情報理論

目次

第 1 章

導入

1.1　通信のモデル化

1.1.1　通信システムのモデル

通信システムは 5 つの段階でモデル化できる．1 つずつ紹介する．

1. 情報源
2. 符号化
3. 通信路
4. 復号
5. 宛先

この 5 段階であり，その中を通報 (message) が流れてゆく．

1.1.2　アナログデータのデジタル通信

アナログデータの例として英文の通信のモデルを考える．以下のような 7 つの段階になる．

1. 英文の情報源
2. 0,1 の列
3. 0 は正パルス，1 は負パルス
4. 通信路
5. パルスの極性判定
6. 0,1 の列
7. 英文

1.2　符号化

1.2.1　2 つの大別

情報源符号化　統計的性質によって通信の効率化を図る．

通信路符号化　誤りに対して高信頼化を図る．

1.2.2　語句の理解

情報源記号　元の情報に対して識別の記号を与えたもの．
符号語　各情報源記号に対して与えられた符号１つ１つのこと．
符号　１セットの情報源に対して与えられた符号語の集合．
符号化アルファベット　符号の用いられる文字の組．例:$\{0,1\}$
q 元符号　q 個の文字の組の符号化アルファベットを用いて作る符号．

1.3　符号化の効率と信頼

1.3.1　符号化に求められること

通信には必ず誤りが生じる．信頼性にかかわる問題である．また誤りを避けるべく冗長な符号[*1]による通信をしてもかまわないがこれでは効率が悪い．こちらは経済性に関わる問題である．ではどのような符号化が求められるのか．

まずは経済性に関わる問題を解決すべく，情報の発生確率[*2]に注目して考える．この場合，そうする回数の多い情報，すなわち発生確率の大きな情報に対してはできる限り短い符号を，その反対に発生確率の小さな方法に対しては長い符号でもいいだろうという発想でクリアできるだろう．

では次に信頼性に関わる問題を解決することを考える．この解決法の発想は簡単で，同じ情報を複数回繰り返して送信することである．同じ情報同士の比較によって誤りを訂正する．これでは通信量が増えるので符号化によってこれを解決する．

1.3.2　新たなモデル

符号化の問題などを考えたうえで新たに通信のモデルを考える．以下のとおりである．

1. 情報源
2. 符号化
   （a）情報源符号化
   （b）通信路符号化
3. 通信路
4. 復号
   （a）情報源復号
   （b）通信路復号
5. 宛先

[*1] 誤りがカバーできる長い符号化とは１ビット誤っても識別できるというイメージだからとんでもなく長くなることが想像できる．
[*2] 情報が送らねばならなくなる出現率のようなものを想像すれば間違いではない．

第 2 章

情報源符号化

先の章では符号化の経済性，信頼性の問題を考え 2 つの符号化を挙げたが以降その符号化自体を問題として扱う．

まず経済性の問題を考える情報源符号化に注目して話を進める．

2.1　情報源符号化の問題設定

2.1.1　評価基準

情報源符号化に対しては以下の 3 つの評価基準がある．

- 1 情報源記号当たりの平均符号長
- 装置の簡単さ
- 遅延

さらには可逆性が大切でここでひずみが生じるかどうかも問題である．

2.1.2　限界

なんでもやって良いわけではないので，結局のところ以下の限界がある．

- 1 情報源当たりの平均符号長の最小値
- 可逆であること

説明するといくら短くしてもその限界はある．そしてその最小値を無視する形でとにかく短い符号化を行うと可逆ではない，つまり元の情報に戻すことができなくなってしまうという事である．

2.1.3　平均符号長の計算

これは期待値の計算と同じ要領で，すべての情報源記号に対し，その情報源記号の情報の発生率とそれに割り当てられた符号の長さの積を計算しそのすべてを足し合わせることで算出できる．

2.2　符号の性質を表す語

符号にはその特徴によって名称がある．性質とともに紹介する．

2.2.1　特異符号

特異符号とは異なる情報源記号に対し同一の符号語が割り当てられているものが存在する符号．特異符号の場合はひずみが生じ可逆ではない．

2.2.2　一意復号可能な符号

情報源記号 1 つ 1 つに可逆な符号を与えたからといって正しく情報を伝えられるとは限らない．符号化された情報源記号を並べ与えた符号のとおり 0,1 の列に直す．これを復号するとき 0,1 の列の区切り方によって複数の復号ができてしまう符号がある．これが一意復号不可能な符号である．これとは逆に 0,1 の列が必ず元の情報源記号の列に戻せる符号を一意復号可能な符号という．

2.2.3　瞬時符号

0,1 の列の中のある符号語をもとの情報源記号に戻すことを想像する．もちろん人がおこなえば見比べるだけなので大したことではないが機械の場合は符号語の先頭から比べて選択肢を絞り情報源記号に行きつく様が想像できるだろう．この時，読みたい符号語は比べ終わったが，その次の符号語の先頭の値を読まなければ決定はできないという状況が存在する．分かりやすく言えば，01 と読み込んだ時点で情報源記号 A の符号語ではあるが，もし次が 1 ならば情報源記号 B の符号 011 となってしまい，まだわからないという状況である．

以上の説明でのすぐさま情報源記号に戻せない符号は瞬時記号ではない符号である．それとは逆にある情報源記号に対応する符号語が現れたらすぐさま復号できる符号を瞬時復号という．

2.2.4　等長符号

読んで字のごとく全符号語が同じ長さの符号語である．計算が楽なのでたくさん出てきて欲しいものである．

2.3　符号化に必要な条件

符号化の性質とその呼び方を理解したうえで符号化に必要な条件を示す．

符号化に必要な条件

1. 一意復号であること，瞬時復号であることが望ましい
2. 1 情報源記号当たりの平均符号長ができるだけ短い
3. 装置化があまり複雑にならない

以降この符号化の条件実現のための内容を述べる．

2.4　瞬時符号の条件

瞬時符号は正直作るとなるとイメージしにくいように思える．まずはその条件について理解したい．

2.4.1　語頭条件

語頭条件とは符号の中のどの符号語も他の符号語の語頭とはならないという瞬時符号であるための必要十分条件である．言われればその通りというくらいの条件である．またこれは符号が瞬時符号化否かを判別するのには使い勝手が良いが，正直瞬時符号の作成には利用できないだろう．

2.4.2　符号の木

図の符号を木で表したものを符号の木という．端的に言うと符号語が符号の木における葉にあればその符号は瞬時符号である．

2.4.3　クラフトの不等式

瞬時符号ある条件は数式によっても与えられている．q 元符号で M この情報源記号を符号化している状況を考える．各情報源に対する符号長を l_m $(1 \leq m \leq M)$ で表すとする．すると以下の関係が成立する．

瞬時符号である

$$\Leftrightarrow \sum_{m=1}^{M} q^{-l_m} \leq 1 \tag{2.1}$$

2.5　平均符号長の限界

ここからは平均符号長に関して述べてゆく．また基本的に 2 元符号しか使わないので以降の式は 2 元符号のときに成立するものしか示さない．

2.5.1　エントロピー

情報源 S を符号化したときの平均符号長 L とその情報源の 1 次エントロピー $H_1(S)$ には以下の関係が成立する．

$$L \geq H_1(S) \tag{2.2}$$

つまり平均符号長の最小は理論値として与えられるということである．また，1 次のエントロピーは次のように計算できる．

$$H_1(P) = -\sum_{i=1}^{M} p_i \log_2 p_i \quad (p_i\text{は発生確率}) \tag{2.3}$$

2.5.2　瞬時符号の平均符号長

とにかく短ければよいわけではないことは符号化の条件について触れたところで述べている．瞬時符号であることが望ましいとあるがそれでも長くなってしまっては困りものである．しかし，瞬時符号に対してもその

平均符号長にはある程度の検討がつくのである．r 元符号による瞬時符号の最小値を L とすると 1 次エントロピーと以下の関係が成立する．

$$H_1(S) \leq L \leq H_1(S) + 1 \tag{2.4}$$

これで瞬時符号を作成するときの目標が定められる．

2.6 ハフマン符号化

2.6.1 ハフマン符号化とは

ハフマンによって与えられたコンパクト符号[*1]の構成方法である．ではここでは 2 元ハフマン[*2]の方法について述べる．

まずハフマン符号は発生率の分かっている情報源記号の組からすべての情報源記号を葉とする符号の木を作るのもであると理解しよう．この符号の木で符号を与えると平均符号長が最小となる符号ができるのである．

方法はまず発生確率最小の葉を 2 つ選びそれらの接点を作る．以降は接点を含めて最も確率の小さな 2 つの組を選んでそれらの接点を作る操作を繰りかえし，すべての情報源記号の葉が符号の木に組み込まれるまで続ける．この操作が終われば情報源記号を葉とする符号の木ができているはずである．あとはそれに対して符号を振ることで平均符号長が最小の符号が与えられる．

ここで面白いのは以上で示した操作は一意ではない[*3]のだがどのようにしても必ず最小符号長の符号化完了するということである．

2.6.2 q 元のハフマン符号

2 元では最小の確率を持つ 2 つの葉または接点の組を選んで新たな接点を作ったが，q 元であれば q 個の組を選べばいいわけである．ただ作成される符号の木の理想形は低い次数での分岐が多く，高い次数での分岐は極力少ない状態である．これは符号の数字を振るときに何となくではあるが気が付く事柄ではなかろうか．どのようにしてということはここでは説明しないが，2 元ではないハフマン符号の場合，低次の分岐が少なく高次の分岐が多い符号の木が得られやすいのが事実である．ひっかけを意図した問題では出題が予想される．

2.7 ブロック符号

2.7.1 ブロック符号の位置づけ

情報源記号を単体で扱いハフマン符号によって平均符号長最小の符号を与えても限界がある．しかし，情報源記号を 2 つ並べたものについてその全組み合わせに対してハフマン符号を与えたらどうなるだろうか．もちろん 2 個の全組み合わせを考えるのだから符号長は長くなるに決まっている．しかし注目すべきは情報源記号 1 つ当たりの符号長である．面白いことに短くなっている．

今の話は少し見えにくいのでもう少し具体的に説明すると，情報源記号 A,B に対してそのままハフマン符号ではなく，情報源記号の組 AA,AB,BA,BB に対してハフマン符号を与えるのである．情報源記号の組の発

[*1] 平均符号長が最小の符号のことをいう．
[*2] 2 元符号でのハフマン符号のこと．
[*3] 最小の確率の 2 つの組が複数選べる場合のことである．

生率はそのままものと情報源記号の発生率の積をとればよい．このような符号をブロック符号と呼ぶ．またハフマン符号との併用であるためハフマンブロック符号化と言ったりもする．

とても有効な方法にも思えるが欠点も存在しとにかく情報源記号の多いものに対してのハフマン符号を構成しなければならないのである．詳しい話は教科書参照のこと．

2.7.2 ブロック符号の平均符号長

n 個の情報源記号の組を 1 つとみたとき，その組み合わせを尽くしたものを M^n 元情報源，また n 次の拡大情報源という．この塊の状態での平均符号長を L_n とすると大元の情報源記号に対する平均符号長 L とは以下の関係がある．

$$L_n = nL$$

当然の関係である．では次は拡大情報源に対する平均符号長をもとめよう．M^n 元情報源の 1 次エントロピー $H_1(S^n)$ と L_n を式 (2.4) に適応させればよいが，実際に欲しいのは元の情報源記号に対する平均符号長である．ここで情報源 S の n 次エントロピー $H_n(S)$ を次の式で定義する．

$$H_n(S) \equiv \frac{H_1(S^n)}{n}$$

これを用いると L は次式で制限される．

$$H_n(S) \leq L \leq H_n(S) + \frac{1}{n} \tag{2.5}$$

n が大きくなると平均符号長が小さくなることが分かる．その極限を次に計算しよう．

2.7.3 エントロピー (その 2)

n 次拡大情報源に対する 1 次エントロピー $H_1(S^n)$ を次数で割ることで元の情報源の n 次情報源を得ることができた．この n を無限大にしたときの極限値は情報源のエントロピーと呼ばれる．

$$\lim_{n\to\infty} H_n(S) = H(S) \tag{2.6}$$

エントロピーはすべての n 次エントロピーより小さい．これに関しての証明はないが覚えておこう．

さて，式 (2.5) の極限を計算すると

$$H(S) \leq L \leq H(S) + \varepsilon \quad ({}^{\forall}\varepsilon > 0)$$

となり，エントロピーが平均符号長の最小を与えることが分かる．

2.8 ランレングス符号化法

ブロック符号のようにハフマン符号化を行った時に平均符号長が小さくなる方法がある．ここで紹介するのはその一つのランレングス符号化だ．例えば情報源 $\{A, B\}$ をブロック符号であれば，2 つや，3 つの組で新たな情報源記号を作る．どんな出現パターンにも対応するには場合を尽くす必要があることは先の説明でもわかっていることだろう．

