第1章

データの符号化と誤り訂正

～ディジタル・データを正確に送受信するために欠かせない技術～

❖

ある場所から別の場所へ送りたいディジタル・データをそのままの形で送らないで,元のディジタル・データに何らかの加工を施すことをデータの符号化といいます.符号化したデータは特殊な規則を用いることで,送受信の際に一部が欠落したり誤ったりしても元のデータに戻すことができます.

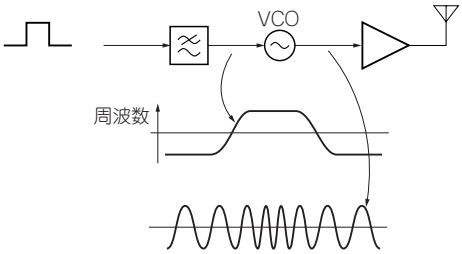
❖

1.1　データの符号化とは

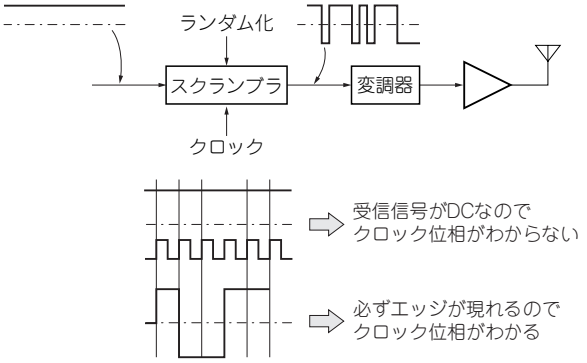
● 通信ではディジタルの情報はアナログ量に変換される

世の中の情報がディジタル化されると，それをそのまま遠隔地や多くの人に伝えることが容易にできるようになります．その場合，電話回線や無線，光ファイバなどさまざまな通信回線が使われますが，これは基本的にアナログの世界です．電流や電波，光などの信号を伝える媒体はディジタルではありません．連続した物理量が変化する中で情報を送るには，1と0からなるディジタルの情報をアナログ量に変換する必要があります．このように，不連続な数値を連続な数値に変換する装置をモデムと呼んでいます．

たとえば，ここでずっと‘1’が続くようなデータをアナログ通信で送ることを考



［図1-1］
FSK変調



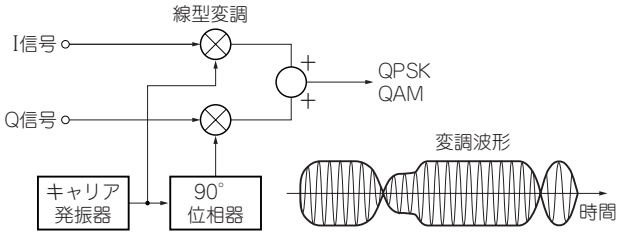
[図1-2]
データのスクランブラ

えます．**図1-1**のように，入力データの変化によってキャリア(搬送波：情報を乗せるための信号)の周波数を変えるFM(Frequency Modulation)変調方式でデータを送るとしましょう．そうすると，このような'1'が連続して続くデータを周波数偏移変調(FSK，Frequency Shift Keying)で送ると，変調のかかっていない連続な正弦波の周波数が$+\Delta F$だけずれた信号を送ることになります．

そのとき受信側では，周波数がずれて'1'が送られていることはわかるかもしれませんが，1ビットの区切りを示すクロックは受信信号から推測することはできません．したがって，一般的には**図1-2**のようにFSKをかける前にランダム化を行い，受け側でクロックが正確に再生できるようにする必要があります．

● ディジタル・データに加工を施すことが符号化

このように，送りたいディジタル・データをそのままの形で送らないで，元のディジタル・データに何らかの加工を施すことを符号化と呼びます．上記のように，通信回線の変調方式に合わせ，それに都合がよいようにデータを符号化することが一般的に行われています．たとえば，**図1-3**に示すQAM(Quadrature Amplitude Modulation：直交振幅変調)では，1と0の元のデータをコンスタレーション



