選択だから)。$\vec{c}$ をあるコードデータとして、全てのコードデータに対して以下を得ることができます。

$$M(\vec{c})^T = \vec{0} \tag{6.13}$$

since this is really the definition of an encoded word: the matrix M annihilates every encoded word $\vec{c}$. A single error $\vec{e}$ corrupts the received message

これは実際に符号化されたデータの定義です。行列 M は全ての符号化されたデータ $\vec{c}$ を無効にします。単一エラー e は以下のメッセージを与えます。

$$\vec{r} = \vec{c} \oplus \vec{e} \tag{6.14}$$

and we have again

また以下を得ます。

$$\begin{aligned} M(\vec{r})^T &= M(\vec{c} \oplus \vec{e})^T && \text{(6.15a)} \\ &= \cancel{M(\vec{c})^T} \oplus M(\vec{e})^T && \text{(6.15b)} \\ &= \text{the syndrome; シンドローム} && \text{(6.15c)} \end{aligned}$$

Thus the syndrome of the error depends only on the position of the error, and not on the code word sent.

このように、エラーのシンドロームはエラーによるものであり、送られた言葉によるものではありません。

Examination of the check matrix through multiplication by the error vector (which has a single 1 in it) extracts the syndrome that is the corresponding column of the matrix. Thus the columns of the parity-check matrix are all the syndromes that can appear for any single error.

エラー行列（1 をエラーの位置に持つ）による行列の掛け算はシンドロームが完全に一致する列であることを証明することができます。そのようにして、パリティチェック行列の列は複数の単一エラーを表せるシンドロームになります。

6.7.1 Example: Hamming (7,4) Encoding and Decoding

One way to encode a message would be to use an explicit dictionary, as in Table 6.19.

メッセージを符号化するひとつの方法は次のとおりです。表 6.19 参照。

(index of) message	codeword (LSB–MSB)
0000	0000000
0001	1101001
0010	0101010
0011	1000011
0100	1001100
0101	0100101
0110	1100110
0111	0001111
1000	1110000
1001	0011001
1010	1011010
1011	0110011
1100	0111100
1101	1010101
1110	0010110
1111	1111111

表 6.19: Hamming (7,4) Codeword Dictionary: Dimension 2^4 Entries × 7 Bits; 次元: $2^4 \times 7$. Any two of the code words differ in at least three positions. Note that not all possible codewords are used, like noisy typewriter with non-overlapping inputs. ハミング (7, 4) の符号語辞書：次元 2^4 の入力 ×7 bits；次元 $2^4 \times 7$。符号語の二つは少なくとも三か所異なります。全ての可能な符号語が使用されない点に注意し、重複しない入力を雑音が多いタイプライターに嘘をついてください。

But this is cumbersome. For instance, a Hamming(15,11) code would need a 2^{11} entry × 15 bits dictionary. A better model is to use matrix multiplication, performed in the usual way:

しかし、これは厄介です。たとえばハミング (15,11) コードは 2 の 11 乗の入力と 15 ビットの辞書が必要です。よりよい方法は次に示される行列の掛け算を使用することです。(通常はこの方法で行われます)

$$C = AB \tag{6.16a}$$

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj} \tag{6.16b}$$

To encode a message using a Hamming (7,4) code, it can be multiplied by an appropriate generator matrix. Such an encoding need only store n rows of generator matrix G (dimension: 4×7) instead of 2^n vectors of the code (dimension of codeword dictionary above: $2^4 \times 7$).

ハミング (7,4) を使ってメッセージを符号化するために適切なジェネレーター行列を掛けることができます。このような符号化はコード (上記のコード、次元: $2^4 \times 7$) の 2^n ベクトルの代わりにジェネレーター行列 G(次元: 4×7) の n 列を格納する必要があります。

$$(\vec{c})^T = \overset{\text{generator matrix}}{\mathrm{G}} (\vec{x})^T \tag{6.17a}$$

$$= \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} \tag{6.17b}$$

This, or the following equivalent, are the algebraic expression of Figure 6.4.

これ、または以下のものと同等なのは図 6.4 に示される代数式である。

$$\vec{c} = \vec{x}G^T = (x_1 x_2 x_3 x_4) \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{6.18}$$

Note that the unity-weighted 3rd, 5th, 6th, and 7th rows of the generating matrix let the data bits "pass through." To decode a received vector, use the corresponding parity check matrix to calculate the syndrome, the location of a single error. (The LSB of the syndrome is indicated by the top row, the middle bit by the middle row, and the MSB by the bottom.)

生成行列の同じ重みを持った3,5,6,7 行はデータビットを通り抜けさせます。受信ベクトルをデコードするためには、単一のエラーの位置を計算するためにパリティーチェック行列を使用してください。(シンドロームの最下位ビットは、一番上の列、中央の列の中央のビットと一番下の最上位ビットによって示されます。)

$$\overset{\text{syndrome}}{(\vec{s})^T} = M \overset{\text{received}}{(\vec{r})^T} \tag{6.19}$$

For example, if $\vec{x} = (0110)$ is the message, it is encoded as above as $\vec{c} = (1100110)$. If an error $\vec{e} = (0000100)$ corrupts the 5th position, the received signal vector becomes $\vec{r} = (1100010)$. The error location can be discovered through multiplying by the parity check matrix so the original message can be recovered (by toggling the bit indicated by the syndrome).

例えば、$\vec{x} = (0110)$ がメッセージである場合、$\vec{c} = (1100110)$ としてコード化されます。エラー $\vec{e} = (0000100)$ によって 5 番目のメッセージが間違っている場合、受信信号ベクトルは $\vec{r} = (1100010)$ になるだろうが、パリティーチェック行列の掛け算をすることによりエラー位置を発見することができ、元のメッセージを回復することができます (一群に示されたビットを切り替えることにより)。

$$
\begin{aligned}
(\vec{s})^T &= M(\vec{r})^T \\
&= M(\vec{c} \oplus \vec{e})^T \\
&= M(\vec{c})^T \oplus M(\vec{e})^T \\
&= \overbrace{\begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}}^{M} \overbrace{\begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}}^{\vec{c}=G\vec{x}^T} \oplus \overbrace{\begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}}^{M} \overbrace{\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}}^{\vec{e}} \\
&= \begin{pmatrix} 1 \oplus 0 \oplus 1 \oplus 0 \\ 0 \oplus 1 \oplus 0 \oplus 1 \\ 0 \oplus 0 \oplus 1 \oplus 1 \end{pmatrix} \oplus \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \\
&= \vec{0}^T \oplus (101)^T
\end{aligned}
$$

This is simply the 5^{th} column of the parity check matrix.

次の5行はパリティチェック行列の簡単な例です。

6.8 (Group-Organized Codes; 組織符号)

6.9 Minimum Distance of a Code; 符号の最小距離

The elements of a vector of length n may be thought of as coordinates in n-space. The weight of a binary vector is the number of asserted positions.

長さ n のベクトルの要素を n 次元空間の原点のように考えるとします。2進数のベクトルの重みは確立されたポジションの数である。

$$\text{weight}(1, 0, 1, 0, 1, 0) = 3$$

For two binary vectors $\vec{x}$ and $\vec{y}$ both of length n, the distance is simply the number of positions that differ between them:

2つの2進数のベクトル $\vec{x}$ と $\vec{y}$ は両方とも長さ n で、その距離はそれぞれの位置でくい違ったビット数になります。

$$\vec{x} = (x_1, x_2, \ldots, x_n) \tag{6.20a}$$
$$\vec{y} = (y_1, y_2, \ldots, y_n) \tag{6.20b}$$

$$\text{(binary) distance}(\vec{x}, \vec{y}) = \sum_{i=1}^{n} |x_i - y_i| \tag{6.21}$$

where $|x_i - y_i|$ naturally equals XOR $(x_i \oplus y_i, x_i \neq y_i)$.

$|x_i - y_i|$ は XOR $(x_i \oplus y_i, x_i \neq y_i)$ 同値である。

Note that this corresponds not to Euclidian (a.k.a. Cartesian) distance, which would be

ただし、これは次式に示されるユークリッド距離 (デカルトとも呼ばれる) とは一致しません。

$$\text{Euclidean distance}(\vec{x},\vec{y}) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}, \tag{6.22}$$

but to "Manhattan distance" or city block distance, suggesting the impossibility of cutting diagonally through dense buildings, but instead having to walk rectilinearly around them, as shown in Figure 6.5.

でも、"マンハッタン距離" では、超高層ビルを斜めに横切って通ることができないので、代わりにビルのまわりを直角に曲がって歩きます。図 6.5 参照

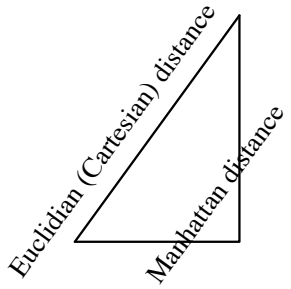


図 6.5: Cartesian and Manhattan Distances; 観念的距離とマンハッタン距離

For example, the Cartesian distance between (0,0,0) ↔ (1,1,0) is $\sqrt{2}$, but the binary distance is 2. The binary distance is still a legitimate distance function, as it satisfies the three conditions:

例をあげると、(0,0,0) ↔ (1,1,0) の ユークリッド 距離は $\sqrt{2}$ ですが、2 進数の距離は 2 になります。2 進数の距離は次の 3 つの状態を満たす正規距離関数です。

conservative The distance from a point to itself is always zero.

保守的 ある点からそれ自身への距離は常に 0。

$$\vec{x} = \vec{y} \Rightarrow d(\vec{x},\vec{y}) = 0 \tag{6.23a}$$

commutative The distance from one point to another is the same as the opposite direction, and is positive.

交換的 1 点から他の点への距離が逆方向にしても等しく、かつ正。

$$d(\vec{x},\vec{y}) = d(\vec{y},\vec{x}) \geq 0 \tag{6.23b}$$

triangle inequality The length of a double-segment path is at least as great as a single-segment path with the same endpoints.

三角不等式 どこかを経由した 2 つの経路からなる経路の距離は、同じ始点と終点を持つ 1 つの経路の距離以上になる。

$$d(\vec{x},\vec{y}) + d(\vec{y},\vec{z}) \geq d(\vec{x},\vec{z}) \tag{6.23c}$$

The minimum distance in a code is the smallest distance between any two code words. Table 6.20 summarizes properties of different minimum distances.