対してランレングス符号では長さが異なる組で出現パターンを網羅する．以下のようになる．

$$\{AAA, AAB, AB, B\}$$

確かにこれでも出現パターンは網羅できる．ここでは暗に A が B より出てきやすい条件を与えている．

2.8.1　ランレングス符号化法の平均符号長

詳しい計算は省くが，ここで注意したいのはランレングス符号では情報源符号の組の長さがまちまちになるので，ブロック符号における次数に相当する数を用意しなければならないことである．これを $\bar{n}$ とする．

$$\bar{n} = \frac{1-(1-p)^{N-1}}{p}$$

で与えられる．ここでの p は 2 つの情報源記号のうち発生率の小さいほうの発生率であり，N はランレングス符号で作った情報源系列[*4]の数を表している．これで計算できた $\bar{n}$ を用いるとランレングスハフマン符号の平均符号長は，

$$L < H(S) + \frac{1}{\bar{n}}$$

となりほとんど式 (2.5) の左辺と同形である．ただ，情報源記号の発生率に偏りがある場合，すなわち同じ情報源記号が連続しやすい場合に極めて有利だ．

2.8.2　ハフマンブロックとランレングスハフマンの比較

平均符号長の最小値を与える不等式がほぼ同形であったことから，$n, \bar{n}$ の比較が両者の比較につながる．ハフマンブロックにおいて，

$$n = \log_2 N$$

対してランレングスハフマンでは，

$$\bar{n} = \frac{1-(1-p)^{N-1}}{p}$$

である．N は情報源系列の個数であるが，これは装置の複雑さに直結している．p が小さいときなど特にランレングスハフマンに軍配が上がるだろう．ハフマンブロックは愚直に装置を増やしていけば理論的な平均符号長に近づくことはできるが現実的ではないのだ．

2.9　算術符号

ハフマンブロック符号化，ランレングス符号化は情報源系列をあるルールで区分し，区分された部分に符号語を対応させてゆく方法であった．このような方法では符号化を終えた情報源系列は符号語を並べた列となっていることが分かる．これに対して，情報源系列前提を 1 つの符号語としまう符号化が存在する．その代表格が算術符号だ．装置は簡単で効率が良く，さらに様々な情報源に対応できるという極めて優れた符号化なんだそうだ．

2.9.1　累積確率

2 元符号の情報源系列をとりあえず符号を 2 進数として読んで順番に並べれば，番号を振ることができる．ここでは 0 から番号を振ることにする．i 番目の情報源系列を a_i と表現し，その発生確率を $P(a_i)$ とする．

[*4] 情報源記号の組

情報源系列の数が多くなければ容易に計算できてしまう．さて，このような状況で以下のように与える $C(a_i)$ を a_i までの累積確率という．

$$C(a_i) = \begin{cases} 0 & (i = 0) \\ \sum_{j=0}^{i-1} P(a_j) & (i \neq 0) \end{cases}$$

情報源系列の数が増えると計算がしんどさを持つようになるが，これは教科書 P.78-79 を参照してほしい．

2.9.2 算術符号

累積確率が計算できたら次は符号化である．端的に言って累積確率をそのまま 2 進数として小数点以下のみに注目すればいいわけだ．当然，小数は無限に続く場合があるしなにも律義に符号とする必要はない．他との区別がつきさえすればいいのでそこは柔軟に符号化しよう．

2.10 ユニバーサル符号

情報源系列の中の記号の出現率などの事前情報なしに符号化でき，なおかつ情報源系列が長いほど高い圧縮効果が期待できるそうだ．ここでは 2 つのユニバーサル符号を紹介する．

2.10.1 LZ77

辞書という発想を用いて圧縮を図る．この方法ではパラメータと呼ぶ符号化のルールを先に決定することで符号化を実行できる．パラメータは以下の 2 つ．

- 辞書のサイズ (a とする)
- 最大一致長 (b とする)

やり方を簡単に説明する．まず辞書とは符号化し終えた情報源系列の末尾 a 個をその順で並べたもの．また辞書の要素にはページのごとく 0 から $a-1$ までの番号が振られている．符号化を実行するときは，まだ符号化を終えていない情報源系列の先頭の部分を確認し，辞書にあるパターンと同じものがあるのかを確認する．このときにどれほどたくさん一致していても b を超える個数分は一致していないものとする．そうすると次の 3 つの情報が手に入る．

- 辞書の何番目から一致しているか (一致がなければ 0)
- 辞書の何個分が一致しているか (一致がなければ 0)
- 辞書と一致できなかった情報源系列の先頭[*5]

手に入った 3 つの情報を符号化することで一致の部分を詳しく符号化しなくてよく圧縮につながる．そしてそのあとに辞書が更新される．

[*5] この表現はわかりにくいが，辞書と二一致があれば一致したパターンの次の情報源記号，一致がなければ符号化していない先頭の情報源記号のことである．

2.10.2　LZ78

この符号化は先ほどの辞書が情報源系列を移動するのとは違い，新たなパターンを発見したらそのたびに辞書に加えてゆく方法である．このとき辞書に追加するときにそのパターンに番号が与えられる．符号化はまず未符号化の系列の先頭を部分を確認する．持っている辞書と一致するパターンがあればその辞書の番号とともに，そうでなければ 0 とともに次の記号を符号化する．このとき，符号化できた記号の組を新たに辞書に追加する．

第 3 章

情報源のモデル

符号化について長々説明したが次は符号化前の情報源それ自体に注目したい．

3.1　情報源系列と確率

3.1.1　結合確率分布

n 個の情報源系列が以下のようなものだ．

$$X_0X_1X_2\cdots X_{n-1}$$

X_i は時点 i での出力であり[*1]，M 個の情報源記号のいずれかをとるものと考えれば，X_i 自体が確率変数とみなせることが分かる．確率変数であるのならば，確率を計算すべきだろう．

まずは足掛かりに i 番目で $X_i = x$ となるときの確率を表し方を示す．

$$P_{X_i}(x_i)$$

全く面白みはないが，ここから話を広げてゆく．情報源系列が列をなし多くの確率変数を擁しているのだから，そのうちの複数をとった確率を考えるのは自然な流れだろう．では実際に情報源系列の 1 番目と 2 番目に対する確率を示す．

$$P_{X_0,X_1}(x_0,x_1)$$

添字の数がずれていることには注意したいがやはり大したことはない．

[*1] 出力とはすなわち情報源アルファベットのことだ．

ところで，示し方が分かったところで何の意味もない．次は公式である．

$$P_{X_0}(x_0) = \sum_{x_1}\sum_{x_2}\cdots\sum_{n-1} P_{X_0X_1X_2\cdots X_{n-1}}(x_0, x_1, \cdots, x_{n-1})$$
$$P_{X_0,X_1}(x_0, x_1) = \sum_{x_2}\cdots\sum_{n-1} P_{X_0X_1X_2\cdots X_{n-1}}(x_0, x_1, \cdots, x_{n-1})$$

式からも計算の発想は理解できる．固定されている部分を除いてすべての場合を尽くしているだけだ．

このような計算で与えられるものを結合確率分布という．

3.1.2 条件付確率

確率と聞いて避けては通れないのが条件付き確率であろう．定義は通常の確率と同じである．

$$P_{X_1|X_0}(x_1|x_0) = \frac{P_{X_0,X_1}(x_0, x_1)}{P_{X_0}(x_0)}$$

表記が分かりにくいが，上の式は X_0 を条件とした X_1 の確率である．

3.2 記憶のない定常情報源

記憶のない情報源，定常情報源．全く聞きなれない言葉に意味することの想像も難しい．まずは言葉の意味から述べる．記憶のない情報源とは，ある時点の情報源記号の発生が他の時点に対して独立している情報源のことをいう．そして定常情報源とは，どの時点の情報源記号の発生分布も同じである情報源のことだ..

記憶のない定常情報源を式で表すと，情報源系列 $X_0X_1\cdots X_{n-1}$ の結合確率分布は

$$P_{X_0,X_1,\cdots,X_{n-1}}(x_0, x_1, \cdots, x_{n-1}) = \prod_{i=0}^{n-1} P_X(x_i)$$

時点に依らず確率は与えられ，独立によって積の法則が使われていることが分かる．

定常情報源に対しては以下の式もその性質が分かりやすく表れている．

$$P_{X_0,X_1,\cdots,X_{n-1}}(x_0, x_1, \cdots, x_{n-1}) = P_{X_i,X_{i+1},\cdots,X_{i+n-1}}(x_0, x_1, \cdots, x_{n-1})$$

情報源系列のどの部分を取り出してもその結合確率分布は変わらないことが分かる．

3.3 エルゴード情報源と情報源の平均

3.3.1 エルゴード情報源とは

サイコロを振って情報源系列を作ることを考える．その情報源系列に何の意味があるのかという話ではあるが，このときの情報源系列は十分に長ければ 6 つの情報源記号が等確率で現れていることが分かるだろう．エルゴード情報源とは，十分に長い任意の情報源系列にその情報源の統計的性質が分かる定常情報源のことをいう．またこのような性質をエルゴード性という．

3.3.2 集合平均・時間平均

エルゴード性についてもう少し数学的に理解を進めたいところである．そこでまずは集合平均という概念をここで与えたい．

まず定常情報源 A からの出力 X を変数とする実数値関数 $f(X)$ を置く．この関数は任意で，数字ではない情報源の出力を数字にする事を目的としている[*2]．関数値は変数によって確率的な変動をするので，関数値自体が確率変数である．その期待値を計算すると，

$$\bar{f(X)} = \sum_{x \in A} f(x) P_X(x)$$

となる．これを集合平均という．出力される情報源記号の期待値を数値にしたと考えれば難しくない．

次は時間平均だ．1 つの情報源からの出力系列 $x_0 x_1 x_2 \cdots$ にについて以下のように与えられる算術平均の極限を時間平均という．意味は式から十分に読み取れる．

$$\langle f(X) \rangle = \lim_{n \to \infty} \frac{1}{n} \sum_{i=0}^{n-1} f(x_i)$$

エルゴード情報源では以上の 2 つの平均が一致する．この性質は有用で，比較的に計測が容易な時間平均から集合平均が導き出せることは便利なのだ．

3.4 マルコフ情報源

マルコフ情報源とは記憶のある情報源の基本的な例だ．記憶のある情報源とは出力の決定を過去の出力によっておこなうものだ．これには状態という発想が必要になる．

確率基礎論で扱った内容の同じ部分もある．遷移確率と行列の知識があると多少だが理解が早まると思われる．

3.4.1 m 重マルコフ情報源

$n, m \in \mathbb{Z}\ (n \geq m)$ とする．任意の時点 i での X_i がその直前 n 個の出力を条件とする条件付確率分布と，直前 m 個を条件とする条件確率分布が等しいとき，その情報源を m 重マルコフ情報源という．

$$P_{X_i|X_{i^1}\cdots X_{i-n}}(x_i|x_{i-1}\cdots x_{i-n}) = P_{X_i|X_{i^1}\cdots X_{i-m}}(x_i|x_{i-1}\cdots x_{i-m})$$

この式は m 以上さかのぼった出力は新たな出力に関与しないと解釈することができる．また，当然ではあるが m はできる限り小さいものを選んでいる．

$m = 1$ のときには特別な呼び名が決まっていて 1 重マルコフ情報源を単純情報源と呼ぶ．

3.4.2 状態

m 重マルコフ 2 元情報源を考える．任意の時点の出力はその直前の m 個の出力にによって定まるが，毎回過去にさかのぼるのは面倒だ．ならば過去 m 個の出力の情報を持った何かを現在に保持[*3]できれば楽な話だと思う．ここで登場するのが状態である．状態というのが過去の出力によって決定するため，過去の情報を現在に保持しているといえるのだ．