[図1-3]
直交変調器

第2章

符号化に必要な数学の知識

～有限体や既約多項式，生成マトリクスを理解～

❖

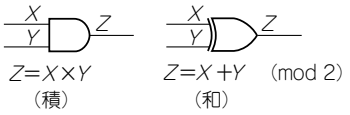
誤り訂正符号を実装するためには，ある程度の数学の知識とその表現を用いて説明することは避けられません．本章では，符号化に必要な数学の知識について説明します．実際の応用では，数式が表す意味を理解することが重要です．数式を眺めただけで，頭の中に現実の応用と結び付くイメージが浮かぶようになることが肝心です．

❖

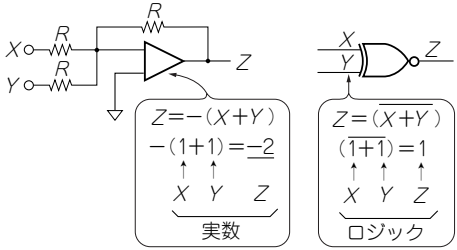
2.1　符号化を行う計算の枠組み

● 符号は‘1’と‘0’の2元数値で構成される

ディジタル符号は‘1’と‘0’のディジタル信号を扱うので，2元数値が基本になっています．符号化を考えるには，そのようなONとOFFの二つの状態しかとらない信号同士の演算を定義することから始まります．二つのディジタル信号があるとき，**図2-1**のような見慣れた論理演算回路を考えます．この回路では，‘1’/‘0’の値を取るXとYの2信号が入力され，‘1’/‘0’のZが出力されます．当たり前のようですが，OPアンプを使うような回路で同じような演算を行う場合は，‘1’/‘0’の出



［図2-1］‘1’/‘0’の論理演算回路



［図2-2］‘1’/‘0’のアナログ加算

力にはなりません(**図2-2**).

2.2　普通の数の計算とは異なる符号化の計算

● ブロック符号の基礎となる有限体

ここでは，ブロック符号の基礎となる有限体(ガロア体，Galois Fields)に関する解説を行います．一般的に，有限体はGF(n)と表現します．たとえば，2元数値ならば，GF(2)となります．4ビットからなる数値では，4ビットのすべての組み合わせは16通りなのでGF(16)，もしくはGF(2^4)と表します．

● 符号化での体は「ある数とある数を同じものと見なす」決まりのある世界

はじめに体について説明しましょう．数学の体という概念は，簡単に言うと，数のグループとその四則演算の法則で一つのセットになったものです．逆に，数のグループを与えてその間に四則演算の法則を与えれば，体と呼べます．いわば，数の世界といえるでしょう．私たちは，今後いろいろな数の世界で計算を行うことになります．代数学においては，このように数と法則をひとまとまりにしてよく扱います．

これから関わっていく体はすべて，元からある数の世界に「ある数とある数を同じものと見なす」という決まりを与えて作られた世界です．この「同一視する」ということが，初めて誤り訂正の数学に対面したときに混乱する壁となります．

この体という概念を理解するために，それぞれの体での数の演算の対応関係すべてを見るという方法と合わせて説明します．あわてずゆっくりと理解を深めるとよいでしょう．

2.3　ディジタルの世界における四則演算

● 有限体GF(2)とは

まず，GF(2)という体を作ってみます．これは整数の世界，つまり…，−1，2，0，1，2，3，…の数の集合において「2と0を同じものと見なす」という決まりを与えます．

しかし，このままでは整数の集合において任意の二つの数n，mが等しいか等しくないかを明確に判断できません．そこでGF(2)を，整数の中で「二つの数n，mが等しいのは，n，mの差が2の倍数である」という決まりを与えた世界とします．

第3章 誤りの評価

～誤り訂正の評価で使われる用語の意味を知る～

❖

いろいろな誤り訂正符号を設計するには，その符号語の訂正能力を知らなければなりません．具体的には，符号のうち何ビットまで誤っても，完全に元に戻せるかという指標をいいます．できるだけ付加するデータを少なくして，より多くの誤りを訂正できる符号がよい符号と解釈できます．本章では，誤りの評価に使われるさまざまな用語について説明します．

❖

3.1 ハミング距離

● ハミング距離とハミング重み

あるGF(2)の元を係数に持つ$n-1$次以下の多項式の集合$R(n)$があって，その中の任意の二つの元Y，Zを選びます．もちろんY，Zは，それぞれ'1'，'0'のnビットの並びです．Y，Zそれぞれの$n-1$次多項式のn個の係数(nビット)のうち，同じ次数に対応するd個の位置の係数がお互いに異なるとします．