符合の最小距離は符合のどんな 2 つの要素に対しても小さくなるような距離です。表 6.20 は最小距離の違いによる特性についてまとめています。

pattern パターン	minimum distance 最小距離	detection/correction 検出/訂正	example 例
·	0	singular (not uniquely decodable)	{(0), (0)}
—	1	uniquely decodable	{(0), (1)}
⋀	2	single error detect	{(0,0), (1,1)}
⋁	3	single error correct or double error detect	{(0,0,0), (1,1,1)}
⋀⋀	4	single error correct and double error detect, or triple error detect	{(0,0,0,0), (1,1,1,1)}
⋁⋁	5	double error correct…	{(0,0,0,0,0), (1,1,1,1,1)}
⋮	⋮	⋮	⋮

表 6.20: Properties of minimum distance; 最小距離のせいしつ

6.10 Vectors and Polynomials; ベクトルと多項式

Code sequences of 0s and 1s can be regarded as vectors. Further, they can be considered polynomials; the digits become the coefficients of powers of a variable. So the sequence (with MSB left-most)

ベクトルは 0 と 1 の数列として見なすことが出来ます。さらに、多項式としてもみなすことが出来ます。0 と 1 が累乗の計数になるので、以下の数列（左端の最上位ビットを含む）

$$(1, 0, 1, 1, 0)$$

could be considered the polynomial

は以下の多項式として考えられます。

$$P_1(x) = x^4 + x^2 + x.$$

Another example is the sequence

他の例として以下の数列

$$(0, 1, 1, 0, 1)$$

which could correspond to the polynomial

は以下の多項式と一致します。

$$P_2(x) = x^3 + x^2 + 1.$$

The set of all polynomials of a given degree with binary coefficients permits addition and subtraction (which in the case of modulo 2 arithmetic are the same). For example, using the above two polynomials,

整数の係数により与えられる多項式同士は足し算、引き算することが出来ます。(2 進であるこの場合も同じです）例えば、上の 2 つの多項式を使うと

$$P_1(x) + P_2(x) = x^4 + x^3 + x + 1$$

Note that addition is associative:

足し算は順序を入れ替えることができるので以下となります。

$$(P_1 + P_2) + P_3 = P_1 + (P_2 + P_3). \tag{6.24}$$

A zero vector $\vec{0} = (0, 0, 0, 0, 0)$ corresponds to the zero polynomial and is identically zero. Since each polynomial is its own additive inverse, and the sum of any two polynomials is another polynomial of (at most) the same degree, the set of polynomials of degree n forms an algebraic group. (Remember that the coefficients of these polynomials can be only 0 or 1.)

ゼロ要素 $\vec{0} = (0, 0, 0, 0, 0)$ はゼロ多項式つまり 0 に一致します。$\vec{0}$ を含まない多項式はそれ自身の逆数であり 2 つの多項式の和はそれらの和と等しい他の多項式であるから、値が n の多項式の組は 代数学の グループを形成するといえます。（係数は 0 か 1 にしかなりません。）

One of the purposes of modular arithmetic is to ensure that all the numbers that can arise will fall within a given range. This is accomplished in modular arithmetic by dividing all numbers by a modulus m and using only the m remainders— $0, 1, \ldots, m-1$— which are all less than the modulus.

モジュール計算の目的の 1 つは与えられた範囲の中での情報落ちする全ての数字を確実なものにすることです。全ての数字を m のモジュールにより分割し、m の余り $0, 1, \ldots, m-1$（それらはモジュールより数が少ない）を使用することにより達成されます。

Considering polynomial arithmetic, note that the product of two polynomials has a degree equal to the sum of the degrees of the individual polynomials. To keep the product within a maximum degree less than n (so that the polynomial can be identifiable with a code sequence), we must similarly reduce the degree. Thus we need a polynomial to use as a modulus. Just as with modular arithmetic in which all numbers are divided by a modulus and the remainder is identified as the result, so we will now divide all polynomials by a modulus polynomial of degree n. As a result, the reminder will have n coefficients (the degree is $n-1$ since there is a zero-order coefficient), which can then be identified as a code symbol.

多項式を考えたとき 2 つの多項式の積が個別の多項式の和に等しいことが分かります。n （多項式は数列とみなせる）より小さい最大値の範囲内の積を維持するために、値の方も同様に減らさなければなりません。そこで、多項式をモジュールとして使用する必要があります。全ての数字をモジュールによって分割することと数字として余りを認識するだけですが、そのように値 n のモジュール多項式により全ての多項式を分割します。結果、余りは n の係数（値は $n-1$）を持つことになり、それはコードシンボルとして認識できます。

For a prime modulus, if the product of two numbers is zero, then at least one of the two numbers is zero. (If the modulus is not a prime, it is possible that neither number is zero although their product is.)

原始多項式に対して、2 つの数字の積が 0 の場合、 少なくともどちらかの数字が 0 になります。（もしそれらのモジュールが原始モジュールでない場合は、それらの積がそうでなくても、両方の数字が 0 ではないことがあります。）

Suppose we have numbers a and b respectively congruent to a' and b' modulo a given modulus m. This means that

a' および b' に一致する数 a および b に対しそれぞれ与えられた係数 m を持っていると仮定する。この意味は

$$a' \triangleq a \bmod m \tag{6.25a}$$
$$a \equiv a' \pmod m \tag{6.25b}$$
$$b' \triangleq b \bmod m \tag{6.25c}$$
$$b \equiv b' \pmod m \tag{6.25d}$$

or

$$a = a' + k_1 m \tag{6.26a}$$
$$b = b' + k_2 m \tag{6.26b}$$

for some integers k_1 and k_2. For product ab,

ある整数 k_1 および k_2 のために。ab を導くために、

$$ab = a'b' + a'k_2 m + b'k_1 m + k_1 k_2 m^2 \tag{6.27a}$$
$$\equiv a'b' \pmod m \tag{6.27b}$$

For example of the danger of using a composite modulus, consider

コンポジットモジュールを使う危険性の例として、

$$a = 15$$
$$b = 12$$
$$m = 10 \quad (= 2 \times 5, \text{composite; コンポジット}).$$

Then

$$a' = 5$$
$$b' = 2$$
$$ab \equiv a'b' \equiv 0 \pmod{10}.$$

But neither a nor b is zero! Only for a prime modulus does the property hold that if a product is zero, then at least one factor is zero. (Otherwise the factors of the modulus might separately divide the multiplicands.)

しかし a も b もどちらも 0 でありません！原始多項式に対して、もし 0 を生成した場合少なくとも 1 つの要素は 0 です。(もしくは、要素の係数が別々に被乗数で割られているかです。)

For example, the ISBN check digit will detect transposed digits, single digit errors, and most (90%) other errors.

例えば ISBN チェックは数字などを転換した場合、1 桁のエラーの検知、および (90%) 以上の確率で他のエラーも検知します。

$$x_{10} = \left(\sum_{i=1}^{9} i \; x_i \right) \bmod 11 \tag{6.28}$$

$$x_{13} = (10 - (x_1 + 3x_2 + x_3 + 3x_4 + \cdots + x_1 1 + 3x_1 2) \bmod 10) \bmod 10 \tag{6.29}$$

Similarly, we want a prime polynomial as our modulus— a prime polynomial being one that cannot be represented as the product (in the field of coefficients) of two non-trivial polynomials.

また、モジュールとして原始多項式を必要とします。原始多項式とは一致しない複数の多項式の積として表現することが出来ないもののことです。

A polynomial of degree (largest exponent) n might by chance have coefficient 0 of the highest power so that the polynomial was essentially of lower degree. Since there are only 0 and 1 in the field of coefficients, the phrase "monic polynomial" is used to denote a polynomial for which the coefficient of the highest power of x is 1.

値が n （指数）の多項式はそれ自身の係数が 0 の最大次数を持つと、多項式が本当に低い値になりました。だから係数は 0 か 1 しか持ち得ません。また最高次の係数が 1 の多項式を示すために一般語「monic polynomial」を使用します。

Why would one want to multiply polynomials together? The answer is simply that we want to introduce as much structure into the coding theory as we can. Then the codes finally obtained will be highly structured, and as a result the encoding and decoding equipment can also be highly structured (meaning easy to build in hardware or program in software).

なぜ多項式を掛け算の形で表すのでしょうか?答え単に出来るだけ沢山の構造を符号化理論においては持たせたいからです。そのときだけ、最終的に得るコードは高い構造になり、また符号化と複合化に使用するものもまた高い構造になることを望みます。（ソフトウェアを簡単に構築、プログラムことを意味しています。）

6.11 Prime Polynomials and Extensions; 原始多項式と拡大体

6.11.1 Prime Polynomials; 原始多項式

A prime polynomial is a monic polynomial which cannot be factored into a product of lower-order polynomials. But since our field of coefficients uses arithmetic modulo 2, we cannot be sure of how familiar results will now manifest. Thus we will now experimentally explore what can happen, avoiding the technical details of finite (Galois) fields to prove how factorization must occur.