[*2] 情報源に番号を振るので十分だ．

[*3] 過去に何が出たかが判別できる象徴的な何かがあれば，その情報 1 つで次の出力が決定できる．

さて，2 元情報源で過去 m 個分の出力の情報を残そうと思うと，状態は 2^m 必要だ[*4]．そして状態は他の状態と同じであることはない．

3.4.3 状態図と遷移

新し出力が発生すると状態は変わり別の状態へ変わる可能性がある．この状態が変化する可能性とその変化先の状態をまとめた図を状態遷移図という．図 3.1 が単純マルコフ情報源の状態図で，図 3.2 はやや一般化されている．

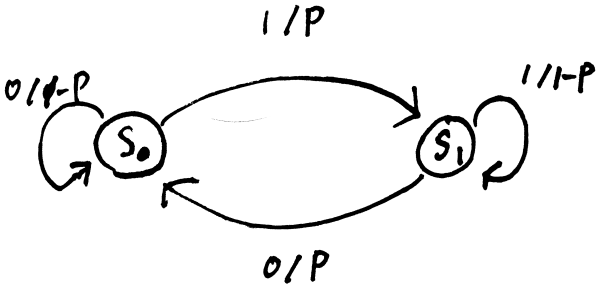

図 3.1　単純マルコフ情報源

矢印が状態の変化先を示し，そのわきに添えられている 2 つの数が，情報源記号/その発生確率，を表している．

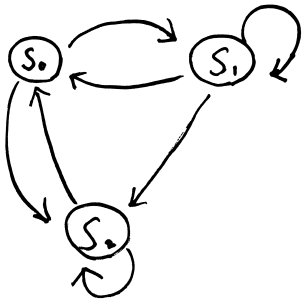

図 3.2　一般のマルコフ情報源の状態図の例

3.4.4 状態の分類

状態図の一部分からその部分がどのような種類の分類に当てはまるのかが分かる．これは情報源の性質を決定する 1 つの要素だ．

過渡状態　図 3.3 のような部分を過渡状態という．2 つの状態を行き来するが，一度その 2 つの状態から抜け出すと戻ることはない．

[*4] 過去の出力によって状態は 1 つに定まる．m 重マルコフ 2 元情報源ならば，過去の出力のすべての組み合わせを区別するには当然 2^m 個の状態が必要だろう．

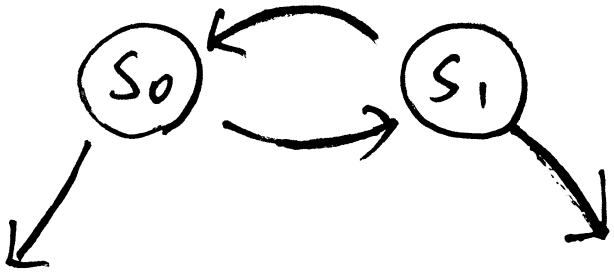

図 3.3　過渡状態

非周期的な閉じた状態集合　図 3.4 のような部分を非周期的な閉じた状態集合という．まずは閉じた状態集合というものを説明しよう．閉じた状態集合とは，複数の状態の集まりの中でも一度その集まりに状態が遷移すると，その集合内に遷移が収まり，他の状態をとることがなくなるものだ．

非周期的とは状態の集合内での遷移に周期性がないもので，図 3.4 では状態 S_2 が他とは異なる遷移をする．

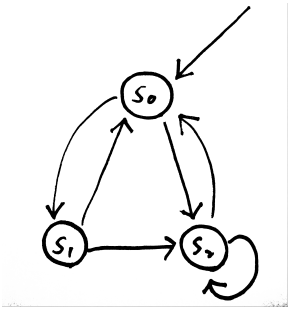

図 3.4　非周期的な閉じた状態集合

周期的な閉じた状態集合　図 3.5 のような部分を周期的な閉じた状態集合という．閉じた集合内の遷移が周期的であることが確認できる．

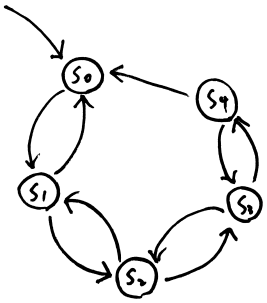

図 3.5　周期的な閉じた状態集合

状態の分類が分かったところで話を周期的な閉じた状態集合に絞る．この状態集合に状態が遷移すると，以

降の時点での状態は大方決定する．例えば 3 の倍数の時点でのみ状態 1 または 2 をとるというようなことだ．このように各状態がある周期的な時点にのみ現れる場合，これを周期的状態集合という．

一般のマルコフ情報源では先に挙げた 3 つの状態を部分的に持つ状態図となるだろうが，十分に長い情報源であれば過渡現象は考えなくてよい．もっと言えば過渡現象を扱うことのほうが稀だ．そこで，ここからは過渡状態を省いて考える．

さらに考えれば十分に長い時間の経過を仮定すれば閉じた情報源集合に状態は既に遷移していると考えるべきである．このような考えから，情報源は 1 つの閉じた状態集合だけからなると考えていい．実際は複数の閉じた状態集合を持っていたとしても，分ければよいのだ．このように 1 つの閉じた状態集合からなるマルコフ情報源を既約マルコフ情報源という．

既約マルコフ情報源のうち閉じた状態集合が非周期的のものを正規マルコフ情報源という．

3.4.5 遷移確率行列とマルコフ情報源のエントロピー

時点 i においてどの状態にあるのかというのは確率で表すことができる．状態に適当に 0 から番号を振り，時点 i において j 番目の状態をとる確率を $w_j^{(i)}$ とする．

$$\boldsymbol{w}_t = (w_0^{(t)}, w_1^{(t)}, \cdots)$$

上に示した式のように時点 i の確率はまとめられる．要素の総和はもちろん 1 だ．

これが次の時点に進むときの確率は以下のようにあらわせる．

$$\boldsymbol{w}_{t+1} = \prod \boldsymbol{w}_t \quad (\prod \in M_n(\mathbb{R}))$$

このような行列 $\prod$ 遷移確率行列という．この行列の作成の方法は各自に任せる[*5]．

遷移確率行列があれば十分に時間が経った状態を式で表すことができる．

$$\boldsymbol{w} = \prod^{n} \boldsymbol{w}_0 \quad (n \to \infty)$$

これを極限分布という．そしてこの極限分布は正規マルコフ情報源では収束し，

$$\boldsymbol{w} = \prod \boldsymbol{w}$$

定常分布と呼ぶ．行列の計算によって容易に $\boldsymbol{w}$ は求められる[*6]．

さて，マルコフ情報源のエントロピーは以下のように与えられる．

$$H(S) = \sum_{i=0}^{N-1} w_i \left[-\sum_{j=1}^{M} P_{(a_j|s_i)} \log_2 P_{(a_j|s_i)} \right] \tag{3.1}$$

$$= \sum_{i=0}^{N-1} w_i H_{s_i}(S) \quad (H_{s_i}(S) \text{ は状態 } s_i \text{ でのエントロピー}) \tag{3.2}$$

[*5] 状態図から連立方程式を立てればあとは無理やり行列を作ればよい．

[*6] $^t\prod$ の核を計算して要素の和が 1 となるよう実数倍すればよい．

第 4 章

誤りの発見と訂正

講義は後半の内容に入っている．前半の行為ではいわゆる通信の経済性の問題を取り扱った．情報源の発生確率に注目した符号化や辞書を用いた符号化で送信するビット列自体を短くする試みを勉強した．これに対して後半の内容では信頼性の問題を取り扱う．

送信中にビット列が誤ってしまうことは避けられない事象だ．これを元の正しいビット列を知らない状態で，発見さらには訂正することができれば信頼性は増すだろう．

この章はその入り口となる内容を扱う．

4.1　単一パリティ検査符号

4.1.1　符号化の方法

誤りの発見を実現する符号だ．発想としては送信するビット列の全ビットの排他的論理和[*1]を計算し，送信したいビット列の最後に付加することで符号語と成す．このようにすると符号語の全体で排他的論理和をとると必ず 0 となる．

実際に符号語を作る．送信したい符号語を $(x_1, x_2, \cdots, x_k)$ とする．ここで c を次のように定義する．

$$c := x_1 \oplus x_2 \oplus \cdots \oplus x_k \tag{4.1}$$

このようにすると送信したいビット列と c には以下の関係が生まれる．

$$x_1 \oplus x_2 \oplus \cdots \oplus x_k \oplus c = 0$$

そして符号語 $\boldsymbol{w}$ を

$$\boldsymbol{w} = (x_1, x_2, \cdots, x_k, c) = (w_1, \cdots, w_{k+1})$$

する．繰り返すようだがこの符号語には以下の式が成立する．

$$w_1 \oplus \cdots \oplus w_{k+1} = 0 \tag{4.2}$$

これをパリティ検査方程式という．

[*1] 複数ビットでの排他的論理和の計算はビットの総和を 2 で割ったあまりと同じだ．

4.1.2 誤りの検出

受信側は元のビット列を知ることなく $\boldsymbol{y} = (y_1, \cdots, y_{k+1})$ を受け取る．これの実情は必ずしも $\boldsymbol{w}$ とは一致しない．送信中に何らかの誤りが物理現象として発生する．この影響を $\boldsymbol{e} = (e_1, \cdots, e_{k+1})$ とする．すると受信されるビット列は

$$\boldsymbol{y} = \boldsymbol{w} \oplus \boldsymbol{e}$$

これはビットごとの排他的論理和を表していると読み取ってほしい．$\boldsymbol{e}$ の要素がすべて 0 なら $\boldsymbol{y} = \boldsymbol{w}$ となるが実際どうなのかはわからない．$\boldsymbol{e}$ の見方は，誤りが発生するビットとの部分が 1 となっている点だ．つまりすべて 0 なら誤りはないことになるし，さらに 1 の個数が誤りの個数となっている．

誤りの発見は検査方程式で行われる．

$$y_1 \oplus \cdots \oplus y_{k+1} = \begin{cases} 0 & \text{誤りのない可能性がある} \\ 1 & \text{誤りがある} \end{cases}$$

全く決定的ではないところに残念な気持ちすら感じるが仕方がない．それは気にせず話を進める．検査方程式の結果から

$$s := y_1 \oplus \cdots \oplus y_{k+1} \tag{4.3}$$

を定義する．s をシンドロームと呼ぶ．シンドロームの計算を進めると

$$\begin{aligned} s &= y_1 \oplus \cdots \oplus y_{k+1} \\ &= w_1 \oplus \cdots \oplus w_{k+1} \oplus e_1 \oplus \cdots \oplus e_{k+1} \\ &= e_1 \oplus \cdots \oplus e_{k+1} \end{aligned}$$

このようになる．シンドロームは誤りの個数を 2 で割ったあまりとなる．誤りがない場合は $s = 0$ だが，誤りが偶数個の場合も $s = 0$ となる．また誤りが奇数個の場合は $s = 1$ となり直ちに誤りがあることが分かる．シンドロームの値は誤りの有無を決定するためのものだ．

4.1.3 復号

最初に送りたかったビット列に直すが，これは誤りがないであろうと考えられた場合 $(s = 0)$ に最後のビットを消すだけのことだ．しかし，偶数個の誤りがある場合に誤った復号をしてしまう．これを復号誤りという．

ここで誤りの個数とシンドローム，復号結果などをまとめる (表 4.1)．復号できない判定が出たらもう一度

表 4.1　単一パリティ検出符号

	シンドローム	判定結果	復号結果
誤りなし	0	誤りなし	正しく復号
単一誤り	1	誤りあり	復号しない
偶数個誤り	0	誤りなし	正しく復号しない
奇数個誤り	1	誤りあり	復号しない