もし，Yがノイズのある伝送路で送られて，受信側で誤りが発生していたとしましょう．その際，YがZに間違われるためには，d個のビットが反転しなければなりません．このdをハミング距離といいます．

GF(2)の世界で，dは$R(n)$の中でYとZの距離を表します．dが大きければ，誤りがあってもお互いに間違われる率は減少します．多項式の0でないGF(2)係数の数を数える関数をweight(X)とすれば，ハミング距離d_Hは，

$$d_H = \mathrm{weight}(Y+Z) \quad \cdots\cdots (3\text{-}1)$$

となります．もし，特別にZがゼロであった場合，ハミング距離は，Y自身の1の係数の数になります．すなわち，

$$d_H = \mathrm{weight}(Y) \quad \cdots\cdots (3\text{-}2)$$

このd_Hを特別にYのハミング重みと呼びます.

● ハミング距離の具体的な概念

$$\text{weight}(Y+Z)=2t+1=d \quad \cdots\cdots (3\text{-}3)$$

としたとき，YとZの関係を**図3-2**に示します.

Yを伝送し，伝送路でのあやまりをk個とします．$R(n)$上の元Yから，k個以下誤った場合に考えられるすべての元の部分集合をEとし，**図3-1**の斜線部分で示します．Eの任意の元Wと，WとY，そしてWとZのハミング距離をそれぞれ計算します.

$$\text{weight}(Y+W)\leqq k$$

$$\text{weight}(Z+W)\geqq d-k \quad \cdots\cdots (3\text{-}4)$$

この場合，もし$k=t$ならば

$$\text{weight}(Y+W)\leqq k=t$$

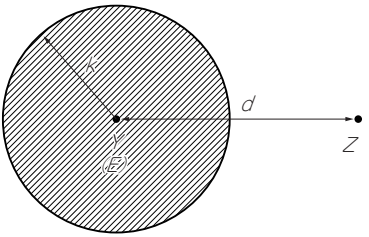
$$\text{weight}(Z+W)\geqq d-k=2t+1-t=t+1 \quad \cdots\cdots (3\text{-}5)$$

受信側であらかじめYとZのハミング距離がd以上だとわかっています．WはYの方にハミング距離的には近いので，WはもとはYだと推測できます．もし，$k=t+1$ならば,

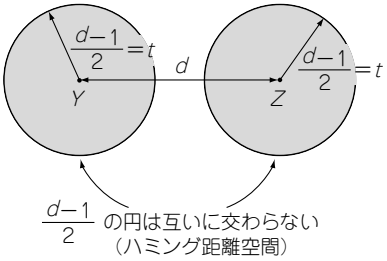
$$\text{weight}(Y+W)\leqq k=t+1$$

$$\text{weight}(Z+W)\geqq d-k=2t+1-(t+1)=t \quad \cdots\cdots (3\text{-}6)$$

となり，今度はWはZだったと間違って判断する可能性が出てきます．すなわち，**図3-2**に示すように，ハミング距離空間でYとZを中心にハミング距離tで描いた円が重なっていなかった場合のみ，WがYの円内ならば，受信側ではハミング距離的にWはYであると判断できます．$k=t+1$の場合は，**図3-7**に示すようなYとZの円に重なりがでます.



［図3-1］k個のエラーの集合



［図3-2］ハミング距離とエラー訂正

第4章

誤りの検出

～正確にデータを伝送するために必要な誤り検出の方法～

❖

誤りフリー（誤りなし）でデータを伝送するには，FEC（前方誤り訂正）やARQ（自動再送要求）が使われます．ARQ方式には誤りの検出が必要になります．そこで，本章ではシンプルなRS-232-Cでのパリティ検査や，一般的に使われるCRC7やCRC16というような，巡回冗長検査CRCの生成や検査法について解説します．