原始多項式とは低位次数の多項式により因数分解できない最高次の係数が 1 の多項式のことです。しかし係数の値に 2 の余りを使用しているので、どんな値が現れるのか確信が持てません。程度 0 の多項式については、単一で重要でない多項式 (程度 0 の多項式を多項式と呼びたいと思う場合) があります。

For polynomials of degree 0, there is only the single, trivial polynomial (if one cares to call such a degenerate case a polynomial), namely

$$P = 1, \tag{6.30}$$

corresponding in normal arithmetic to the number 1. (Just as we disregard the use of 1 as a factor when factoring integers, so must we disregard the trivial polynomial "1" as a factor when factoring polynomials.) For monic polynomials of degree 1, there are only two,

普通数学の数字の 1 に一致します。(1 を整数の要素として無視するのと同様に値が 1 の多項式を要素としては無視します) 値が 1 の最高次の係数が 1 の多項式については 2 つの多項式しかありません。

$$x \tag{6.31a}$$

$$x + 1, \tag{6.31b}$$

both of which are prime polynomials. For monic polynomials of degree 2, there are four possibilities (as each of the coefficients of lower powers x $[x^1]$ and unity $[x^0]$ may be 0 or 1):

上記の両方とも原始多項式です。値が 2 の最高次の項の係数が 1 の多項式については 4 つあります。($x[x^1]$ と単位元 $[x^0]$ の係数はそれぞれ 0 もしくは 1 です)

$$x^2 = x \cdot x \tag{6.32a}$$

$$x^2 + 1 = (x + 1)(x + 1) \tag{6.32b}$$

$$x^2 + x = x(x + 1) \tag{6.32c}$$

$$x^2 + x + 1 \quad \text{prime; 原始多項式} \tag{6.32d}$$

The fact that

実際は、

$$x^2 + 1 = (x + 1)^2$$

might seem surprising at first, but multiplying out the right side obtains

最初は驚くかもしれませんが、上記右辺の掛け算の結果は、

$$x^2 + 2x + 1.$$

Since $2 \equiv 0 \pmod 2$,

ここで $2 \equiv 0 \pmod 2$ なので、以下を得ます。

$$x^2 + 2x + 1 = x^2 + 1.$$

Consider the polynomial

以下の多項式を考えます。

$$x^2 + x + 1.$$

Why is it prime? Try dividing by the two lower-degree polynomials. If $x^2 + x + 1$ is composite, these are the only two possible factors. Clearly x is not a factor. Then try $x + 1$:

これは原始多項式でしょうか？ 2 つの低位次数多項式を使用して因数分解してみましょう。もし $x^2 + x + 1$ が合成多項式の場合は、2 つの因数しか持ちません。しかし x は因数ではありません。そこで $x + 1$ で試してみましょう。

$$
\begin{array}{r|l}
 & x+0 \\ \hline
x+1 & x^2+x+1 \\
 & x^2+x \\ \hline
 & 0+1 \\
 & 0 \\ \hline
 & 1 \text{ remainder; 余り}
\end{array}
$$

So x^2+x+1 is a prime polynomial. For further practice, examine the eight possible cubics:

よって x^2+x+1 は原始多項式となります。さらに 8 つの 3 次の多項式を調べます。

$$
\begin{aligned}
x^3 &= x \cdot x \cdot x && \text{(6.33a)} \\
x^3+1 &= && \text{(6.33b)} \\
x^3+x &= x(x^2+1) && \text{(6.33c)} \\
x^3+x+1 &= && \text{(6.33d)} \\
x^3+x^2 &= x^2(x+1) && \text{(6.33e)} \\
x^3+x^2+1 &= && \text{(6.33f)} \\
x^3+x^2+x &= x(x^2+x+1) && \text{(6.33g)} \\
x^3+x^2+x+1 &= && \text{(6.33h)}
\end{aligned}
$$

where the four obvious factorizations have been given, leaving the others blank as an exercize for the reader.

4 つの因数分解は与えられているので、残りの空白を埋めてみてください。

If a cubic can be factored, then at least one factor must be of the first degree. Since the polynomials divisible by x have been factored above, we need only try $x+1$. For x^3+1,

もし 3 次のものが因数分解できるのならば、一つの因数は 1 次のものになります。よって、x による因数分解は上に記されているので、あとは $x+1$ を試すだけです。x^3+1 に対して、

$$
\begin{array}{r|l}
 & 1\ 1\ 1 \\ \hline
1\ 1 & 1\ 0\ 0\ 1 \\
 & 1\ 1 \\ \hline
 & 1\ 0 \\
 & 1\ 1 \\ \hline
 & 1\ 1 \\
 & 1\ 1 \\ \hline
 & 0
\end{array}
$$

Hence,

従って、

$$(x+1)(x^2+x+1) = x^3+1.$$

Next try to factor x^3+x+1:

x^3+x+1 に対しても行ってみましょう。

$$
\begin{array}{rr|rrrr}
 & & & 1 & 1 & 0 \\
\hline
1 & 1 & 1 & 0 & 1 & 1 \\
 & & 1 & 1 & & \\
\hline
 & & & 1 & 1 & \\
 & & & 1 & 1 & \\
\hline
 & & & & 0 & 1 \\
 & & & & 0 & 0 \\
\hline
 & & & & & 1
\end{array}
$$

So we can note that x^3+x+1 is prime. Similarly $x^3 + x^2 + 1$ is also prime, observing that

$x^3 + x + 1$ が原始多項式であることは記したので、類似して $x^3 + x^2 + 1$ も下記により原始多項式となります。

$$x^3 + x^2 + 1 = (x \cdot x)(x + 1) + 1.$$

Finally, try to factor $x^3 + x^2 + x + 1$. From the experience above, one can see how to factor directly. One can write

最後に $x^3 + x^2 + x + 1$ に対しても行ってみましょう。上記によりどのように因数分解すれば分かります。下記のように記せて、

$$x^3 + x^2 + x + 1 = x^2(x + 1) + (x + 1) = (x^2 + 1)(x + 1) = (x + 1)^3$$

and observe that it is composite. Therefore, we have

また、合成多項式であることが分かります。従って、以下を

$$x^3 + x + 1$$
$$x^3 + x^2 + 1$$

as the only two prime monic polynomials of degree 3. For each particular polynomial, we need only try lower-order polynomials, factoring up to (the floor of) half the degree of the given polynomial to establish primality, the same way at least one factor of an ordinary composite number will be no greater than its square root.

2 つしかない最高次の係数が 1 の 3 次の原始多項式として得ます。個々の高次多項式については、与えられた多項式の半分の次元の低次多項式を試みる事だけを素数性を証明するために必要とします、同様にある合成数の少なくとも 1 つの要素はその平方根より小さくなる。

6.11.2 Primitive Roots; 原始累乗根

The n^{th} roots of unity are the n solutions of

1 の n 乗根は以下の累乗根となります。

$$x^n = 1 \tag{6.34a}$$
$$x^n - 1 = 0. \tag{6.34b}$$

The first root is

最初の累乗根は、

Degree 1	Degree 2	Degree 3	Degree 4
x	x^2	x^3	x^4
$x+1$	$x^2+1=(x+1)^2$	$x^3+1=(x+1)(x^2+x+1)$	$x^4+1=(x+1)(x^3+x^2+x+1)$
	$x^2+x=x(x+1)$	$x^3+x=x(x^2+1)$	$x^4+x=x(x^3+1)$
	$\boxed{x^2+x+1}$	$\boxed{x^3+x+1}$	$\boxed{x^4+x+1}$
		$x^3+x^2=x^2(x+1)$	$x^4+x^2=x^2(x^2+1)$
		$\boxed{x^3+x^2+1}$	$x^4+x^2+1=(x^2+x+1)^2$
		$x^3+x^2+x=x(x^2+x+1)$	$x^4+x^2+x=x(x^3+x+1)$
		$x^3+x^2+x+1=(x+1)(x^2+1)$	$x^4+x^2+x+1=(x^3+x^2+1)(x+1)$
			$x^4+x^3=x^3(x+1)$
			$\boxed{x^4+x^3+1}$
			$x^4+x^3+x=x(x^3+x^2+1)$
			$x^4+x^3+x+1=(x+1)(x^3+1)$
			$x^4+x^3+x^2=x^2(x^2+x+1)$
			$x^4+x^3+x^2+1=(x+1)(x^3+x+1)$
			$x^4+x^3+x^2+x=x(x^3+x^2+x+1)$
			$\cancel{x^4+x^3+x^2+x+1}$

表 6.21: Prime Polynomials with Primitive Roots

$$x = \sqrt[n]{1} \tag{6.35}$$

but the explicit form of all the roots is given by　しかし、すべての累乗根の明示的な形は以下のように与えられる、

$$e^{2\pi ik/n}, \qquad k = 0, 1, 2, \ldots, n-1 \tag{6.36}$$

which uses Euler's equation,

$$\overset{\text{complex}}{e^{i\theta}} = \overset{\text{real}}{\cos\theta} + \overset{\text{imaginary}}{i\sin\theta}, \tag{6.37}$$

where

$$i = \sqrt{-1}, \tag{6.38}$$

a special instance of which is Euler's Identity,

$$e^{\pi i} + 1 = 0. \tag{6.39}$$

This equation is often thought of as one of the most beautiful equations in mathematics, elegantly containing as it does so many basic concepts— including the base of natural logarithms (calculus), exponentiation, the ratio of the circumference of a circle to its diameter (trigonometry), multiplication, the square root of -1 (algebra and complex analysis), addition, unity (arithmetic), equality, and zero (number theory).

この等式は数学の中でも最も美しい等式といわれている、優雅に多くの基本的な概念を含んでいる— 自然対数の底、指数関数、円周率、乗算、-1 の平方根、加算、1、等式、0。

For the unit circle, DeMoivre's Theorem is:

$$(\cos\theta + i\sin\theta)^n = \cos(n\theta) + i\sin(n\theta) \tag{6.40}$$

This is easy to understand when polar representation is used:

$$(e^{i\theta})^n = e^{ni\theta} = e^{i(n\theta)} \tag{6.41}$$

According to the Fundamental Theorem of Algebra, an equation has as many roots as the degree of the equation. The roots of unity always start on the positive real axis, and are evenly spaced around the unit circle in the complex plane (and are vertices of a regular n-sided polygon).

代数学の基本定理で方程式の次数と同数の累乗根があります。累乗根は常に軸の正の部分から始まり、複素平面中の単位円のまわりに等間隔で配置されます。(そして、正 n 角形の頂点から成り立ちます。)

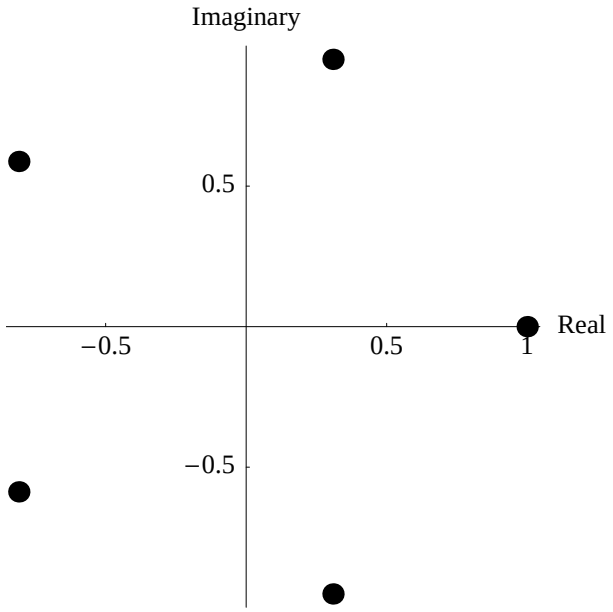


図 6.6: Fifth Roots of Unity

Some of these roots, those that are relatively (mutually) prime (coprime) to n (for example k = 1), are such that their successive powers generate all the other roots, and are therefore called "primitive roots." Primitive roots are important because a primitive root can serve as a generator for a whole set of needed numbers.