送信をお願いすることで対処する．

4.2　組織符号

直前の節で送信したいビット列 $\boldsymbol{x}$ にその全ビットの排他的論理和 c を加えることで送信する符号語 $\boldsymbol{w}$ を作成した．このように情報ビットと検査ビットを組み合わせることで符号語とするものを組織符号という．符号全体の長さを n，そのうちの情報ビット数を k とすると，(n, k) 符号とも呼ぶ．また組織符号には効率という概念があり，効率 h は以下のように定義される．

$$h := \frac{k}{n}$$

ちなみに，単一パリティ検査符号においては $n = k + 1$ だ．

4.3　水平垂直パリティ検査符号

単一パリティ検査符号では 1 ビットの誤りのみが検出できたが偶数個では検出もできず，なおかつ正しいものに誤り訂正することもできない．そこでもっとシンドロームが増えるようにしたい．

これを達成する考えとして送信する情報ビットを長方形に並べ，各行と各列に単一パリティ検査符号と同じことをする．

$$\begin{bmatrix} x_{11} & \cdots & x_{1k} \\ \vdots & \ddots & \vdots \\ x_{k1} & \cdots & x_{kk} \end{bmatrix} \rightarrow \begin{bmatrix} x_{11} & \cdots & x_{1k} & c_1 \\ \vdots & \ddots & \vdots & \vdots \\ x_{k1} & \cdots & x_{kk} & c_{k_1} \\ c'_1 & \cdots & c'_{k_2} & c'' \end{bmatrix}$$

このような符号化だ．縦と横でシンドロームを計算し，もし 1 ビットの誤りがあった場合は行のうち 1 つ，列のうち 1 つのシンドロームが 1 となるのでそこから誤りの位置が分かる．つまり水平垂直パリティ検査符号では誤り訂正が可能だ．しかし，2 ビットの誤りでは検知は可能でも訂正はできない．これはシンドロームが 4 つ現れることで誤り箇所が 4 か所だと誤認してしまったり，行または列のいずれか一方に $s = 1$ が集まることで誤り箇所が分からなかったりするからだ．さらに 3 ビット誤りでは，1 ビット誤りだと誤認して誤った復号誤りが発生する．

送信するビット数が増えた割にあまり効果的ではないように思える．

4.4　(7,4) ハミング符号化

水平垂直パリティ検査符号では 1 ビットの誤りを訂正するのに多くのビットを足しすぎた印象だ．では実際に 4 ビットの情報を送信するとき何ビット新たに加えれば 1 ビットの誤りを訂正できるのか．答えから述べると 3 ビットだ．しかし正確な表現ではないだろう．送信する合計 7 ビットのうち 3 ビットを検査にあてることで 7 ビットのどこが間違っているのかを指示できるのだ．これは 3 ビットで 8 つの場合を表現できることに関係していると考えるべきだ．

4.4.1 符号化

送信したいビット列 (x_1, x_2, x_3, x_4) に対して，検査ビットとなる c_1, c_2, c_3 を次の式で定義する．

$$
\begin{aligned}
c_1 &= x_1 \oplus x_2 \oplus x_3 \\
c_2 &= x_2 \oplus x_3 \oplus x_4 \\
c_3 &= x_1 \oplus x_2 \oplus x_4
\end{aligned}
$$

この組み合わせ自体に大きな意味があるのかといわれると，それほどでもない．具体例の 1 つとして考えるとよいだろう．しかし，この最初のこの部分の定義が肝心なのでこのまま進める．

符号語は情報ビットと検査ビットより，

$$
\begin{aligned}
\boldsymbol{w} &= (x_1, x_2, x_3, x_4, c_1, c_2, c_3) \\
&= (w_1, \cdots, w_7)
\end{aligned}
$$

となる．この符号化によるパリティ検査方程式は

$$
\begin{aligned}
w_1 \oplus w_2 \oplus w_3 \oplus w_5 &= 0 \\
w_2 \oplus w_3 \oplus w_4 \oplus w_6 &= 0 \\
w_1 \oplus w_2 \oplus w_4 \oplus w_7 &= 0
\end{aligned}
$$

この 3 式の連立方程式となる．この辺りは単一パリティと似ている．

4.4.2 誤り検出・誤り訂正

シンドロームは 3 つ定義でき，

$$
\begin{aligned}
s_1 &= y_1 \oplus y_2 \oplus y_3 \oplus y_5 \\
s_2 &= y_2 \oplus y_3 \oplus y_4 \oplus y_6 \\
s_3 &= y_1 \oplus y_2 \oplus y_4 \oplus y_7
\end{aligned}
$$

となる．単一パリティ同様の式変形から，誤りのビット列を $\boldsymbol{e}$ とすると，

$$
\begin{aligned}
s_1 &= e_1 \oplus e_2 \oplus e_3 \oplus e_5 \\
s_2 &= e_2 \oplus e_3 \oplus e_4 \oplus e_6 \\
s_3 &= e_1 \oplus e_2 \oplus e_4 \oplus e_7
\end{aligned}
$$

このような結果となる．1 ビットの誤りがある場合，$\boldsymbol{e}$ の要素のいずれか 1 つのみが 1 となる．そこで $e_1 = 1$ を仮定すると，$s_1 = 1, s_2 = 0, s_3 = 1$ となる．では他のものが誤った場合はどうなるのか．ここは各自に任せるがいずれの場合でもシンドロームの組が同じになることはない．つまり，シンドロームの組と誤り箇所に 1 対 1 の関係がある．これは誤り訂正が可能だといっているものだ．ちなみに今の条件ではシンドロームの組と誤りパターには表 4.2 の関係がある．誤りが 2 ビットを超えると復号誤りが発生する．

表 4.2　誤りパターンとシンドロームの関係

誤りパターン							シンドローム		
e_1	e_2	e_3	e_4	e_5	e_6	e_7	s_1	s_2	s_3
1	0	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	1	1	1
0	0	1	0	0	0	0	1	1	0
0	0	0	1	0	0	0	0	1	1
0	0	0	0	1	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0
0	0	0	0	0	0	1	0	0	1
0	0	0	0	0	0	0	0	0	0

4.4.3　ハミング符号を行列から理解

シンドローム s と送信するビット列 w には表 4.2 などを参考にすると検査方程式は次のようにあらわすことができる．

$$\boldsymbol{w}\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = 0$$

シンドロームに対しても，

$$\boldsymbol{s} = \boldsymbol{y}\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

となる．右辺の 7 行 3 列行列を検査行列という．注目すると上の 4 行分が各列 2 つ以上の 1 がある．またこの 4 行は行ベクトル，列ベクトルともに 1 次独立だ．下 3 行は単位行列になっている．この検査行列を以下のように表記する．

$$\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{C}_{43} \\ \mathbf{1}_3 \end{bmatrix}$$

この $\boldsymbol{C}$ さえ作成しておけば容易にハミング符号が作れる．符号化は以下の式だ．

$$\boldsymbol{w} = (x_1, x_2, x_3, x_4) \begin{bmatrix} \mathbf{1}_4 & \boldsymbol{C}_{43} \end{bmatrix}$$

シンドロームと検査行列の i 行目の比較で第 i ビットの誤りを検知できる．これがハミング符号の行列による表現だ．

4.4.4 ハミング符号の能力

ハミング符号は 1 ビットの誤りに対して誤り訂正をおこなえる．2 ビット誤りに対しては復号誤りを起こす．さらに 3 ビット誤るとこれは別の符号語となる．これがハミング符号の限界となる．この根拠はどこからかか．実際書いてみればわかることだがそれでは芸がない次の節にはその部分をから扱おう．

4.5 符号の能力

4.5.1 諸定義

前章のハミング符号化では 1 ビットの誤りを復号できるということを実際にやればできるというような説明で示した．しかし，これでは数多ある符号をすべていちいち試せばいいのかという話になってしまう．これはかなり困ったことになる．そこで，符号語を空間の座標にも近い捉え方をすることで考える．まずは以下の 4 つを定義する．

ハミング距離　長さの同じ 2 つのビット列で異なるビット数をハミング距離という．

ハミング重み　n 次元ベクトルの 0 ではない要素数のことをハミング重みという．これはあるビット列と同じ長さの要素がすべて 0 のビット列とのハミング距離にあたる．

最小距離　符号のすべての符号語でハミング距離を計算したときの最小値．

最小重み　符号の中で要素がすべて 0 ではない符号語のうち最小のハミング重み．

厳密な定義ではないがこれで十分だ．またしばらく扱う符号は最小重みと最小距離は同じだ．

また，送信した符号語 $\boldsymbol{w}$ に $\boldsymbol{e}$ の影響が加わった結果，$\boldsymbol{y}$ を受信したとする．このとき，$\boldsymbol{w}, \boldsymbol{y}$ のハミング距離と $\boldsymbol{e}$ のハミング重みは等しい．

$$d_{\mathrm{H}}(\boldsymbol{w}, \boldsymbol{y}) = w_{\mathrm{H}}(\boldsymbol{e})$$

4.5.2 復号領域

送信した符号語と受信したビット列を座標のようにとらえると，両者はハミング距離の分離れているとイメージできるだろう．さらに同じ符号語に対して 1 ビットずれたビット列をすべて挙げると，半径 1 の球ができ，2 ビットずれたものを挙げたら半径 2 の球ができたりする様が想像できたら完璧だ (図)．

このような球体は符号語の個数分できる．ここで注目したいのは最小距離を与える 2 つの符号語のみだ．この 2 点ともいうべき 2 つの符号語を中心とする 2 つの球を想像する．この 2 つの球が共有する領域のないように取れる最大の半径 t_0 はどのようになるだろうか．それは最小距離 $d_{\min}$ によって次のように与えられる．

$$t_0 = \begin{cases} \dfrac{d_{\min} - 1}{2} & (d_{\min}\text{は奇数}) \\ \dfrac{d_{\min} - 2}{2} & (d_{\min}\text{は偶数}) \end{cases}$$

表 4.3　誤りの個数と復号，誤り検出

誤りの個数	結果
t_1 以下	復号可能
t_1+1 以上 t_1+t_2 以下	確実に誤りを検出
t_1+t_2+1 以上	復号誤りの可能性

では符号を作成した後に，$t_1 \leq t_0$ なる t_1 をとる[*2]．そして，あらかじめ符号語の周りに半径 t_1 の球を想定したらどうなるか．球は必ず他の球と共有する部分を持たない状態である．その場合，伝送の間に誤りが乗じたビット列も球の中に納まると，その中心の符号語だと結論づけることは可能ではないか．もちろん大幅に誤って別の符号語の球の領域のビット列になってしまえばどうしようもない話だが，t_1 個のビット数の誤りは訂正できるという事になる．

このような復号領域，誤り訂正能力という考え方に整理されている．今の話で現れた t_0 が誤り訂正能力というもので．t_1 は復号領域の半径という．t_1 は都合よく決定していい．さらに t_2 を

$$t_2 = d_{\min} - 2t_1 - 1$$

と定義しておくことで次の表の関係が分かる．

これでもう前章のハミング符号が 1 ビットの誤り訂正できる理由はわかっただろう．(7,4) ハミング符号は最小距離が 3 だからだ．

さて，感が良ければ t_1 をとにかく t_0 に近づければ誤り訂正できる範囲が大きくなる[*3]と思いつく．しかし，これはいいところばかりというわけではなく，t_2 の値が小さくなることで，復号誤りの可能性が大きくなるのだ．つまり，誤り訂正が可能な確率と復号誤りを起こしてしまう確率は両天秤の関係にある．

第 5 章

巡回符号

巡回符号とはビット列を多項式の係数に見立てて多項式にできる計算から符号語を作成するものだ．数学的な知識が多く盛り込まれているため心して学習したい．

5.1　符号化

何においても符号化の方法を最初に扱う．

[*2] これは任意だ．

[*3] より多くのビットが誤っていても訂正できる．という意味．

5.1.1 多項式表現

符号語はビットの列だが，これを行ベクトルと見ることは難しくない．これを $\boldsymbol{v}$ とする．$\boldsymbol{v}$ の要素の番号を次のように降順で振る．

$$\boldsymbol{v} = (v_{n-1}, v_{n-2}, \cdots, v_1, v_0)$$

そうすると行ベクトル $\boldsymbol{v}$ を多項式 V で次のように表現できる．

$$V(x) = \sum_{i=0}^{n^1} v_i x^i \tag{5.1}$$

係数が 1 のみの多項式となる．

この表現によって情報多項式や検査多項式，受信多項式といったりする．

5.1.2 巡回符号の発想

生成多項式なる関数を $G(x)$ として用意する．これはベクトルの多項式表現で得られるような係数が 1 のみの式だ．ここで符号を多項式表現したときに必ず $G(x)$ で割り切れるように作成したらどうなるだろうか．送受信する両者が $G(x)$ を共有することで誤りを検出することができるだろう．また，誤りがあれば受信語の多項式を $G(x)$ で割ったときにあまりの多項式が出てくるがこれがシンドロームに対応するだろう．