❖

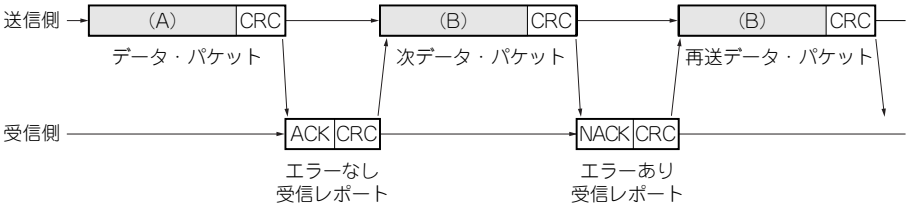
4.1　正確なデータを伝送するには

● 誤りフリーでデータを伝送するためのFECとARQ

伝送の最終的な目的は，受信側に誤りフリー（誤りなし）で送信側と同じデータを伝送することです．それには大きく分けて2通りの方法が考えられます．一つは，誤り訂正を行う方法で，これはFEC（Forward Error Correction）と呼ばれます．もう一つは，受信側で誤りが検出されれば何度でも同じデータを送り返してもらう方法です．この方式は実際にはさまざまな分野で使われています．

たとえば，X.25プロトコルはデータ・パケット伝送にHDLC（High-level Data Link Control）を使い，誤りを検出して，誤りがあれば再送要求を出すことにより誤りフリーを実現しています．このような高信頼性の通信方式をARQ（Automatic Repeat reQuest）と呼んでいます．ARQ方式のデータのやり取りを**図4-1**に示します．受信側からは，受け取った信号に誤りがなければACKを，誤りがあればNACKを送って再送要求をしています．その信頼性を高めるためには，いかに正確にデータ・パケットが正しいかどうかを判断できるかが鍵です．

FECを選ぶかARQを選ぶかは，状況次第です．たとえば，デジタル・テレビ放送のように，個々の受信機から再送要求を受け付けられないような状況では，FEC



［図4-1］ARQ方式

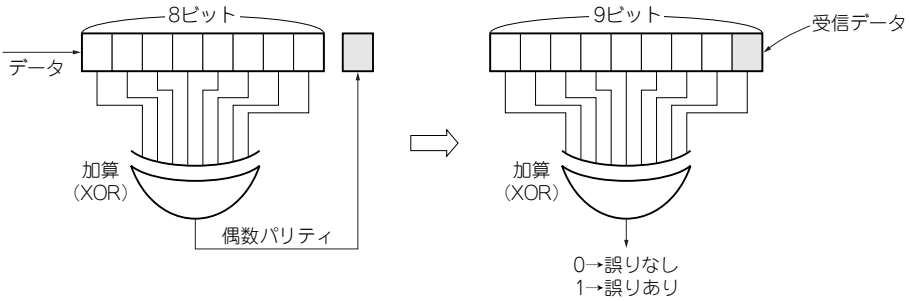
に頼るしかありません．1対1の伝送で比較的伝送回線の品位が高い場合は，ARQを選んだ方が楽です．回線品位が悪い状況でARQのみを選択した場合は，再送要求を際限なく繰り返して一向に終わらない状況が起こるかもしれません．そのときはFECとARQの両方の組み合わせを選択するかもしれません．

4.2　最も簡単なパリティ検査

● 一番シンプルなRS-232-Cでの誤り判断

一番シンプルなのは，第1章で解説したRS-232-Cなどで採用している1ビット付加データ(パリティ・ビット)によって誤りがあるかどうかを判断する方法です．情報が8ビットの場合，符号語を9ビットとします．そして，符号語のハミング重みが偶数，または奇数のみになるように最後のパリティ・ビットを決めるものです．9ビットすべての組み合わせ512通りのうち，半分の256通りしか符号として使わない約束で送り，誤りがあるかどうかを判断するものです．

図4-2に，符号の生成方法と誤り検出方法を示します．たとえば，偶数パリティの場合，符号化では9ビットの'1'の数が偶数になるようにパリティ・ビットを



［図4-2］(9，8)符号の生成と誤り検出

第5章

ブロック符号の基本

～1個の誤りを訂正できるシンプルなハミング符号の仕組み～

❖

CRCは，複数個の誤りを検出することを目的としました．第5章から第7章では，誤りを検出し誤り訂正を行う代表的なブロック符号について解説します．ターボ符号などの高性能なものもありますが，シンプルなブロック符号には，現在でも多くの応用例があります．なにより，畳み込み符号とは違って，誤りがブロック内でとどまり，次のブロックへ判断ミスが感染しないことが特徴です．本章では，ブロック符号の基本であるハミング符号について解説します．