これらの累乗根のうち幾つか、n に対応する原始符号（例えば k = 1）、はそれらの連続する累乗が他の累乗根を導き出すものであり、原始累乗根と呼ばれています。原始累乗根は重要です。それらは全ての必要とされている組を導き出すことができます。

As described earlier (Equation 6.12, p. 137) the Hamming (7,4) code uses the check matrix

先に述べたように (方程式 6.12, p. 137)、ハミング (7,4) コードは行列を使用します

$$M = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Any rearrangement of the columns of the matrix M does not fundamentally change the code; it would only affect how and where the column corresponding to the syndrome that emerges when a single error occurs is found. Suppose one uses the prime polynomial

行列 M の列の再構築はコードを根本的には変えません。それはどこでどのように単一エラーが発生時に現れるシンドロームに対応している列を見つけるかということだけに影響します。もし以下の原始多項式

$$x^3 + x + 1$$

as a modulus. Let a be a root of this polynomial. Then

をモジュールとして使用したとします。a は多項式の累乗根とします。すると、

$$a^3 + a + 1 = 0$$

or

または、

$$a^3 = a + 1.$$

degree, no. of roots: n	rectangular (Cartesian)	polar: $k = 0..(n-1)$ degrees: $\frac{360}{n}k$	polar: $k = 0..(n-1)$ radians: $\frac{2\pi}{n}k$	relatively prime to n
1	$\{1\}$	$\{0°\}$	$\{0\}$	1
2	$\{1, -1\}$	$\{0°, 180°\}$	$\{0, \pi\}$	1
3	$\{1, -\frac{1}{2} + i\frac{\sqrt{3}}{2}, -\frac{1}{2} - i\frac{\sqrt{3}}{2}\}$	$\{0°, 120°, 240°\}$	$\{0, \frac{2\pi}{3}, \frac{4\pi}{3}\}$	1, 2
4	$\{1, i, -1, -i\}$	$\{0°, 90°, 180°, 270°\}$	$\{0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}\}$	1, 3
5	...	$\{0°, 72°, 144°, 216°, 288°\}$	$\{0, \frac{2\pi}{5}, \frac{4\pi}{5}, \frac{6\pi}{5}, \frac{8\pi}{5}\}$	1, 2, 3, 4
6	?	?	?	1, 5

表 6.22: Some Roots of Unity: solutions to $x^n = 1$

Powers of a that are 3 or higher can be reduced, using this equation, to sums of lower powers, so every polynomial in a can be reduced to a linear combination of 1, a, and a^2. Consider a column 3-vector which has components 1, a, and a^2. Multiplication by a shifts each component one position, and $a^3 = a + 1$ is used whenever a cube occurs, so that only 1, a, and a^2 appear in the vector. For the various powers of a, the corresponding 3-vectors are:

a の次数が 3 以上のものは減らすことができ、このことを用いて低い次数の和としてどの a の多項式も 1 と a と a^2 の一次結合に減らすことができます。1 と a と a^2 の要素を持つ 3 ベクトルの列を考えます。a による掛け算は、各要素を 1 つシフトし、a^3 がある場合は、$a^3 = a+1$ を使用します。それにより多項式の中には 1 と a と a^2 しか現れません。a の色々な次元は以下となります。

$$a^0 = 1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{6.42a}$$

$$a^1 = a = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \tag{6.42b}$$

$$a^2 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{6.42c}$$

$$a^3 = a + 1 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \tag{6.42d}$$

$$a^4 = a^2 + a = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} \tag{6.42e}$$

$$a^5 = a^3 + a^2 = a^2 + a + 1 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \tag{6.42f}$$

$$a^6 = a^3 + a^2 + a = a^2 + 1 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \tag{6.42g}$$

$$a^7 = a^3 + a = 1 = a^0 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{6.42h}$$

Thus is it shown that a is a primitive root of the prime polynomial, as its successive powers generate all seven possible different non-zero 3-tuples. The concatenation of these vectors,

a^0 から a^7 のベクトルは要素全てが 0 のもの以外の全通りのベクトルになるので a は原始多項式の原始累乗根と言えます。これらのベクトルを結合すると

$$\begin{array}{c} a^0=1 \quad a \quad a^2 \quad a^3 \quad a^4 \quad a^5 \quad a^6 \\ \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}, \end{array}$$

is merely matrix M rearranged. Using this new arrangement of the columns, compare the syndrome computed at the receiving end with the successive powers of a (which can be either stored or generated as needed). When there is a match between the syndrome and the column, we have that column as the corresponding power of a that matches. (A slightly different decoding matrix would result if the other prime cubic, $x^3 + x^2 + 1$, were used.)

これは単なる再配列された行列 M であることがわかります。その新しい列の配列において a の相次ぐ次元（それらは必要に応じて発生、保持される）を用いて終了を受け取る時のシンドロームを比較します。シンドロームと列が対応している場合、a の次数に対応している行を持ちます。（もし原始 3 次多項式 $x^3 + x^2 + 1$ を使用した場合、復元した行列は微妙に違うが実は等しいものなります。）

Check Bits n	Syndromes 2^n	Non-zero Syndromes $2^n - 1$	Primitive Polynomials (culled from $\sum_{i=0}^{n} c_i x^i$)	Code
1	2	1		
2	4	3	$x^2 + x + 1$	Hamming (3,1), majority
3	8	7	$x^3 + x + 1$, $x^3 + x^2 + 1$	Hamming (7,4)
4	16	15	$x^4 + x + 1$, $x^4 + x^3 + 1$	Hamming (15,11)

表 6.23: Check Bits and Prime Polynomials

How does one encode? Since the syndrome $\vec{0} = 0, 0, \ldots, 0$ should mean no error, we require that the sent polynomial be exactly divisible by the modulus polynomial. Therefore the message is put into the positions corresponding to a^6, a^5, a^4, a^3, temporarily putting 0s in the other three positions, corresponding to a^2, a, $a^0 = 1$. After this polynomial is divided by the modulus polynomial, the remainder is put into (added to or subtracted from) the last positions of the written polynomial, so the result will be exactly divisible by the modulus polynomial.

どのように符号化するのでしょうか？エラーがないようにシンドローム $\vec{0} = 0, 0, \ldots, 0$ を得たいので、送られる多項式はモジュール多項式により完全に分割される必要があります。だから a^6, a^5, a^4, a^3 にメッセージを、一時的に a^2, a, $a^0 = 1$ に 0 を設定します。モジュール多項式によって多項式が分割された後に、余りは記述された多項式の最後の位置に置かれます（加えられます）。だから、結果はモジュール多項式によって完全に分割できます。

This, then, is the encoding process: supply the last three positions by using the remainder upon division by the modulus polynomial; thus is the whole polynomial congruent to zero modulo the prime polynomial.

そのようにして、これは符号化の過程となります：モジュール多項式によって分割された余りを使用して最後の 3 位置を与えます。そのようにして全ての多項式はゼロ法原始多項式に一致します。

To decode at the receiving end, simply divide the received polynomial by the modulus polynomial. A non-zero syndrome (remainder) that results when it is expressed as a power of a matches the column where the error occurred.

終了の受信時に複合化するために受け取られる多項式をモジュール多項式によって簡単に分割します。a の次元として結果が書かれた場合の 0 ではないシンドローム（余り）はエラーが生じた列を示します。

6.11.3 Example of Coding; 符号化の例 (3^{rd}-degree Polynomial)

Given the code above, let's encode the message

メッセージを符号化してみましょう。

$$1\ 0\ 0\ 1\ \_\ \_\ \_$$

which corresponds to the polynomial

このメッセージは以下の多項式に符号化されます。

$$a^6 + a^3 = (a^3 + 1)a^3.$$

To encode, divide this polynomial by the chosen modulus polynomial $a^3 + a + 1$ to get the remainder:

余りを得るためにモジュール多項式 $a^3 + a + 1$ を使用して多項式を分割します。

$$\begin{array}{rr|l}
 & & 1\ 0\ 1\ 0 \\
\hline
1\ 0\ 1\ 1 & 1\ 0\ 0\ 1\ \underline{0}\ \underline{0}\ \underline{0} & \\
 & 1\ 0\ 1\ 1 & \\
\hline
 & 0\ 1\ 0\ 0 & \\
 & 0\ 0\ 0\ 0 & \\
\hline
 & 1\ 0\ 0\ 0 & \\
 & 1\ 0\ 1\ 1 & \\
\hline
 & 0\ 1\ 1\ 0 & \\
 & 0\ 0\ 0\ 0 & \\
\hline
 & 1\ 1\ 0 & = \text{remainder}
\end{array}$$

That is,

$$\overbrace{(a^3 + a + 1)}^{\text{divisor; 除数}}\ \overbrace{(a^3 + a)}^{\text{quotient; 商}} = \overbrace{(a^3 + 1)a^3}^{\text{dividend; 被除数}} + \overbrace{(a^2 + a)}^{\text{remainder; 余り}}$$

Therefore the encoded message is

従って、符号化されたメッセージは

$$(1\ 0\ 0\ 1\ 1\ 1\ 0)$$

As verification, confirm that it divides evenly by the modulus polynomial:

確認として、それが率多項式によって均一に分かれることを確認してください:

$$a^6 + a^3 + a^2 + a = (a^3 + a + 1)(a^3 + a).\ \surd$$

Suppose there is an error at the underlined place (which happens to be the a^3 place):

もし下線が引かれた位置に（a^3 の位置）にエラーがあった場合は、

$$(1\ 0\ 0\ \underline{0}\ 1\ 1\ 0)$$

Divide the received message by the same modulus polynomial:

同じモジュール多項式を使用して受け取られたメッセージを分割してみましょう。

$$
\begin{array}{rr|l}
 & & 1\ 0\ 1\ 1 \\
\hline
1\ 0\ 1\ 1 & & 1\ 0\ 0\ 0\ 1\ 1\ 0 \\
 & & 1\ 0\ 1\ 1 \\
\hline
 & & 0\ 1\ 1\ 1 \\
 & & 0\ 0\ 0\ 0 \\
\hline
 & & 1\ 1\ 1\ 1 \\
 & & 1\ 0\ 1\ 1 \\
\hline
 & & 1\ 0\ 0\ 0 \\
 & & 1\ 0\ 1\ 1 \\
\hline
 & & 0\ 1\ 1 \quad = \text{remainder}
\end{array}
$$

But this remainder is

しかしこの余りは

$$
\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = a^3,
$$

which is the fourth position from the left (which is where the error occurred). Toggle this position to get the corrected message, and drop the last three places (LSBs) to recover the original message.