5.1.3 巡回符号の作成

基本的は方法は今までのパリティ検査符号と同じで，情報ビット列 $\boldsymbol{x}$ に検査ビット列 $\boldsymbol{c}$ を足す．これを多項式表現で実現するには情報多項式 $X(x)$ に x^m をかけることで検査多項式 $C(x)$ を足すスペースを作るが必要だ．このような m は次の計算で求められる．

$$m = \mathrm{deg}G(x)$$

いきなり何の計算だという話だが，$G(x)$ の最高次の次数をとればいい．これは多項式を $G(x)$ で割ったときのあまりを検査ビットとするため，あまりの次数に 1 加えた幅[*1]が必要だということを表している．

このように m を決定すれば検査多項式 $C(x)$ は

$$C(x) := X(x)x^m \bmod G(x) \tag{5.2}$$

となる[*2]．これによって符号語多項式 $W(x)$ は

$$W(x) = X(x)x^m + C(x) \tag{5.3}$$

となる．どうして引かないのかと考える人は多いだろうが，これは係数が 0 か 1 しかないことに理由があり足すことも引くことも変わらない．これに関してはのちに詳しく説明する．

[*1] 幅という表現で伝わるだろうか．つまり，あまりの最高次数より m が小さいと情報ビット列と検査ビット列が被る部分が発生する．これを回避する最小の m を与えたいのだ．

[*2] "$X(x)x^m \bmod G(x)$"この書き方は全く正式ではない．$X(x)$ を $G(x)$ で割ったときの剰余の多項式という意味で読んでほしい．

巡回符号は以上で説明した方法で与えられる．何度も現れた $G(x)$ は生成多項式と呼ばれ巡回符号の，作成・検査のカギとなる．そして

$$G(x) = x^3 + x + 1$$

のとき，与えられる巡回符号は (7,4) ハミング符号だ．

5.1.4　巡回符号の検査

この符号の検査方程式は

$$\begin{aligned} W(x) \bmod G(x) &= [X(x)x^m + C(x)] \bmod G(x) \\ &= C(x) + C(x) \\ &= 0 \end{aligned} \tag{5.4}$$

誤りがあればシンドロームがあるわけだが，多項式 $S(x)$ として

$$S(x) = E(x) \bmod G(x) \tag{5.5}$$

で与えられる．シンドロームが 0 なら誤りなしと判定するが，$E(x) = G(x)$ のときは誤りがあってもシンドロームが 0 となってしまう．このパターンでは検出ができないことを覚えておこう．

5.2　違和感のある計算

巡回符号の符号語の作成では所どころ違和感のある計算があった．もっと言えば係数が 1 のみの多項式での除算とはどのようなものか確かなイメージがわかない．

5.2.1　ガロア体

定義は”有限個の元からなる体”のことだ．しかし，この”体”というものから難しい．体とはその中で加法・乗法が閉じ，可換．さらには結合法則と分配法則が成立し，逆元・零元が存在しているものをいう．書いていてもよくわからないが実数でできることが有限個の元で可能であればそれは体だ[*3]．

元の個数が q のときのガロア体を $GF(q)$ と表記する．

5.2.2　$GF(2)$

元の数が 2 つのガロア体だ．既に計算での違和感があったので表 5.1 にまとめる．

この $GF(2)$ の元を係数に持つ多項式を $GF(2)$ 上の多項式という．今まで出てきているビット列の多項式表現はすべてこれだ．多項式の計算の際，係数は表 5.1 に従って行えばいい．加算・減算が排他的論理和に相当するのは面白くさえ感じる．

5.3　巡回符号の詳細

巡回符号を作れるくらいではわかった内には入らない．ここからは小さい事柄から大きな事柄まで巡回符号の性質に触れてゆく．

[*3] 実数は実数体と呼ばれている．

表 5.1　$GF(2)$ について

元の集合	$\{0,1\}$
加算・減算	通常の計算後に mod2 の計算をする
加算・減算の逆元	$-0=0,\ -1=1$
乗算・除算の逆元	$1^{-1}=1$

5.3.1　周期の導入

生成多項式 $G(x)$ に対し

$$(x^n-1) \bmod G(x)=0$$

を満たす最小の n を $G(x)$ の周期という．いきなりの定義だが，この周期は巡回符号の性質を表す重要な数だ．

周期の計算は力技ではあるがあたりをつけることはできる．m 次の既約多項式において，その周期は 2^m-1 の約数となる．さらに，一般の多項式は既約多項式の積として表すことができる．このとき多項式の周期は因数として現れた既約多項式の周期の最小公倍数として与えられる．これらからいかなる多項式の周期も頑張れば計算できる[*4]．

さて，周期 n の $G(x)$ による符号長 n の巡回符号を考える．符号多項式 $W(x)$ は当然だが式 (5.2) を満たす．では以下の計算から出る多項式はどうだろうか．

$$\begin{aligned} xW(x)-w_{n-1}(x^n-1) &= w_{n-2}x^{n-1}+\cdots w_0x+w_{n-1} \\ &= W'(x) \end{aligned}$$

これが同じく式 (5.2) を満たすことは周期の定義から明らかだ．しかし，注目すべきはビット列のほうで，これはある符号語を巡回シフトさせたものだ．つまり，巡回符号は符号語を巡回シフトしたものが符号語となる[*5]．

周期からは他にもわかることがある．符号長が周期 n より大きくした巡回符号では，$W(x), W(x)+(x^n-1)$ の両方が符号語となる．これらのハミング距離は 2 で，最小のハミング距離はこれ以下になる．したがって，この場合の符号では 1 ビットの誤り検出しかできないことが分かる．もちろん，通常は符号長は周期以下の値に設定してある．

5.3.2　実際の誤り検出

様々な規格で異なる生成符号による巡回符号が用いられている．詳しくは Wikipedia を見てくれればよい．ここで実際に使用されている 16 次の生成多項式 G_{16} を紹介する．これは CRC-16-CCITT という規格だそうだ．

$$\begin{aligned} G_{16} &= x^{16}+x^{12}+x^5+x \\ &= (x+1)(x^{15}+x^{14}+x^{13}+x^{12}+x^4+x^3+x^2+x^1+x+1) \end{aligned}$$

[*4] 既約多項式の周期は前もって与えられていると考えていい．

[*5] もちろん名前の由来だ．

因数分解をしておいたが，これ以上はできない．これ以上は因数分解できないたこうしきを既約多項式といい生成多項式に重要な要素となる[*6]．既約多項式の周期は次数 n によって与えられ，

$$p = 2^n - 1$$

となる．例に挙げた生成多項式の周期は $p = 2^{15} - 1 = 32767$ だ．符号語長をこの周期より短いものとしたときの最小距離を考えよう．まず，符号長が周期より小さい時点で最小距離が 3 以上だということが分かっている．次に G_{16} には因数に $(x+1)$ が含まれていることに注目すると符号語多項式 $W(x)$ は

$$W(1) = 0 \quad (\because W(x) = (x+1)(x^{15} + x^{14} + x^{13} + x^{12} + x^4 + x^3 + x^2 + x^1 + x + 1)Q(x))$$

となる．これによって符号語のハミング重みは必ず偶数となる[*7]．これによって最小距離は 4 以上と分かる．最後に G_{16} 自体がハミング重み 4 だといえる[*8]．これで最終的に最小距離が 4 だと結論づけられる．

実際に CRC では復号領域半径を 0 に設定して利用している．したがって 3 ビットまでの誤り検出を可能にしている．

5.3.3 特殊な誤り

先の例で挙げた G_{16} による巡回符号は 3 ビットの誤り検出が可能であった．これは十分なのかという話になる．これは符号語長が 3 万を超えるからだ．もちろん符号語長を短くすればいいがこれは効率の面から避けたい．しかし，このような考えは符号語に現れる誤りが符号語の無作為な位置に現れることを無意識に前提にしている．実際に発生にする誤りについての理解が必要だ．

無線通信路では符号語中の l ビットの区間で誤りが頻発する形が多い．これをバースト誤りという．l ビットの区間のみに注目し，誤り多項式 $E(x)$ は $l-1$ 次多項式 $B(x)$ によって

$$E(x) = x^i B(x)$$

と与えられる．生成多項式 $G(x)$ の次数より，$B(x)$ の次数が小さければ，必ず

$$B(x) \mathrm{mod} G(x) \neq 0$$

となる．これで誤り検知ができる．このバースト誤りの訂正能力はひとえに生成多項式の次数で決定する．

5.3.4 ハミング符号と巡回符号

符号長を 7 とし生成多項式を

$$G(x) = x^3 + x + 1$$

として作られた巡回符号は正しく (7,4) ハミング符号だ．ハミング符号は行列からの作成をした．巡回符号の側面を使うとまた異なる方法がある．

この説明の前に原始多項式という多項式の説明が必要だ．m 次の多項式の周期は高々 $2^m - 1$ だ．これを最大周期といったりもする・ちょうど周期が $2^m - 1$ となる m 次の多項式のことを原始多項式という．これは配布資料や教科書に具体的な原始多項式の例を挙げているので各自参照してほしい．

[*6] 重要な要素ではあるが発見が容易というわけではない．全探索によって発見しているそうだ．

[*7] ピント来なければ実際に計算せよ．

[*8] 生成多項式に重みは考えないが，$Q(x), \{Q(x)+1\}$ による符号語のハミング距離は生成多項式の重みとなることはわかってもらえるはずだ．

この原始多項式を生成多項式とし，符号長を周期と同じ $2^m - 1$ として作成した符号をハミング符号というのだ．もちろん 3 次の原始多項式は $x^3 + x + 1$ だ．

第 6 章

代数学の知識

前章でも少しガロア体について触れている．この講義は代数学の講義でないので表面的なことしか扱わないが，それでも巡回符号の理解の助けになるだろう．符号化が数学の恩恵を受けていることを体感したい．

6.1 体の定義

体の定義は集合の元が次の条件[*1]を満たすものだ．

- 加算の交換が可能
- 加算の結合法則が成立
- 加算に零元[*2]がある
- 加算に逆元[*3]がある
- 積算で交換が可能
- 積算で結合法則が成立
- 積算に単位元[*4]が存在
- 零元でないすべての元に対して積算を行うと単位元を返す元が存在
- 分配法則が成立

$GF(2)$ であれば元は 0，1 の 2 つだがこのうち零元は 0，単位元は 1 となる．また逆元はその元自身となっている．

6.2 $GF(2)$ の拡張

拡張といわれてもという話だが，QR コードの標準規格には $GF(2^3)$ のような表記がある．これは何なのか．前章で考えていた $GF(2)$ 拡張したものだ．この拡張というものについてこの節では扱う．

[*1] 数学らしく書けば公理だ．
[*2] 加算での零元は足しても結果が変わらない元のこと．
[*3] 加算の結果が零元となる元のこと．マイナス元とも．
[*4] 積算の結果がそのままとなる元のこと．

6.2.1 具体的に体の拡大

体を拡張するといってもこれは初めてのことではない．気づかぬうちにすでに体の拡大は経験している．例として実数体上の既約多項式 $f(x) = x^2 + 1$ を扱う．$f(x) = 0$ として方程式を解こうとしても解くことはできない．これは判別式からも確認できよう．しかし，すでにわれわれは虚数単位を知っていため虚数体まで考えれば $\pm i$ が解であるとわかる．ここでわれわれは実数体を虚数体へ拡張している．これは虚数単位によって体の拡張がなされたのだ．