❖

5.1　ハミング符号とは

● ハミング符号は1個の誤りの訂正と2個以下の誤りを検出できる

第2章で説明したハミング符号は1個の誤りを訂正でき，2個以下の誤りを検出できます．すなわち，符号語間のハミング距離は3以上あるといえます．また，シンドロームで表現できる異なる誤りの状態をすべて使い切っているので，完全符号でもあります．とても効率の良い符号です．

● ハミング符号の生成と検査

$n-1$次以下のGF(2)係数の多項式の集合$R(n)$(nビットのデータすべての組み合わせ集合)の中で，m次の既約多項式を$G(X)$とします($n=2^m-1$)．

$G(X)$を原始多項式，$G(X)=0$の根の一つをαとしたとき，$G(X)$で作られる有限体GF(2^m)上の符号がハミング符号です．そして，符号の生成多項式が$G(X)$，すなわち原始多項式と一致した符号です．

ここで，符号長を$n=2^m-1$とし，kビット($k=n-m$)の情報多項式を$I(X)$とすると，

$$I(X)=A_{k-1}\cdot X^{n-1}+A_{k-2}\cdot X^{n-2}+\cdots+A_1\cdot X^{m+1}+A_0\cdot X^m \quad \cdots\cdots (5\text{-}1)$$

(A_iはGF(2)の元)

$I(X)$を$G(X)$で割った余りを$M(X)$とします．$G(X)$がm次の多項式なので，$M(X)$は$m-1$次の多項式になります．

$$M(X)=B_{m-1}\cdot X^{m-1}+B_{m-2}\cdot X^{m-2}+\cdots+B_1\cdot X+B_0 \quad \cdots\cdots (5\text{-}2)$$

(B_iはGF(2)の元)

ハミング符号$F(X)$は，以下のようになります．

$$\begin{aligned}F(X)&=I(X)+M(X)\\&=A_{k-1}\cdot X^{n-1}+A_{k-2}\cdot X^{n-2}+\cdots+A_1\cdot X^{m+1}+A_0\cdot X^m+B_{m-1}\cdot X^{m-1}\\&\quad+\cdots+B_1\cdot X+B_0 \quad \cdots\cdots (5\text{-}3)\end{aligned}$$

係数だけを取り出したビット列で表すと，nビットのデータ列になります．

$$(A_{k-1},\ A_{k-2},\ \cdots,\ B_{m-1},\ B_{m-2},\ \cdots,\ B_1,\ B_0) \quad \cdots\cdots (5\text{-}4)$$

したがって，ハミング符号は，情報ビットがkビット，パリティがmビット，全体でnビットの$(n,\ k)$符号であることがわかります．ここで，$F(\alpha)$を計算すると，

$$F(\alpha)=I(\alpha)+M(\alpha)=M(\alpha)+M(\alpha)=0 \quad \cdots\cdots (5\text{-}5)$$

となり，言い換えれば，$F(X)$は$G(X)$で割り切れます．ここでハミング符号を伝送し，途中でiビット目がノイズで反転した(誤った)とします．このとき誤り多項式を$E(X)$とすると，

$$E(X)=X^i \quad \cdots\cdots (5\text{-}6)$$

1ビットの誤りを含む受信多項式(受信ビット列)は，

$$D(X)=F(X)+E(X) \quad \cdots\cdots (5\text{-}7)$$

となるので，シンドロームを計算します．

$$D(\alpha)=F(\alpha)+E(\alpha)=E(\alpha)=\alpha^i \quad \cdots\cdots (5\text{-}8)$$

となり，0ではなくなります．すなわち，誤りがあることを示しています．GF(2^m)では，0を除く元は$2^m-1=n$個あります．符号長はnビットなので，$0\leqq i<n$となり，シンドロームは誤ったビットの位置の違いで，0以外のすべてのGF(2^m)の元を使い切っています．

言い換えれば，ビット誤りの位置とGF(2^m)の0以外の元は1対1の関係にあり，シンドロームを計算すれば，ビット誤り位置を検出できるということです．その検出した位置のビットを反転すれば，誤り訂正が可能です．ハミング符号は1ビットの誤りを訂正し，mビットのすべてのパリティを使い切った完全符号です．