これは左から 4 番目の位置（エラーが起こった位置）です。正しいメッセージにするためにその位置に 1 を加えて、最後の 3 位置を破棄します。

6.11.4 Example of Coding; 符号化の例 (4^{th}-degree Polynomial)

Encode "1234" $= \overbrace{1024}^{2^{10}} + \overbrace{128}^{2^7} + \overbrace{64}^{2^6} + \overbrace{16}^{2^4} + \overbrace{2}^{2^1}$. Shift over 4 bits to make space for check bits. There are two 4^{th}-degree prime polynomials with primitive roots, $x^4 + x + 1$ and $x^4 + x^3 + 1$. For this example, use the first. ($x^4 + x^3 + x^2 + x + 1$ is also prime, but its roots are not primitive since they don't generate the other roots.) Let a be a root of $x^4 + x + 1 = 0$, so $a^4 + a + 1 = 0 \Leftrightarrow a^4 = a + 1$. Generate the check matrix:

「1234」$(= \overbrace{1024}^{2^{10}} + \overbrace{128}^{2^7} + \overbrace{64}^{2^6} + \overbrace{16}^{2^4} + \overbrace{2}^{2^1})$ を符号化する。チェックビット用のスペースを確保するために 4 ビットを転換する。原始根を持つ 4 次の原始多項式 x^4+x+1 と x^4+x^3+1 がある。例として、1 つ目を使う。($x^4+x^3+x^2+x+1$ も原始多項式だが、これらは他の累乗根を作り出さないので、その累乗根は原始根ではない)　a を $x^4+x+1=0$ のルートとするので $a^4+a+1=0 \Leftrightarrow a^4=a+1$。チェック行列を作る：

$$
\begin{array}{c}
a^0=1 \quad a^1=a \quad a^2 \quad a^3 \quad a^4 \quad a^5 \quad a^6 \quad a^7 \quad a^8 \quad a^9 \quad a^{10} \quad a^{11} \quad a^{12} \quad a^{13} \quad a^{14} \\
\begin{pmatrix}
1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1
\end{pmatrix}
\end{array}
$$

Divide by modulus polynomial and add the remainder.

係数で割り、あまりを足す。

$$
\begin{array}{rl}
 & 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1 \\
\hline
1\ 0\ 0\ 1\ 1 & \big|\ 1\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0 \\
 & \ \ 1\ 0\ 0\ 1\ 1 \\
 & \ \ \overline{\ 0\ 0\ 0\ 0\ 0\ } \\
 & \ \ \ \ 0\ 0\ 0\ 0\ 0 \\
 & \ \ \ \ \overline{\ 0\ 0\ 0\ 0\ 0\ } \\
 & \ \ \ \ \ \ 0\ 0\ 0\ 0\ 1 \\
 & \ \ \ \ \ \ \overline{\ 0\ 0\ 0\ 0\ 0\ } \\
 & \ \ \ \ \ \ \ \ 0\ 0\ 0\ 1\ 0 \\
 & \ \ \ \ \ \ \ \ \overline{\ 0\ 0\ 0\ 0\ 0\ } \\
 & \ \ \ \ \ \ \ \ \ \ 0\ 0\ 1\ 0\ 0 \\
 & \ \ \ \ \ \ \ \ \ \ \overline{\ 0\ 0\ 0\ 0\ 0\ } \\
 & \ \ \ \ \ \ \ \ \ \ \ \ 0\ 1\ 0\ 0\ 1 \\
 & \ \ \ \ \ \ \ \ \ \ \ \ \overline{\ 0\ 0\ 0\ 0\ 0\ } \\
 & \ \ \ \ \ \ \ \ \ \ \ \ \ \ 1\ 0\ 0\ 1\ 0 \\
 & \ \ \ \ \ \ \ \ \ \ \ \ \ \ \overline{\ 1\ 0\ 0\ 1\ 1\ } \\
 & \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ 0\ 0\ 0\ 1\ 0 \\
 & \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \overline{\ 0\ 0\ 0\ 0\ 0\ } \\
 & \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ 0\ 0\ 1\ 0\ 0 \\
 & \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \overline{\ 0\ 0\ 0\ 0\ 0\ } \\
 & \ 0\ 1\ 0\ 0\ 0 \\
 & \overline{\ 0\ 0\ 0\ 0\ 0\ } \\
 & \ 1\ 0\ 0\ 0\ 0 \\
 & \overline{\ 1\ 0\ 0\ 1\ 1\ } \\
 & \ 0\ 0\ 1\ 1
\end{array}
$$

That is,

$$
\overbrace{(a^4+a+1)}^{\text{divisor; 除数(じょすう)}}\overbrace{(a^{10}+a^4+1)}^{\text{quotient; 商(しょう)}} = \overbrace{(a^{10}+a^7+a^6+a^4+a)a^4}^{\text{dividend; 被除数(ひじょすう)}} + \overbrace{(a+1)}^{\text{remainder; 余(あま)り}}
$$

Assume an error in a middle bit. Redivide by the same primitive polynomial.

中のビットにエラーがあると仮定する、もう一度同じ原始多項式で割る。

$$
\begin{array}{rl}
 & 1\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0 \\
\hline
1\ 0\ 0\ 1\ 1 & \big|\ 1\ 0\ 0\ 1\ 1\ 0\ 1\ \underline{1}\ 0\ 1\ 0\ 0\ 0\ 1\ 1 \\
 & \ \ 1\ 0\ 0\ 1\ 1 \\
 & \ \ \overline{0\ 0\ 0\ 0\ 0} \\
 & \ \ \ \ 0\ 0\ 0\ 0\ 0 \\
 & \ \ \ \ \overline{0\ 0\ 0\ 0\ 0\ 1} \\
 & \ \ \ \ \ \ 0\ 0\ 0\ 0\ 1 \\
 & \ \ \ \ \ \ \overline{0\ 0\ 0\ 0\ 0\ 0} \\
 & \ \ \ \ \ \ \ \ 0\ 0\ 0\ 1\ 1 \\
 & \ \ \ \ \ \ \ \ \overline{0\ 0\ 0\ 0\ 0} \\
 & \ \ \ \ \ \ \ \ \ \ 0\ 0\ 1\ 1\ 0 \\
 & \ \ \ \ \ \ \ \ \ \ \overline{0\ 0\ 0\ 0\ 0} \\
 & \ \ \ \ \ \ \ \ \ \ \ \ 0\ 1\ 1\ 0\ 1 \\
 & \ \ \ \ \ \ \ \ \ \ \ \ \overline{0\ 0\ 0\ 0\ 0} \\
 & \ \ \ \ \ \ \ \ \ \ \ \ \ \ 1\ 1\ 0\ 1\ 0 \\
 & \ \ \ \ \ \ \ \ \ \ \ \ \ \ \overline{1\ 0\ 0\ 1\ 1} \\
 & \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ 1\ 0\ 0\ 1\ 0 \\
 & \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \overline{1\ 0\ 0\ 1\ 1} \\
 & \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ 0\ 0\ 0\ 1\ 0 \\
 & \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \overline{0\ 0\ 0\ 0\ 0} \\
 & \ 0\ 0\ 1\ 0\ 1 \\
 & \overline{0\ 0\ 0\ 0\ 0} \\
 & \ 0\ 1\ 0\ 1\ 1 \\
 & \overline{0\ 0\ 0\ 0\ 0} \\
 & \ 1\ 0\ 1\ 1
\end{array}
$$

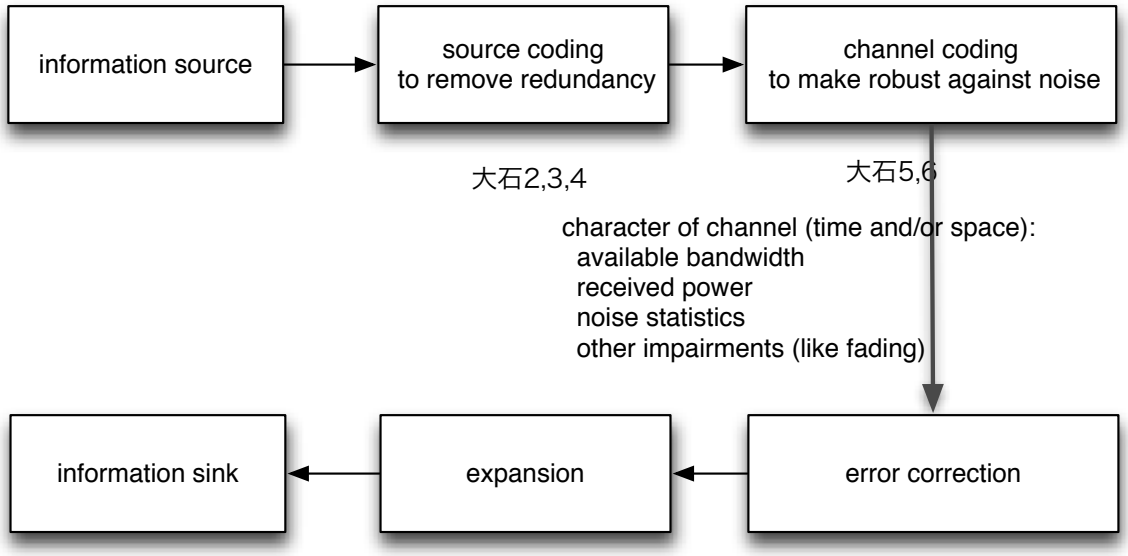
The location of this remainder in the parity check matrix corresponds to the location of the error.