この例を $GF(2)$ にも適応したい．$GF(2)$ 上の既約多項式 $g(x) = x^2 + x + 1$ を用意する．$g(x) = 0$ を考えると，これは $g(0) = g(1) = 1$ であることから $GF(2)$ 上の解がないことがわかる．ではこの方程式の解となる数 α を導入してみよう．そうすると 0，1 に加えて α という数が現れたことになる．この 3 数で体を形成するのかは α の累乗が体に含まれるのかを確認することでできる[*5]．すると α^2 を加えれば体を形成することがわかる[*6]．ここから拡大体 $GF(2^2) = \{0, \alpha, \alpha^2, 1\}$ ができる．

6.2.2 拡大体

実数体から複素数体への拡張は 2 次の既約多項式からおこなわれた．これに似せた方法で $GF(2)$ も 2 次の既約多項式の根を元に加えることで $GF(2^2)$ への拡大をおこなった．では一般に $GF(2)$ を $GF(2^m)$ に拡大するのはどのようになるか．これは m 次の既約多項式のこの根を α とし，α の累乗を乗算が閉じるようになるまで元に加え続けることでなされる．ここでの α を $GF(2^m)$ の元素元という．

まとめると体 F 上の既約多項式の根を元に付加し体を形成する操作が体の拡大ということがわかる．

拡大のもととなる体は $GF(2)$ だけではない．素数 q に対して $GF(q)$ を用意すると先と全く同じ方法で拡大体 $GF(q^m)$ へ拡大できる．

6.2.3 拡大体と周期

拡大をするとき，α を用意して，その累乗を元に加える操作をする．これはいくつ元を付加すればよいのか．これに対する答えは $GF(2^m)$ の元の個数が m 個になるようにすることだ．これに関しては $GF(2^4)$ への拡大を原始多項式で行うことで確かめる[*7]．

$GF(2)$ 上 4 次の原始多項式は $g(x) = x^4 + x + 1$ がまず挙げられる[*8]．これは $GF(2)$ 上では根がないので根となる数 α を与える．ここから α の累乗を計算していくのだが

$$\alpha^{2^4-1} = \alpha^{15} = 1$$

より，$\{0, 1, \alpha, \cdots, \alpha^{14}\}$ が $GF(2^4)$ の元であるとわかる．4 次の原始多項式の周期は $m = 2^4 - 1$ であるが，根 α が

$$\alpha^m = 1$$

となる m というのが周期の根本的な意味といえる．

[*5] 下方について閉じているのかを確認しているのだ．

[*6] $\alpha^3 = \alpha^2 \cdot \alpha = (\alpha + 1) \cdot \alpha = 1$ が計算できる．

[*7] 原始多項式は既約多項式だ．

[*8] 1 つの次数の原始多項式は 1 つだけとは限らない．4 次では $x^4 + x^3 + 1$ も原始多項式だ．

これで分かったのは $GF(2^m)$ の元は $GF(2)$ 上の m 次原始多項式の根 α をとることで $\{0, 1, \alpha, \alpha^2, \cdots, \alpha^{2^m-2}\}$ となる．ここで重要なのは m が変われば α も当然変わるということで，すべての $GF(2^m)$ は $GF(2)$ から拡大されて作られるということだ．

6.2.4　拡大体の元

拡大体の元は α の累乗によって与えられるため，α のほかの元は何となく重要度が低いような気がしてしまう．しかしそのようなことはないだろう．というもの，4 次の原始多項式 $x^4 + x + 1$ から根 α を与えたが，これは 4 つの選択肢があるはずだからだ．これに関して次の式を計算してみる．

$$\begin{aligned} M_1(x) &= (x + \alpha)(x + \alpha^2)(x + \alpha^4)(\alpha^8) \\ &= x^4 + x + 1 \end{aligned}$$

つまり，$x^4 + x + 1$ の根は α が決定した瞬間 $\alpha^2, \alpha^4, \alpha^8$ の合わせて 4 つが根となるのだ．逆に言えば $GF(2^4)$ の元素元の候補は 4 つあると考えてもいい．それ以外に根がないことは各自で計算してほしい．

この話がのちに活きてくる．

第 7 章

複数誤りの訂正

今まで登場した符号化は単一誤りの検知 (単一パリティ検査)，単一誤りの訂正 (ハミング符号)，複数誤りの検出 (CRC-16-CCITT) で，まだ複数誤りの訂正に至っていない．これを達成する符号を考える．

7.1　BCH 符号

2 元巡回符号で 2 ビットの誤り訂正が可能だ．2 ビットの誤りを訂正するのだから単純にシンドロームの組み合わせの個数が多くなることを考えればいい．

7.1.1　生成多項式

BCH 符号は生成多項式

$$G(x) = x^8 + x^7 + x^6 + x^4 + 1$$

で作られる符号長 15 の符号語だ．生成多項式は

$$G(x) = (x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1) = M_1(x) \cdot M_3(x)$$

と因数分解される．M_1 は前章に現れた式だ．$GF(2^4)$ の原始元 α と $\alpha^2, \alpha^4, \alpha^8$ を根とする多項式だ．M_3 は次の式で表現される．

$$M_3(x) = (x + \alpha^3)(x + \alpha^6)(x + \alpha^{12})(x + \alpha^9)$$

つまり，M_3 は α^3 を根に持つ多項式だとわかる．原始元の候補が複数あることは述べたがそのすべての選択肢に対して原始元の 3 乗が用意されている．

$$\alpha^3, \quad (\alpha^2)^3 = \alpha^6, \quad (\alpha^4)^3 = \alpha^{12}, \quad (\alpha^8)^3 = \alpha^9$$

ここで符号長 15 の意味についても考えるがこれは G の周期からとっている．

7.1.2 シンドロームの表現

BCH 符号の検査の方法に入る前にシンドロームの表現方法を考える．ここで考えた表現で BCH 符号も検査する．

まず今までのシンドロームの計算は受信多項式 $Y(x)$ を $G(x)$ で割ることでそのあまりの多項式を $S(x)$ とするものだ．

$$\begin{aligned} S(x) &= Y(x) \bmod G(x) \\ &= E(x) \bmod G(x) \\ \Rightarrow Y(x) &= W(E) + E(x) = Q(x)G(x) + S(x) \\ \Rightarrow Y(\alpha) &= E(\alpha) = S(\alpha) \end{aligned}$$

つまりは受信多項式に根を代入すればシンドロームは立ち現れてくるのだ．これは根を代入したときに $G(\alpha) = 0$ となることが効いている．

7.1.3 BCH 符号の検査

シンドロームの表現方法がわかったので，BCH 符号にどのように応用するのかという話になる．生成多項式が $G = M_1 \cdot M_3$ であるからその根は α, α^3 の 2 つだ．$Y(x)$ に対して α, α^3 の 2 つを代入することで $S(\alpha), S(\alpha^3)$ のシンドロームを得る．1 つのシンドロームは高々 3 次の式だが，4 つのシンドロームによって 2^8 この場合を区別できる．これによって 2 ビットの誤りを訂正できる．

まとめると検査方程式は

$$\begin{aligned} W(\alpha^1) &= 0 \\ W(\alpha^3) &= 0 \end{aligned}$$

の 2 つであり，シンドロームは

$$\begin{aligned} S_1 &:= E(\alpha^1) \\ S_3 &:= E(\alpha^3) \end{aligned}$$

となる．

7.1.4 $GF(2^4)$ を再検討

$GF(2^4)$ の原始元が 4 つの選択肢を持つことは先にも述べている．M_3 は 4 つの選択肢のどれが原始元となっても α^3 が根となるように作られている．ではその外にも α^5 が根になるものと α^7 が根になる式などを考えてみる．$\alpha, \alpha^2, \alpha^4, \alpha^8$ を 5 乗したり，7 乗すればいい．

$$M_5(x) = (x+\alpha^5)(x+\alpha^{10}) = x^2+x+1$$
$$M_7(x) = (x+\alpha^7)(x+\alpha^{14})(x+\alpha^{13})(x+\alpha^{11}) = x^4+x^3+1$$

となる．あと忘れないように

$$M_0(x) = x+1$$

を加えておく．そうすると M_0, M_1, M_3, M_5, M_7 の根で $GF(2^4)$ の 0 以外のすべての元が現れたことになる．そして次の計算をしてみる．

$$\begin{aligned} M_0(x)\cdot M_1(x)\cdot M_3(x)\cdot M_5(x)\cdot M_7(x) &= \prod_{i=0}^{14}(x+\alpha^i) \\ &= x^{15}-1 \end{aligned}$$

となる．ここでも 4 次の原始多項式の周期 15 が現れる．

7.1.5 BCH 符号の定義

最後に BCH 符号の定義を示しておく．$GF(2^m)$ の原始元を α とし $0 < d \leq 2^m - 1$ をとる．

$$\alpha^l, \alpha l+1, \cdots \alpha^{l+d+2} \quad (0 < \forall \leq n-1)$$

のすべてを根にもつ最小次数の多項式を生成多項式とするとその符号が BCH 符号となる．

これではあまりよくわからない．もっと簡単にいうと，

$$G(x) = M_1(x)M_3(x)\cdots M_{2e-1}(x)$$

このような生成多項式を用意すると e ビットの誤りを訂正できるのだ，シンドロームの計算は α^i を $1 \leq e$ で代入すればいい．これで何とかわかるようになる．符号長 n は $d \leq n \leq 2^m - 1$ でとればいい．

7.2 RC 符号

BCH 符号は 2 元符号であった．それを非 2 元符号として拡張したい．q 次元符号として考える．$GF(q)$ の元が情報シンボルに対応する．拡張はしない．符号化や詳しい検査方法は割愛する．

RC 符号は CD に利用される．このとき，1 つの符号語は 28 または 32 ビットの情報ビットを持ち，検査ビットは 4 ビットだ．CD の符号訂正能力を示そう．訂正可能とは，誤りを正しい元のビット列に誤り訂正ができるということだ．では内挿可能とは何なのか．内挿とは，誤りの前後のデータからもっともらしいデータを作成し補完するというものだ．そしてこれら 2 つの能力を上回る誤りが発生した場合，音を消すことで対処する．CD で音が飛ぶとはそれほど広範囲に連続した誤りが発生しているということだ．

表 7.1　CD の誤り訂正能力

訂正可能な最大バースト長	約 4000 ビット (ディスク上 2.5 mm)
内挿可能なバースト長	約 12000 ビット (ディスク上 8 mm)

7.3　畳み込み符号

7.3.1　ブロック符号と畳み込み符号

今まで扱ってきた符号はブロック符号化と総称されるものだ．その意味とは，長い情報ビット列を適当な長さに区切り，その 1 つの区切りに対して何らかの計算などすることで符号語を作っていく．そのときの 1 つのまとまりをブロックという．1 つのブロックは 1 つの符号語と対応している．

これに対して畳み込み符号化はブロックに区切りはするが，符号語は 1 つのブロックと対応しているのではない．これには拘束長という数がかかわってくる．例えば拘束長を 3 とすると 1 つの符号語は 3 つのブロックと対応している正確には符号多項式 W_n は X_n, X_{n-1}, X_{n-2} に依存している．このような符号化だと同じ符号化率であればブロック符号より性能が高い．

7.3.2　最尤復号法

送信ビット列からそのビット列になる確率が最も高くなるような元の送信ビット列を推定するのが最尤復元法だ．畳み込み符号の復号に用いられ，特にビタビ復号法という方法が多く使われる．

第 8 章

情報量

通信路符号化の最後の内容といえる．エントロピーや情報量などの情報に対してその量というものをはかる物差しを導入し，通信路符号化の限界を考える．

8.1　情報量

ビットの個数が単純に情報量だというのはあまりに幼稚な考えだ．情報の価値をあらわすような量というものを定義するべきだ．

8.1.1 定義

S を M 元情報源をとする．情報源記号が $a_i(i = 1, 2, \cdots, M)$ と表され，各情報源記号は確率 p_i で発生する．すると，情報源記号 a_i を得ることで獲得できる情報量は

$$I(p_i) = -\log_2 p_i \ (\text{bit/symbol}) \tag{8.1}$$

と定義される．単位は対数の体に対応している．

この定義への経緯を軽く説明する．情報量表すの関数として求められる条件は次の通りだ．

- 単調減少関数であること
- $I(p_1 p_2) = I(p_1) + I(p_2)$ が成立すること
- 連続関数であること
- $I(1/2) = 1$ であること