第6章

BCH符号

～複数の誤りも訂正できるように考えられた巡回符号～

❖

ハミング符号は，符号語間のハミング距離が3以上で誤りを1個しか訂正できません．本来の開発目的であるコンピュータ・メモリの信頼性向上のためには充分ですが，そのほかの信頼性が低い無線伝送路などへの応用では，回線の誤り率に対して誤り訂正能力が充分ではありません．そこで，複数の誤りも訂正できるように考えられた符号の一つにBCH符号があります．

❖

6.1 BCH符号とは

● 2元BCH符号と多元BCH符号

BCH符号は，1959年にHocquenghemによって，また1960年にBoseとChaudhuriによってそれぞれ独立に発見されました．発見者の頭文字をとってBCH符号と呼ばれています．ハミング符号もBCH符号も巡回符号で，同じ仲間だと考えられています．さらに，BCH符号も大きく分けて符号語の多項式の係数がGF(2)の場合と，有限体GF(2^m)のベクトルの場合があります．実践的に使用されているほとんどの巡回ブロック符号はBCH符号の一種で，応用範囲が広い符号です．ここでは，その中の多項式の係数がGF(2)の2元BCH符号について説明します．

GF(2^m)係数の多元BCH符号は，発見者にちなんでリード・ソロモン符号と呼ばれています．リード・ソロモン符号については第7章で説明します．

● BCH符号を具体的に考える

$n-1$次以下のGF(2)係数の多項式の集合$R(n)$（nビットのデータのすべての組み合わせ集合）の中で，m次の既約多項式を$G(X)$とします．$G(X)$を原始多項式，$G(X)=0$の根の一つをαとしたとき，BCH符号は$G(X)$で作られる有限体GF(2^m)上の

巡回符号です．ハミング符号もこの符号の仲間ですが，一般的に2個以上の誤り訂正能力を持つものをBCH符号と呼びます．符号長は，

$$n=2^m-1 \quad \cdots\cdots (6\text{-}1)$$

となります．前述したように，GF(2^m)の0以外の元は，

$$X^n+1=0 \quad \cdots\cdots (6\text{-}2)$$

の根になっています．またX^n+1はGF(2)上で$G(X)$を含むいくつかの既約多項式に因数分解ができました．それぞれの既約多項式は最小多項式と呼ばれます．$G(X)$の共役根はα，α^2，α^4，…です．そこで，α^3を根として持つ最小多項式を$J(X)$とします．$G(\alpha^3)$は0ではありませんが，$J(\alpha^3)=0$です．

符号の生成多項式を$H(X)$としたとき，

$$H(X)=J(X)\cdot G(X) \quad \cdots\cdots (6\text{-}3)$$

を規定します．$H(X)$で生成される符号は，2個の誤りを訂正し，4個の誤りを検出できる能力をもつBCH符号です．多項式$H(X)$の最高次数をpとします．

kビット$(k=n-p)$の情報多項式を$I(X)$とすると，

$$I(X)=A_{k-1}\cdot X^{n-1}+A_{k-2}\cdot X^{n-2}+\cdots+A_1\cdot X^{p+1}+A_0\cdot X^p$$

$(A_i$はGF(2)の元$)$ $\cdots\cdots (6\text{-}4)$

$I(X)$を$H(X)$で割った余りを$M(X)$とします．$H(X)$がp次の多項式なので，$M(X)$は$p-1$次の多項式になります．

$$M(X)=B_{p-1}\cdot X^{p-1}+B_{p-2}\cdot X^{p-2}+\cdots+B_1\cdot X+B_0$$

$(B_i$はGF(2)の元$)$ $\cdots\cdots (6\text{-}5)$

2個の誤りを訂正するBCH符号$F(X)$は，以下のようになります．

$$\begin{aligned}F(X)&=I(X)+M(X)\\&=A_{k-1}\cdot X^{n-1}+A_{k-2}\cdot X^{n-2}+\cdots+A_1\cdot X^{p+1}+A_0\cdot X^p\\&\quad +B_{p-1}\cdot X^{p-1}+\cdots+B_1\cdot X+B_0 \quad \cdots\cdots (6\text{-}6)\end{aligned}$$

係数だけを取り出したビット列で表すとnビットのデータ列になります．

$$(A_{k-1},\ A_{k-2},\ \cdots,\ B_{p-1},\ B_{p-2},\ \cdots,\ B_1,\ B_0) \quad \cdots\cdots (6\text{-}7)$$