パリティチェック行列の中のこのあまりの位置とエラーの位置は一致する。

第7章　Appendix: Overview

Huffman: successively combine least-probable nodes
Markov process: state dependency
RLE (run-length encoding): useful with sparse data or prediction
ZL (Ziv-Lempel): construct dynamic dictionary
arithmetic: represent as interval in real number continuum

parity checks (checksum, ISBN)
block coding
  rectangular
  triangular
Hamming
polynomial



Choose coding scheme based on characteristics of desired performance,
like data rate and error performance.

目 次

第 1 章 1: Introduction; はじめに **2**

1.1 Sets and Probability; 集合と確率 . . . 2

1.1.1 Membership . . . 2

1.1.1.1 Empty Set; 空集合 . . . 2

1.1.1.2 Singleton: Set with only one element; 単集合: ただ一つの集合 . . . 2

1.1.1.3 Venn Diagrams; ベン図 . . . 2

1.1.2 Structure . . . 2

1.1.2.1 Sets; 集合 . . . 2

1.1.2.2 Elements; 要素、元 . . . 5

1.1.2.3 Subsets; 部分集合: $A \supset B \equiv B \subset A$ . . . 5

1.1.2.4 Proper Subsets; 真部分集合: $B \subsetneq A$ . . . 5

1.1.2.5 Infinite Sets; 無限集合 . . . 5

1.1.3 Operations . . . 6

1.1.3.1 Set Subtraction; 差集合: $A - B$ (non-commutative; 可換でない) . . . 6

1.1.3.2 Set Complement; 補集合: $\overline{A} = \Omega - A$ . . . 6

1.1.3.3 Set Union; 和集合: $A \overset{\text{“cup”}}{\cup} B$ (commutative; 可換) . . . 7

1.1.3.4 Set Intersection; 積集合: $A \overset{\text{“cap”}}{\cap} B$ (commutative; 可換) . . . 7

1.1.3.5 DeMorgan's Laws; ド・モルガンの法則 . . . 7

1.1.4 Cardinality; 要素数 . . . 8

1.1.4.1 Power Set (Set of Subsets); 累乗集合 (巾集合、冪集合)　部分集合の集合 . . . 8

1.2 Trials; 試行、試行列と事象 . . . 9

1.2.1 Stochastic Process; 確率過程 . . . 10

1.2.2 Event; 事象 . . . 10

1.3 Probability and Probability Space; 確率と確率空間 . . . 10

1.3.1 Intervals; 間隔 . . . 10

1.3.2 Probability Space; 確率空間 . . . 11

1.4 Random Variables: Distribution; 確率変数: 分布 . . . 12

1.4.1 Random Variables; 確率変数 . . . 12

1.4.2 Probability Distribution; 確率分布 . . . 13

1.4.2.1 Example: coin; 例: コイン . . . 13

1.4.2.2 Example: fair die; 例: さいころ . . . 13

1.4.2.3 Example of random variable: sum of two dice throws; 確率変数の演算の例 . . . 14

1.5 Mean (Average) & Variance; 平均と分散 . . . 15

1.5.1 Mean; 平均, Expectation; 期待値, Expected value; X の確率平均 15
1.5.2 Variance; 分散 16
1.5.2.1 Standard Deviation; 標準偏差 17
1.5.2.2 Combining Variances; 分散の組み合わせ 19
1.5.2.3 Covariance; 共分散 and Correlation; 相関 22
1.6 Conditional Probability; 条件付確率 23
1.7 Independence of Random Variables; 確率変数の独立性 27
1.7.1 Independence; 独立性 27
1.7.2 Memorylessness; 無記憶性 28
1.7.3 Stationarity; 定常 28
1.8 Law of Large Numbers; 大数の法則 30
1.8.1 Chebyshev Inequality; チェビシェフの不等式 30
1.8.2 Average of Sequence; 確率変数の列の平均 31
1.8.3 (Jakob) Bernoulli Trials; （ヤコブ）ベルヌーイの法則 31
1.8.4 Law of Large Numbers; 大数の法則 32
1.8.5 Weak Law of Large Numbers; 大数の弱法則 34
1.9 Markov Process; マルコフ過程 35

第 2 章 2: Source Coding; 情報源符号化 37
2.1 Information Source Model and Encoding; 情報源符号 37
2.1.1 Model of Information Source and Source Encoding Problem; 情報源のモデルと情報源符号化問題 37
2.2 (Information) Source Coding; 情報源符号 38
2.2.1 Bits (binary digits) and Binary States; ビット と 二値法 38
2.2.2 Binary Information Source Coding; 2 元情報源符号化 38
2.2.3 Code, Coding, Code Word; 符号, 符号化, 符号語 39
2.2.4 Code Length; 符号長 39
2.2.5 Mean Code Length; 平均符号長 39
2.2.6 Decoding; 復号 39
2.2.7 Uniquely Decodability; 一意的に復号可能 40
2.2.8 (Examples of) Various Coding Techniques; 各種の符号の例 40
2.2.8.1 Variable-length Code; 可変長符号 40
2.2.8.2 Comma Code; コンマ符号 40
2.2.8.3 Instantaneous Code; 瞬時符号 40
2.2.8.4 Singular Code; 特異符号 40
2.2.9 (Information) Source Coding; 情報源符号 41
2.3 Instantaneous Codes; 瞬時符号 42
2.3.1 Instantaneous Codes (Expressed by Code Trees); 瞬時符号 (符号の木による表現) 42
2.3.1.1 Tree; 木 42
2.3.1.2 Node, Vertex (plural: nodes, vertices); 節点 42
2.3.1.3 Edge, Branch; 枝 42

2.3.1.4 Path; パス . . . 42
2.3.1.5 Connected Graph; 連結グラフ . . . 43
2.3.1.6 Loop; 閉路 . . . 43
2.3.2 Representation as Coding Tree; 符号の木による表現 . . . 43
2.4 Kraft Inequality; クラフトの不等式; . . . 45
2.5 Code Space; 符号空間 . . . 47
2.6 Uniquely Decodable; 一意的に復号可能な符号 . . . 48
2.6.1 McMillan Inequality; マクミランの不等式 . . . 50
2.7 Mean Code Length and Entropy; 平均符号長とエントロピー . . . 55
2.7.1 Logarithms; 対数 . . . 55
2.7.2 Remarks; 注意 . . . 60
2.8 Source Coding Theorem; 情報源符号化定理 . . . 61
2.9 Practice Exercizes; 演習 . . . 64
2.9.1 Questions . . . 64
2.9.2 Answers; 解答 . . . 65

第 3 章 3: Various Kinds of Source Codings; いろいろな情報源符号 68
3.1 Huffman Codes; ハフマン符号, 1962 . . . 68
3.2 Compact Code; コンパクト符号 . . . 69
3.3 Markov Information Source (Process); マルコフ情報源 . . . 69
3.4 Transition Diagram; 遷移図 . . . 70
3.4.1 Trellis Diagram: Markov Information Source (Process); マルコフ情報源 (2) 71
3.5 Ergodic Markov Process; エルゴード的 マルコフ情報源 . . . 72
3.5.1 Formal Consideration . . . 78
3.6 Source Coding of (First-Order) Markov Process (Information Source); マルコフ情報源に対する情報源符号化 . . . 80
3.7 Run-length encoding (RLE); ランレングス符号 . . . 80
3.8 Source Coding and Algorithm; 情報源符号化とアルゴリズム . . . 82
3.9 Ziv-Lempel Coding; ZL 符号 . . . 82
3.10 Arithmetic Coding . . . 83
3.11 Gray Codes; グレイ符合 . . . 85
3.11.1 Baguenaudier, Patience Puzzle, Chinese Rings, Devil's Needle . . . 85

第 4 章 4: Information and Entropy; 情報量とエントロピー 87
4.1 Information quantity; 情報量 . . . 87
4.1.1 Logarithms; 対数 . . . 87
4.2 Equation which follows information quantity; 情報量の従う方程式 . . . 88
4.3 Complexity and program length; 系列の複雑さとプログラムの長さ . . . 88
4.3.1 Kolmogorov Complexity; コルモゴロフ・コンプレキシティ . . . 88
4.4 Characteristics of Entropy; エントロピーの関数としての性質 . . . 89
4.4.1 Equiprobable Distribution; 等率発生分布 . . . 91
4.4.1.1 Trivial Example: Computer Memory . . . 91
4.4.1.2 Example: Fair Coin; 例： 公平な硬貨 . . . 91
4.4.2 Nonequiprobable Two-State Distribution: Unfair Coin; 不公平な硬貨 . . . 92
4.4.3 Diversion: Zero Probabilities; 偏向： 確率ゼロ . . . 92

4.4.4 Properties of Entropy; エントロピーの特性 94
4.4.5 Example: 4-state Random Variable; 例： 四つの状態のある任意の変数 . . 95
4.5 Conditional Entropy; 条件付きエントロピー 96
4.5.1 Joint Entropy; 結合 エントロピー 96
4.5.1.1 Joint Distributions; 結合分布 96
4.5.1.2 Joint Entropy; 結合エントロピー 98
4.5.1.3 Example: Joint Distribution; 例：結合分布 98
4.5.2 Conditional Entropy and the Chain Rule; 条件付きエントロピーと結合法則 100
4.5.2.1 Example: Conditional Entropy; 例： 条件付きエントロピー . . . 101
4.5.2.2 Conditioning reduces entropy on the average; 平均的な条件付き減少エントロピー 102
4.5.2.3 Example: Joint Distribution and Conditional (Relative) Entropy; 例： 結合分配と条件付き（相対的）エントロピー 103
4.6 Mutual Information; 相互情報量 104
4.6.1 Entropy and Mutual Information; エントロピーと相互情報量 105
4.6.2 Example: Two Coins; 例： ２つのコイン 105