1 つずつその意味について述べてゆく．まず最初の条件の意味するとことは発生確率の小さな情報源は大きな情報量を持っているということだ．次の条件は独立な事象が同時に発生するとき，その確率は 2 つの確立の席として与えられるが，情報量としては和として考えたいことを反映している．3 つ目に関してはたいして述べることはない．最後の条件は以上の条件では任意の定数を用意した関数が出来上がるため，1 点を固定させたに過ぎない．4 つの条件から式 (8.1) が導き出される．

8.1.2 平均情報量

情報源記号の発生確率を変数とした関数が情報量であった．そこから期待値を計算しようと考えるのは必然的な流れだ．平均情報量と呼び以下のように計算される．

$$\begin{aligned} \bar{I} &= E[I(p_i)] \\ &= -\sum_{i=1}^{M} p_i \log_2 p_i \end{aligned}$$

この式は情報源 S の 1 次エントロピーに一致している．また記憶のない情報源ならば，エントロピーとも一致していることになる．エントロピーには平均情報量という意味があったのだ．

8.1.3 エントロピーの最大値

情報量は大きいほうがいいことは当然だ．この評価は平均情報量を意味するエントロピーの大きさで行うことができる．エントロピーが最大となる条件を考えよう．

ラグランジュの未定乗数法を用いる．最大を求める関数は

$$f(\boldsymbol{p}) = -\sum_{i=1}^{M} p_i \log_2 p_i$$

条件となる関数は

$$g(\boldsymbol{p}) = \sum_{i=1}^{M} p_i - 1 = 0$$

となる．ここから

$$\begin{aligned}\frac{\partial}{\partial p_i}(f+\lambda g) &= \frac{\partial}{\partial p_i}\left[-\sum_{i=1}^{M} p_i \log_2 p_i + \lambda\left(\sum_{i=1}^{M} p_i - 1\right)\right] \\ &= -\left(\frac{1}{\ln 2} + \log_2 p_i\right) + \lambda \\ &\Rightarrow -\left(\frac{1}{\ln 2} + \log_2 {p_i}^*\right) + \lambda^* = 0,\ {}^\forall i \\ &\Rightarrow p_i = \frac{1}{M},\ {}^\forall i \\ &\Rightarrow H(S) \leq \log_2 M\end{aligned}$$

となる．最大値がわかると，最小値も気になるが，これはある p_i のみが 1 でそのほかが 0 となるときにエントロピーが 0 となる．これはすぐにわかる．

8.2 確率変数

確率変数は確率基礎論などの講義で聞いたことがあるものだと思うがいまいち何なのかがわからない．この節では確率変数を導入の上，それによってエントロピーを扱ってみたい．

8.2.1 確率変数の導入

目的は数字ではない情報源記号と数字に変換する関数を用意することで，数学的な計算をしやすくしようというもの．情報源記号 a_i によって添字の i を返す関数を X とする．そうすると

$$p_i = \mathbb{P}(X(a_i) = i)$$

という風に数字で確率を表現できる．またこのときの X のカッコは省略されることが多い．

以下の確立を累積分布関数といい関数 F_X で表す．

$$F_X(x) := \mathbb{P}(X \leq x)$$

この $F_X(x)$ が微分できるとき次のように確率密度関数を定義できる．

$$f_X := \frac{\mathrm{d}}{\mathrm{d}x} F_X(x)$$

もし，$F_X(x)$ が微分できないとき[*1]で，なおかつ

$$f_X(x) = \mathbb{P}(X = x)$$

とできるときはこの f_X を確率質量関数という．

[*1] 階段状の関数のときは値が飛ぶ部分で微分ができない．

8.2.2　確率変数とエントロピー

確率変数を導入すると，エントロピーは情報源に対して与えられるものから，確率変数に対して与えられるものへ変化する．

$$H(X) = -\sum_{x=a_1}^{a_M} P_X(x) \log_2 P_X(x)$$

添字の X は情報源を示すためのものだが，明示する必要はあまりなく[*2]省略する．そしてエントロピーの計算は確率変数から眺めると明らかに期待値の計算だ．ここから

$$H(X) = -\mathbb{E}[\log_2 P(x)]$$

と略記することができる．

8.3　相互情報量

誤りがどの程度あるのかわかっている通信路から 10 ビットの系列を得たとき，実際は何ビットの情報量が得られたのかを考える．

8.3.1　イメージ

送信語 X に対するあいまいさは受信語 Y の持つエントロピーによってへる．このとき．受信語と送信語が一緒であれば元のあいまいさは 0 になるわけだが，あらかじめ誤りがあるということがわかっていれば 0 にはならないことがわかる．

受信語には雑音のエントロピーが加えれれ，真の情報の一部は失われる．このような状態を模式的に表したのが図 8.1 だ．これはベン図というわけではない．位置のズレという理解のほうが適当かもしれない．

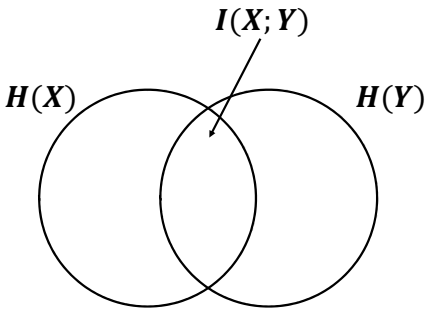

図 8.1　相互情報量の概念図

この図における送信後のあいまいさと，受信語のエントロピーの重なる部分が相互情報量に相当する．このようなイメージで理解を進めよう．

[*2] 明らかだということ．

8.3.2　条件付きエントロピー

相互情報量を求め得るにはその前段階として条件エントロピーの計算が必要だ．

問題をとらえやすくするために問題を設定する．問題は 2 つの区別できる箱 (赤，青) に 0 から 9 までの数字が書かれたカードが重複など考えずに無作為な枚数が入っていするとする．要は赤と青の箱で数字の出現確率が異なる状態を作っている．箱を X，数字を Y として考える．

表 8.1 は問題設定をまとめたものだ．各確率は箱の中での数字の出現確率となっている．つまりは条件付確率だ．確率の計算は

表 8.1　設定された問題 (条件付確率)

数字	赤	青
0	0.2	0.02
1	0.01	0
2	0.03	0.08
⋮	⋮	⋮

$$\mathbb{P}_{赤}(Y=0)=\frac{\text{赤箱内の 0 の個数}}{\text{赤箱内の総数}}=0.2$$

となっている．赤箱，青箱の列それぞれで確率の総和が 1 になる．

表 8.2 は設定を別の見方で表したものだ．確率は結合確率としている．結合確率ということで各マスの確率

表 8.2　設定された問題 (結合確率)

数字	赤	青
0	0.1	0.01
1	0.005	0
2	0.015	0.04
⋮	⋮	⋮

の計算は

$$\mathbb{P}(X=赤\wedge Y=0)=\frac{1}{2}\cdot\frac{\text{赤箱内の 0 の個数}}{\text{赤箱内の総数}}=0.1$$

というような計算をしている．つまり，全確率の総和が 1 となる．

設定を示したところで本題に入る．ここでは出てきた数字からその数字を引いた箱の色を推定する．

この問題を考えるとき一番簡単なのは 1 を引いた時だ．このときは確実に箱が赤であるとわかる．また，ほかにも 0 が出たときも箱は赤ではないだろうか．これは条件付確率で評価できる．

$$\mathbb{P}(X=赤\,|Y=0)=\frac{0.1}{0.1+0.01}\approx 0.91$$

数字が 0 であることが条件の条件付確率だが，これは結合確率の値を利用して求めている．

このような推定を情報量を用いて考える．問題としているのは赤と青の区別なので，そのあいまいさは 1 ビットだ．ここから数字を引いてどれほどのあいまいさが余るのかを条件付エントロピーとして計算する．

ここで数字が 0 であったときの箱の色に関するエントロピー $H(X|Y=0)$ を計算する．

$$\begin{aligned} H(X|Y=0) &= -P_{X|Y}(\text{赤}\,|0)\log_2 P_{X|Y}(\text{赤}\,|0) - P_{X|Y}(\text{青}\,|0)\log_2 P_{X|Y}(\text{青}\,|0) \\ &\approx 0.345 \quad (???) \end{aligned}$$

となる．これは $Y=0$ の場合であったがこれを平均したものが条件付エントロピーだ．$H(X|Y)$ と表記する．

$$\begin{aligned} H(X|Y) &:= \mathbb{E}_Y[H(X|Y=y)] \\ &= \sum_y P(y)H(X|Y=y) \\ &= -\sum_y P(y)\sum_x P(x|y)\log_2 P(x|y) \\ &= -\sum_y\sum_x P(x,y)\log_2 P(x|y) \end{aligned}$$

条件付エントロピーは結合確率と条件付確率から計算できることがわかる．

8.3.3 相互情報量

元のあいまいさから，数字がわかった後の残ったエントロピーである条件付エントロピーを引くことで，数字によって得られた情報量がわかる．これを相互情報量という．相互情報量を $I(X;Y)$ とすると，

$$I(X;Y) = H(X) - H(X|Y) \tag{8.2}$$

となる．これは Y によって得られる X についての情報量だ．計算をいちいちシグマで表現すれば，X,Y を交換することができることがわかる．つまり，

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) = I(Y;X) \tag{8.3}$$

である．そしてここから，Y を知ることで得る X についての情報量と，X を知ることで得られる Y についての情報量が等しいという結論を得る．

相互情報量はシグマ計算で次のようになる．

$$I(X;Y) = \sum_x\sum_y P(x,y)\log_2\frac{P(x,y)}{P(x)P(y)} \tag{8.4}$$

8.3.4 結合エントロピー

結合エントロピーとは X,Y の両方を同時に知ることで得られる平均情報量のことだ．$H(X,Y)$ と表現され次のように計算される．

$$H(X,Y) = -\sum_x\sum_y P(x,y)\log_2 P(x,y) \tag{8.5}$$

$$= H(X) + H(Y|X) = H(Y) + H(X|Y) \tag{8.6}$$

$$= H(X) + H(Y) - I(X;Y) \tag{8.7}$$

8.4　通信路容量

相互情報量の話を箱 (X) と数字 (Y) として進め，最終的には X, Y のみで相互情報量を算出できるまでに至った．この X, Y 通信路の前後での情報ととらえると通信路を通過する情報量を考えることができるのではないか[*3]．

このような考えのもと登場するのが通信路容量だ．

8.4.1　通信路の設定

いきなり通信路容量の説明では理解が困難なので，順を追ってゆく．

通信路の入力に関して，入力アルファベットを用意する．これは入力記号として使用できる集合だ．

$$\mathcal{A} = \{a_1, \cdots, a_r\}$$

とする．これに対して出力記号として利用できる集合の出力アルファベットも同様に用意する．

$$\mathcal{B} = \{b_1, \cdots, b_s\}$$

相互情報量のときと同様で表 8.1 に相当する確率の集合が存在する．通信路においては通信路行列と呼ばれるものとなる．ある入力 a_i が通信路を介して b_j として出力される確率を p_{ij} とすると，通信路行列 T は

$$T = [p_{ij}] \qquad (i = 1, \cdots, r \quad j = 1, \cdots, s)$$

となる．ここで肝心なのは p_{ij} というのが a_i が入力であることを条件とする条件付確率であるということだ．

$$p_{ij} = P(b_j | a_i)$$

ということだ．表 8.1 を踏襲した表 8.3 で表すと表になる．位置関係は通信路行列に対して転置行列のように

表 8.3　通信路行列の要素

	a_1	$\cdots$	a_r
b_1	p_{11}	$\vdots$	p_{r1}
$\vdots$	$\ddots$	$\ddots$	$\vdots$
b_s	p_{1s}	$\cdots$	p_{rs}

なっているが，これは気にしない．

相互情報量のときは次の表は結合確率のものであった．ただ結合確率ではなく a_n の発生確率がわかれば十分であることは条件付確率と結合確率の関係から明らかだ．さらに，入力アルファベットの発生確率が既知であることはそれほどおかしな初期設定ではない．これを $\boldsymbol{p} = \{p_1, \cdots, p_r\}$ とする．