第 5 章 5: Channel Coding; 通信路符号化の限界 109
5.1 Channels; 通信路 109
5.2 Channel Capacity; 通信路容量 110
5.3 Noiseless binary channel; ノイズのない二値通信路 111
5.4 Noisy channel with nonoverlapping outputs; 重複した出力がなくノイズのある通信路112
5.4.1 Noisy typewriter; ノイズのあるタイプライター 113
5.5 **B**inary **s**ymmetric **c**hannel (BSC); ２元対称通信路 115
5.6 Symmetric Channel and Maximum Liklihood Decoding; 対称通信路 117
5.6.1 Shannon-Hartley Capacity Theorem 119
5.6.1.1 Large S/N 120
5.6.1.2 Small S/N 120
5.6.2 Z-channel 124
5.7 (Average Error Rate) 127
5.8 (Channel Coding) 127
5.9 (Hamming Distance) 127
5.10 (Channel Coding Theorem) 127
5.11 (Lemma for Proof of Channel Coding Theorem) 127

第 6 章 6: Limits of Coding Theory; 符号理論 128
6.1 Parity Check; パリティチェック 128
6.1.1 Binary functions; ２進関数 128
6.1.2 Magic cards; マジックカード 131
6.1.3 Error detection; エラーの検知 131
6.2 Modulo Operation; 剰余演算 132
6.3 Parity Check Codes; パリティチェック 符号 132
6.4 Rectangular Codes; 長方形符合 133
6.5 Triangular Codes; 三角形符合 133
6.5.1 Higher Dimension Parity Checks; 高次元パリティチェック 134

6.6 Hamming Codes; ハミング符号 134
6.7 Generator Matrix and Parity Check Matrix; 生成 (ジェネレーター) 行列とパリティチェック行列 136
6.7.1 Example: Hamming (7,4) Encoding and Decoding 138
6.8 (Group-Organized Codes; 組織符号) 140
6.9 Minimum Distance of a Code; 符号の最小距離 140
6.10 Vectors and Polynomials; ベクトルと多項式 142
6.11 Prime Polynomials and Extensions; 原始多項式と拡大体 145
6.11.1 Prime Polynomials; 原始多項式 145
6.11.2 Primitive Roots; 原始累乗根 148
6.11.3 Example of Coding; 符号化の例 (3^{rd}-degree Polynomial) 154
6.11.4 Example of Coding; 符号化の例 (4^{th}-degree Polynomial) 155

第 7 章 Appendix: Overview **158**

図 目 次

1 Part of the Mandelbrot Set, which has low Kolmogorov complexity; 低いコルモゴロフ・コンプレキシティをもったマンデルブロー集合の一部 1

1.1 Communication Model: source→sink (receiver); コミュニケーションモデル：情報源 → 受信者. Source encoding minimizes representation (compression), and channel encoding increases robustness (enabling error detection/correction). 情報源符号化は情報の表現を最小にする (圧縮)、通信路符号化は堅牢性を高める (エラー検知/修正を可能にする)。 3
1.2 History and Organization of Information Theory; 情報理論の歴史と体系 3
1.3 Venn Diagram: Probability Space; 確率 空間. $A \cup B = (A - B) + (A \cap B) + (B - A)$. 4
1.4 Venn Diagram: Colors— {red, $\overset{\text{primaries}}{\text{green}}$, blue}, Cyan, Magenta, Yellow, White; ベン図: 色— { 赤, 緑, 青 }、青緑(シアン)、紫(マゼンタ)、黄色(イエロー)、白(ウイト) 4
1.5 DeMorgan's Law: the intersection of the two circular regions is the complement of the union of the complementary areas (vertically and horizontally striped regions); ド・モルガンの法則：２つの円の交差領域は、補集合領域の結合部分を補う部分である (縦線と横線で示された領域)— $AB = A \cap B = \overline{\overline{A} \cup \overline{B}}$ 8
1.6 Conditional Probability; 条件付き確率 12
1.7 Distribution of throw of a single die; 一つのさいころの出目の分布 14
1.8 Distribution of sum of two dice; 二つのさいころの出目の和の分布 15
1.9 Expectation of a single die throw!; 一つのさいころの期待値のイメージ図！ 16
1.10 Probability density function of normal (Gaussian) distribution: Two distributions with unequal variances. The area under both curves is the same. 正規 (ガウス) 分布の確率密度関数：異なる分散による 2 つの分布。両方の曲線の下の範囲は同じです。($\overset{\text{unity}}{1} = 100\%$.) A normal or Gaussian distribution has the form $e^{-(t-t_0)^2}$. 16
1.11 RMS: **r**oot **m**ean **s**quare: original unit sinusoid, its square, and its cummulative RMS; 二乗平均平方根：初期単位の正弦曲線、その平方とその RMS 18
1.12 Temperature Conversion from Fahrenheit to Centigrade (Celsius); 華氏から摂氏への温度変化 20
1.13 Independent vs. Correlated Datasets 21
1.14 Height vs. Weight for a Population Sample; あるサンプル個体群における身長 対 体重 22

1.15 Correlation: Several sets of (x, y) points, with the correlation coefficient of x and y for each set. Note that the correlation reflects the noisiness and direction of a linear relationship (top row), but not the slope of that relationship (middle), nor many aspects of nonlinear relationships (bottom). (Note: The figure in the center has a slope of 0, but in that case the correlation coefficient is undefined because the variance of Y is zero.); 相関: (x, y) で示される点が数セットあり、それぞれのセットにおける x と y の相関係数がある。上段における相関はノイズと線形関係の傾斜を反映しているが、中段における相関においては線形関係の傾きは反映されていない。下段の非線形関係においても同様に反映されない。この点は注意すべきである。(注意: 中央の図の傾きは 0 だが、その場合 Y の分散は 0 になるので相関係数は定義されない。) 23
1.16 Visualization of Chebyshev Inequality; チェビシェフの不等式の図 30
1.17 (1^{st}-order) Mechanical Moment Arms Create Torque; (一次) 機械的モーメントアーム (支点から力の作用点に降ろした垂線の距離) によるトルク 32
1.18 First-order Markov Process Model; 単純マルコフ過程(かてい)のモデル. Example: Each day's weather depends in part on the previous day's. 例: 前日の天気に応じた各日の天気。 36

2.1 Completely Balanced Binary Tree 45
2.2 Complete Binary Subtree; 完全 2 分木の部分木 45
2.3 Partitioning Rectangle; 長方形分割 47
2.4 Variable Length Phone Number Prefixes 48
2.5 $(4n)^{\frac{1}{n}}$, $(2n)^{\frac{1}{n}}$, $n^{\frac{1}{n}}$, $\frac{n}{2}^{\frac{1}{n}}$ 52
2.6 Generic Functions; 関数 54
2.7 $e\left(=\lim_{n\to\infty}(1+\frac{1}{n})^n\right)=\lim_{n\to 0}(1+n)^{\frac{1}{n}}$ 56
2.8 $\tilde{\lg}_2^{1.4}(\frac{1}{t})$, $\frac{1}{t}$, $\tilde{\log}_{10}^{.43}(\frac{1}{t})$: $\log_e(x)\equiv\ln(x)\triangleq\int_1^x(\frac{1}{t})dt$; $\log_{\{2,e,10\}}\frac{1}{x}=\log x^{-1}=-\log x$; $\int_1^{1/x}(\frac{1}{t})dt=-\int_{1/x}^1(\frac{1}{t})dt=-\int_1^x(\frac{1}{t})dt$; $\log(\frac{1}{2})=-\log(2)$ 57
2.9 $\log_e=\ln$ (dashed) and $x-1$ (solid) 58
2.10 Code Trees; 符合木 65
2.11 Code Containment (Taxonomy); 符合の包含図 (分類) 66

3.1 State Diagram: Ergodic (Hidden) Markov Chain/Model; マルコフ情報源 (repeated from §1.9) 70
3.2 Trellis Diagram of 3-State Markov Process: $p'(a)=p(a)p(a|a)+p(b)p(a|b)+p(c)p(a|c)$; $p'(b)=p(a)p(b|a)+p(b)p(b|b)+p(c)p(b|c)$; $p'(c)=p(a)p(c|a)+p(b)p(c|b)+p(c)p(c|c)$ 71
3.3 3-Dimensional Probability Transition; 状態遷移の 3 次元表現 ($p_a+p_b+p_c=1$)—$(p'_a,p'_b,p'_c)=(p_a,p_b,p_c)T$ 72
3.4 Reducible Graph; "Leaky Flip-FLop" 73
3.5 Arithmetic Coding Example; 算術符合の例 84
3.6 Gray Code Sequence 86

4.1 $\{\frac{\log_2(e)}{t},\frac{1}{t},\frac{\log_{10}(e)}{t}\}$. $\frac{d}{dx}\ln(x)=\frac{1}{x}$; $\ln(x)=\int_1^x(\frac{1}{t})dt\Rightarrow\ln(\frac{1}{2})=-\ln(2)$; $\frac{d}{dx}\log_{2,e,10}(x)=\frac{1}{x}\log_{2,e,10}(e)$ 87
4.2 Part of the Mandelbrot Set, which has low algorithmic (Kolmogorov) complexity; 低いコルモゴロフ・コンプレキシティをもったマンデルブロー集合の一部 90

4.3 [p, $\log(1/p)$, and] $p\log(1/p) = -p\log p$. Note that $\lim_{p\to 0}(-p\log p) = 0$ as the linear function overpowers the logarithmic. 93
4.4 Dyadic entropy $H_2(p) = H(p, 1-p)$ 94
4.5 Flow Chart for Determining Value of Random 4-state Variable; 任意の４変数の値を決めるためのフローチャート 98
4.6 Venn Diagram: Joint Entropy and Mutual Information; ベン図: 結合エントロピーと相互情報量 99
4.7 $H(X) = H(\frac{1}{8}, \frac{7}{8}) \approx .5$ bits; $H(X) = H(\frac{1}{7}, \frac{6}{7}) \approx .6$ bits; $H(Y) = H(\frac{1}{4}, \frac{3}{4}) \approx .8$ bits; $H(X, Y) = H(\frac{3}{4}, \frac{1}{8}, \frac{1}{8}) \approx 1.1$ bits; 103
4.8 Information Flow; 情報の流れ: Irrelevance; 無関係 (ノイズや遮蔽物のため), Equivocation; 曖昧な量 (圧縮を失う) 106
4.9 Example of Conditional Entropy 107
4.10 Information in Coin-tossing Experimental Channel 108

5.1 Channel Model: $x \in$ input alphabet $\mathcal{X}$, $y \in$ output alphabet $\mathcal{Y}$, $p(y|x)$ probability mass function; 通信路のモデル:$x \in$ 入力される確率変数 $\mathcal{X}$, $y \in$ 出力される確率変数 $\mathcal{Y}$, $p(y|x)$ 確率関数 109
5.2 Noiseless Binary Channel; ノイズのない二値通信路 111
5.3 Always Lying Channel; 「常に虚偽の」通信路 111
5.4 Information in Noiseless and "Always Lying" Binary Channels; ノイズのない「常に虚偽」の二値通信路の情報 112
5.5 Noisy Channel with Nonoverlapping Outputs; 重複した出力がなくノイズのある通信路 112
5.6 Information in Noisy Channel with Nonoverlapping Outputs; 重複した出力がなくノイズのある通信路の情報 114
5.7 Noisy Typewriter; ノイズのあるタイプライター（出力が重複するものとしないもの） 115
5.8 Information in Noisy Typewriter with Overlapping Outputs; 重複した出力があるノイズのあるタイプライターの情報 116
5.9 Information in Noisy Typewriter with Non-Overlapping Outputs; 重複した出力がないノイズのあるタイプライターの情報: Equivocation eliminated by restricting input symbols; 入力記号の制限により曖昧な表現が除かれる 116
5.10 **B**inary **S**ymmetric **C**hannel (BSC) with crossover probability p; 2 要素対称通信路 117
5.11 AWGN: power spectrum density of **A**dditive **W**hite **G**aussian **N**oise 121
5.12 Shannon-Hartley Limit: normalized energy/bit vs. bandwidth efficiency (eqn. (5.19)); シャノン・ハートレー制限：正規化エネルギー/ビット対帯域幅効率 (eqn. (5.19)) . 123
5.13 Quantized Gray Scale; 量子化グレースケール 123
5.14 Z-Channel 124
5.15 Z-Channel Capacity; throughput vs. error probability. Even when $p = \frac{1}{2}$, some channel capacity can be salvaged. However, when $p = 1$, output is constant ($y_1 = 0$), so capacity vanishes completely. This plot shows the capacity, which does what one would expect: full capacity when error probability is zero, dropping to zero capacity when error probablility is unity. 125

5.16 Z-Channel Optimization: optimal fraction of use of noisy side vs. error probability. This plot shows the optimal α to use for given p, the chance of cross-symbol error in a z-channel. Note that for zero error probability, highest throughput obtained when 50% use of each side, as for usual simple noiseless channel, but as error probability increases, use of the error-prone side is decreased for optimal throughput. 126

6.1 Symbols for boolean gates; ブーリアンゲートの記号 128
6.2 XOR can be implemented with just NAND gates. This circuit is also deployable as a half-adder, as it calculates the LSB of 1-bit addition. (The MSB is just an AND, or NANDed NAND.); XOR は NAND だけの組合わせで実装できる。これは 1 ビット加算の LSB を計算するのでこの回路は半加算器としても配置可能である。(MSB は AND や NAND された NAND である。) 130
6.3 Magic Cards; マジックカード 131
6.4 Hamming (7,4) Code; ハミング (7,4) 符合 135
6.5 Cartesian and Manhattan Distances; 観念的距離とマンハッタン距離 141
6.6 Fifth Roots of Unity 150

表 目 次

1.1 Classes of Algebraic Solutions 6
1.2 Diagonal ennumeration of rational numbers; 有理数の対角線上での列挙 9
1.3 Greek Letters; ギリシア文字 11
1.4 Sum of throw of two dice; 二つのさいころの出目の和 14
1.5 Easily confused symbols that resemble 'E'ach other 35

2.1 Analog and Digital 38
2.2 Binary (2-state) Name Pairs; 一組の状態を表す言葉の例 39
2.3 Examples of Codes; 情報源符号の例 (例題 1: 符号 C_1–C_6 [大 93, p. 17], 例題 2: 符号 C_7, C_8 [大 93, p. 29]) 41
2.4 Implication; $A \xleftarrow[\text{"is necessary for"}]{\text{"suffices for"}} B$. For example, if snow implies cold, then cold is necessary for snow, and snow is sufficient for cold. 例えば、「雪」⇒「冷たい」という命題が与えられた時、「冷たい」は「雪」の必要条件であり、「雪」は「冷たい」の十分条件である。 44
2.5 $X + \overline{Y}$, $Y \Rightarrow X$ (necessity) 44
2.6 $(X \oplus Y, X \neq Y)$: XOR— exclusive or (inequality, symmetric difference), same as CNOT (controlled not, used in quantum computing) 44
2.7 IMPLIES (sufficiency): $X \Rightarrow Y$ $(Y + \overline{X})$ 44
2.8 $X \equiv Y$: EQUIV, XNOR (equality, coincidence, mutual implication, iff [if and only if], necessity and sufficiency) 44
2.9 Dyadic binary functions; 2 変数の 2 進関数の組 44
2.10 Binary Representations and Extensions; 2 進法表現 (しんほうひょうげん) と 拡張 (かくちょう) 49
2.11 Second Extension of C_7; C_7 を用いた 2 次の拡大情報源 S^2 の符号 50
2.12 Generic Function Complexity (Growth); 関数の増大 53
2.13 Logarithmic Bases; 対数の底 57
2.14 Containment Hierarchy of Codes 63
2.15 符合の階層 63

3.1 Algebraic Equation System Comparison 75
3.2 Sample 2-Pass Huffman Coding [大 93, p. 53] 82

4.1 Complexity; 複雑さ 89
4.2 Entropies of Some Common Equiprobable Distributions; 一般的な等率発生分布のエントロピー 92
4.3 Dimensions and Arrays. (Note: "primitives" in modern programming languages are types like `int`, `float`, `enumerated`, ...); 次元と配列 (注：現代プログラミング言語における基本要素は `int`, `float`, `enumerated` といった型である。) 97

4.4 Example of Dependent Variables (cloudiness is hint about rain); 例題 従属変数 (降水確率): rainy ⇒ cloudy and uncloudy ⇒ not rainy: $H(\text{rain}) = H(\frac{1}{4}, \frac{3}{4}) \approx 0.8; H(\text{Rain}|\text{Cloudiness}) = .5 \cdot 0 + .5 \cdot 1 = \frac{1}{2}$ bit; $H(\text{Cloudiness}|\text{Rain}) = \frac{3}{4}H(\frac{1}{3}, \frac{2}{3}) + \frac{1}{4}H(0, 1) \approx \frac{3}{4} \cdot 0.9 \approx 0.7$ bits 97

6.1 ZERO, RESET, INHIBIT, OFF (0) 129
6.2 ONE, SET, ASSERT, ON (1) 129
6.3 AND, conjunction (XY) 129
6.4 NAND ($\overline{XY} \equiv \overline{X} + \overline{Y}$) 129
6.5 $X\overline{Y}$ (set difference $X - Y$) 129
6.6 IMPLIES (sufficiency): $X \Rightarrow Y$, $Y + \overline{X}$ 129
6.7 X 129
6.8 $\overline{X}$ 129
6.9 $\overline{X}Y$ (set difference $Y - X$) 129
6.10 $X + \overline{Y}$, $Y \Rightarrow X$ (necessity) 129
6.11 Y 129
6.12 $\overline{Y}$ 129
6.13 ($X \oplus Y, X \neq Y$): XOR— e**x**clusive or (inequality, symmetric difference), same as CNOT (controlled not, used in quantum computing) 129
6.14 $X \equiv Y$, $X \Leftrightarrow Y$: EQUIV, XNOR (equality, coincidence, mutual implication, iff [if and only if], necessity and sufficiency) 129
6.15 (**i**nclusive) OR, disjunction ($X + Y$) 129
6.16 NOR ($\overline{X + Y} \equiv \overline{X}\,\overline{Y}$) 129
6.17 Dyadic binary functions; ２変数の２進関数の組 129
6.18 Bitwise Operations (and C and Java notation); ビット演算 (と C や Java での表記) 130
6.19 Hamming (7,4) Codeword Dictionary: Dimension 2^4 Entries $\times$ 7 Bits; 次元: $2^4 \times 7$. Any two of the code words differ in at least three positions. Note that not all possible codewords are used, like noisy typewriter with non-overlapping inputs. ハミング (7, 4) の符号語辞書：次元 2^4 の入力 $\times$7 bits；次元 $2^4 \times 7$。符号語の二つは少なくとも三か所異なります。全ての可能な符号語が使用されない点に注意し、重複しない入力を雑音が多いタイプライターに嘘をついてください。 138
6.20 Properties of minimum distance; 最小距離のせいしつ 142
6.21 Prime Polynomials with Primitive Roots 149
6.22 Some Roots of Unity: solutions to $x^n = 1$ 151
6.23 Check Bits and Prime Polynomials 153

関連図書

[Ash65] Robert B. Ash. *Information Theory*. Dover Publications, Inc., 1965. ISBN 0-486-66521-6.

[CT91] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley & Sons, Inc., 1991. ISBN 0-471-06259-6.

[GS93] Larry Gonick and Woollcott Smith. *The Cartoon Guide to Statistics*. HarperCollins, 1993. ISBN 0-06-273102-5.

[Ham80] Richard W. Hamming. *Coding and Information Theory*. Prentice-Hall, 1980. ISBN 0-13-139139-9.

[Mac02] David J. C. MacKay. *Information Theory, Inference & Learning Algorithms*. Number 0. Cambridge University Press, 2002.

[Pie61] John R. Pierce. *Introduction to Information Theory; Symbols, Signals, and Noise*. Dover, 1961. ISBN 0-486-24061-4.

[Rez94] Fazollah M. Reza. *An Introduction to Information Theory*. Dover, 1994. ISBN 0-486-68210-2.

[Skl01] Bernard Sklar. *Digital Communications*. Prentice-Hall, second edition, 2001. ISBN 0-13-084788-7.

[vdL97] Jan C. A. van der Lubbe. *Information Theory*. Cambridge University Press, 1997. ISBN 0-521-46760-8.

[大 93] 進一 大石. **例にもとづく情報理論入門**. 講談社 サイエンティフィク, 1993. ISBN 4-06-153803-9.