この $\boldsymbol{p}$ は送信者，すなわち我々が任意に決定できる．つまり $\boldsymbol{p}$ はパラメータだ．

[*3] 通信路を通った後の Y によって通信路を通る前の X についてどの程度の情報が得られたのかとは，$I(X;Y)$ に他ならないだろう．

8.4.2　通信路での相互情報量

通信路によって決定する通信路行列 T と送信者が任意に設定できるパラメータ $\boldsymbol{p}$ から相互情報量を計算する．まずは受信者が b_j を得る確率 q_j を求める．これは送信者が $a_i, (i = 1, \cdots r)$ を送信し，確率行列の要素 $p_{ij}, (j = 1, \cdots s)$ によって b_j となる結合確率を i に対して総和をとることで計算される．

$$q_j = \sum_{i=1}^{r} p_i p_{ij}$$

あとは式 8.4 を使い相互情報量を計算する．ここでいちいち結合確率まで計算する必要がないことに注意する．

$$I(X;Y) = \sum_{i=1}^{r} p_i \sum_{j=1}^{s} p_{ij} \log_2 \frac{p_{ij}}{q_j} \text{ (bit/symbol)}$$

これで相互情報量が計算できた．繰り返しになるが $\boldsymbol{p}$ は送信者側が任意に決定できるパラメータだ．$I(X;Y)$ を最大とするときの $\boldsymbol{p}$ をとり，このときの相互情報量を通信路容量と定義する．

$$C := \max_{\boldsymbol{p}} \{I(X,Y)\} \tag{8.8}$$

8.5　各種通信路

いくつかの種類の通信路に対して通信路容量を考える．

8.6　加法的 2 元通信路

図 8.2 のような通信路だ．送信された情報 X に対して誤り E が加わり，受信側が Y として受け取る．2 元符号なので入出力のアルファベットは 0，1 だ．

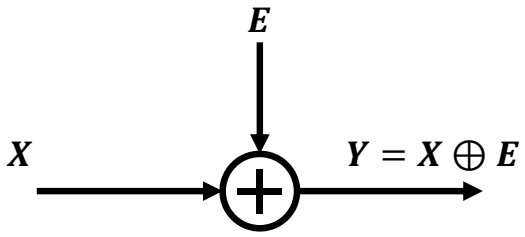

図 8.2　加法的 2 元通信路

このときの相互情報量は次のように計算される．

$$\begin{aligned} I(X;Y) &= H(Y) - H(Y|X) \\ &= H(Y) - H((X \oplus E)|X) \\ &= H(Y) - H(X|X) - H(H|X) \end{aligned}$$

ここで．X, E が独立していることから (H(E—X)=H(E)) であることに注意する．また $H(X|X) = 0$ だ．

したがって

$$I(X;Y) = H(Y) - H(E)$$

となる．ここで $P_X(0) = P_X(1) = 1/2$ ならば，E によらず $P_Y(0) = P_Y(1) = 1/2$ となる．そしてそのとき $H(Y) = 1$ という最大値を与える．以上から加法的 2 元通信路の通信路容量は

$$C = 1 - H(E) \tag{8.9}$$

で与えられる．

8.6.1 2 元対称通信路 (BSC)

通信路が図 8.3 のような概念図としてあらわされる通信路だ．図から直ちに通信路行列 T を得ることができる．

$$T = \begin{bmatrix} 1-p & p \\ p & 1-p \end{bmatrix}$$

言い換えると，確率 p で $E = 1$ となる加法的 2 元通信路だ．したがって，

$$H(E) = -p\log_2 p - (1-p)\log_2(1-p) = \mathcal{H}(p)$$

となる．ここで現れた関数 $\mathcal{H}$ は

$$\mathcal{H}(x) := -x\log_2 x - (1-x)\log_2(1-x)$$

である．

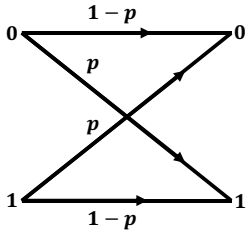

図 8.3　2 元対称通信路

これで BSC の通信路容量を計算できる．

$$C = 1 - \mathcal{H}(p) \tag{8.10}$$

誤りがない状態，つまり $p = 0$ ならば $C = 1$ となり，送信記号と受信記号に関連のない状態である $p = 1/2$

のときは $C = 0$ となる．

第 9 章

通信路符号の限界・現実の通信路

現実の通信路を考えることで，情報理論の後半を通し扱ってきた通信路符号化に対する一定の結論を与えたい．

9.1　通信路の種類

通信路は次の 2 種に分けられる．

離散通信路　アルファベットが離散値の通信路
連続通信路　アルファベットが連続値の通信路

今まで多く取り扱ってきた通信路は離散通信路に分類する．この章でもしばらくは離散通信路を扱う．

9.2　シャノンの通信路容量

ホワイトノイズのある通信路に対する通信路容量を考える．通信路における信号と雑音の電力比を S/N 比という．S/N で計算される．そして通信路の帯域が $0 \sim W$ Hz とする．

このようにすると単位時間 (秒) あたりの通信路容量は次のように計算される．

$$C = W \log_2(1S/N) \text{ bit/s} \tag{9.1}$$

シャノンの通信路容量を情報理論の体系から計算すると

$$W \ln(1 + SNR) \text{ nat/s/通信路} \tag{9.2}$$

となる．これは先の式と同じだ．この公式の導出は以降の講義で扱うのでここでは述べない．

9.3　通信路容量と伝送速度

この節で後半の内容を完結させる．

9.3.1　通信符号化定理

通信路符号化定理

与えられた通信路の通信路容量を C とする．このとき，$R < C$ であれば，任意の正数 ε に対し，復号誤り率 P_e が

$$P_e < \varepsilon$$

を見たす通信速度 R の符号が存在する．しかし，$R > C$ であればそのような符号は存在しない．

この定理が示すのは通信速度の上限だ．これにより，通信路容量を下回る通信速度であれば誤りなく伝送が可能であることがわかる．

9.3.2　情報伝送速度

M 個の符号語によって伝送できる情報量は，各符号語を同じ確率で使用するという条件のもと最大化できる．このときの情報伝送速度 R は次のように計算される．

$$R := \frac{\log_2 M}{n} \tag{9.3}$$

n は平均符号長である．

9.4　情報理論の意義

伝送速度の最大はいくらのなのかという問題に対して答えを与えるのが情報理論だ．データの究極の圧縮を導くのがエントロピーである，最大の伝送速度を与えるのが通信路容量だ．これらの内容を半年で扱った．

第 10 章

情報セキュリティ

情報理論の大枠の話とはそれるが，情報セキュリティに関する講義が 2 回あった．そのうち意味のありそうな部分をまとめる．

10.1　セキュリティに求められる問題

重要な情報を通信する際は，送信元で暗号化を施し，受信側で復号する，暗号化通信がおこなわれる．通信路は公の回線であるため，送信されている途中に悪意ある第三者が影響を及ぼしたり，間違った送り先に送信

されてしまったりしてしまう．このような問題を簡潔にまとめると次の 3 点になるだろう．

- 送信する情報を第三者に盗聴されることなく受信者へ届けたい
- 送信中に情報が改ざんされることなく受信者に届けたい
- 受信者が意図した相手であるのかを確認したい

以上の 3 点を解決する手法をこの章で扱う．

10.2　暗号の世界

10.2.1　アリスとボブ

暗号化を扱うときは情報の送信者や受信者などに特別な名前が与えられている[*1]．

アリス　メッセージ送信者
ボブ　メッセージ受信者
イブ　盗聴者
マロリー　能動的攻撃者

10.2.2　暗号化の構図

暗号化には”鍵”を使う．これは大きな素数によって作られる数字で，これによって送信する情報を一見すると意味の分からない暗号文に変換させる．対して復号化するときも同様に”鍵”を利用する．暗号文に鍵を作用させることで元の情報に変換する．ただし，一般生活で想像する鍵と異なり，暗号化と復号化で鍵が同じであるとは限らない．

10.3　種々の暗号

10.3.1　対称暗号

対称暗号とは，暗号化と復号化で使用する鍵が同じである暗号だ．アリス，ボブが鍵を共有することで情報の送受信を可能にする．

この場合は一方の鍵が流出すると，双方の通信の機密性が担保されなくなる．

10.3.2　公開鍵暗号

公開鍵暗号とは，暗号化に必要な鍵のみをアリスに公開し，受信した暗号文をボブのみが所有する鍵で復号するという暗号だ．この手法はアリスに鍵を与えることから通信が始まる暗号だ．復号は自身の持つ鍵でのみおこなえるので，その鍵が流失しない限り，通信の機密性が担保される．

[*1] 大きな意味があるわけではないが，語句の簡潔化にはなっている．

10.3.3　シーザー暗号

紀元前 100 年ごろ，ローマで用いられた暗号だ．アルファベットをすべて同じ数ずつずらすことで暗号化し，逆にずらすことで復号化した．ここでは文字をずらす数が暗号の鍵だといえる．

さらに発展し，すべての文字を別の文字に対応させて置換する単一換字暗号というものある．この暗号はシーザー暗号の鍵が高々 26 個しかないのに対して，26! 個の鍵の選択肢があるので解読に時間がかかることがわかる．

10.4　公開鍵暗号

先に紹介した公開鍵暗号についてさらに詳しく扱う．

10.4.1　鍵配送問題

紹介の際にも述べた通り，通信はボブがアリスに対して公開鍵を送信することから始まる．ここで最初に問題となるのは，公開鍵の盗聴だ．鍵自体がイブに盗聴される可能性は排除できない．

しかし，公開鍵暗号でアリスへ送信される鍵は暗号化専用の鍵で，復号には使えない．したがって公開鍵をイブに知られてもボブが受け取る暗号文が盗聴されることはない．

10.4.2　RSA アルゴリズム

公開鍵暗号における暗号化用の鍵と復号化用の鍵との対を作成するアルゴリズムだ．アルゴリズム紹介の前に暗号化と復号化の方法を示す．

$$(\text{暗号文}) = (\text{平文})^E \bmod N$$
$$(\text{平文}) = (\text{暗号文})^D \bmod N$$

ここでの E, N が公開鍵，D, N がプライベート鍵だ．鍵の生成には共通する N が重要となる．

N は 2 つの素数 p, q の積だ．

$$N = p \times q$$

次に残りの E, D を生成する助けとなる数 L を用意する．

$$L = \mathrm{lcm}(p-1, q-1)$$

lcm は最小公倍数を表す．この L から直ちに E, D は生成される．

$$E : \gcd(E, L) = 1$$
$$D : E \times D \bmod L = 1$$

このようにしてプライベート鍵と公開鍵は生成される．

10.5　一方向ハッシュ関数

ハッシュ関数とは，入力値の微小な変化で出力値を大きく変化させる関数だ．そして逆関数は存在しない．このような関数は情報の微小な変化を検知する働きをする．

10.5.1　正真性

情報を保存している際にもセキュリティを脅かす事態が存在する．マロリーによる情報の書き換えだ．一晩経った情報が真の情報であるかは一見してわからないことがある．このような問題に対して，素朴に思いつく解決策はデータの写しをとり，変化がないかを確認する方法だ．しかし，この方法ではデータの大きさによっては実行できないことも考えられる．

このようなときに利用されるのがハッシュ関数だ．データを入力とし，これを保存しておく．正真性を確認するにはもう一度同じハッシュ関数にデータを入力し，出力の比較をおこなえばいい．

10.6　認証

ボブがアリスが真のアリスかを確認する方法について考える．これにかんしては公開鍵暗号での鍵の立ち位置を入れ替えることで実現する．

通常，公開鍵で暗号化を施し，プライベート鍵で復号する．しかし署名を送信する際はこの逆でプライベート鍵による暗号化と，公開鍵による復号をおこなう．これは唯一アリスしか所有しないプライベート鍵による暗号化をボブが確認できるということが本人の確認であるという理解でよい．具体的にはメッセージを送信したり，ハッシュ値を送信する場合があるが，本質はプライベート鍵の所有を根拠に認証するという点だ．