BCH符号は，情報ビットがkビット，パリティがpビット，全体でnビットの(n, k)符号とわかります．$F(X)$は$H(X)$で割り切れます．また，$H(X)=G(X)\cdot J(X)$ですから，$F(X)$は$G(X)$と$J(X)$でそれぞれ割り切れます．$F(X)$は，$G(X)$のイデアルの元，かつ$J(X)$のイデアルの元です．

● BCH符号のシンドローム

ハミング符号は，すべての1個の誤り位置とGF(2^m)の元が1対1に対応した完全

第7章

リード・ソロモン符号

～CDやDVD，QRコードにも利用されている符号の生成～

❖

本章では，リード・ソロモン符号について解説します．2元BCH符号を多元に拡大したリード・ソロモン符号は，多くの機器に利用されています．地上波デジタル放送をはじめ，CDやDVD，QRコードなどにも使われています．バーレカンプ・マッシィ法やForneyアルゴリズム，逐次計算法などを利用して，誤り位置や誤り訂正値を計算します．

❖

7.1　リード・ソロモン符号とは

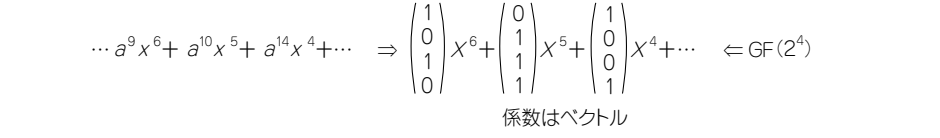
● 2元BCH符号を多元に拡大したリード・ソロモン符号

2元BCH符号では，α，α^2，α^3，…のように生成多項式は連続する$2t$個のα^jを根に持ちました．しかし，そのBCH符号での多項式$F(X)$の係数は，GF(2)の元をとります．そのため，生成多項式の係数もGF(2)の元でなければなりませんでした．そこで，直接必要のない共役根も含めて生成多項式にはαの項が残らないようにする必要があります．これは実数と複素数の関係を考えればわかりやすいと思います．実数係数の多項式の複素数根は，重根を除いて必ず共役複素根のペアでなければなりません．

リード・ソロモン符号は，2元BCH符号を多元に拡大したものです．実数と複素数の関係でいえば，多項式の係数としてそのまま複素数を持つことを許すものです．その場合の変数は，すべて実数2個のベクトルとなります．

● 4ビットを一つのデータ単位とする

2元BCH符号と同じようにリード・ソロモン符号は，**図7-1**のように符号多項式の係数としてGF(2^m)の元がそのまま使われます．つまり，ベクトルを係数に持



［図7-1］ベクトルを係数に持つ符号語

つ多項式です．たとえば，GF(2⁴)の多項式の場合，多項式の係数としてα^jを持ちます．つまり，4ビットで一つの係数を表します．2元BCH符号の場合と異なり，GF(2⁴)の符号は4ビットを一つのデータ単位として扱います．このデータ単位をシンボルと呼んでいます．

もちろん，誤り訂正もその4ビット単位で行われます．もっと一般的にいえば，GF(2^m)のリード・ソロモン符号の場合，係数はmビットのベクトルとなりGF(2^m)の元です．したがって，データもmビットを単位に扱わなければなりません．

そのため，まとまって誤りが発生するバースト誤りには特に有効です．たとえば，コンパクト・ディスク(CD)の傷やごみ，ハード・ディスクのドロップ・アウトなどにより発生する誤りなどです．実際の応用では，リード・ソロモン符号が単独で使われるよりも，畳み込み符号の外符号として組み合わせて使われる場合が多いと思います．畳み込み符号で取り切れなかったバースト誤りに対処するためです．

符号化効率がよいことはわかっていましたが，数学的で直感的にわかりにくいのと，2元BCH符号でも説明したように難解な非線形連立方程式を解かなければならずなかなか実用化されませんでした．ところが前述したピーターソン法などのように実用的な解法が発見されて，一挙に実施例が増えてきました．特に有名なのがCDの符号化への利用です．

7.2 リード・ソロモン符号の生成多項式

● 2元BCH符号から拡張する

t個の誤り訂正を可能にするBCH符号の生成多項式は，連続したαのべき乗で$2t$個の根を持つと述べました．すなわち，

$$G(X)=(X+\alpha)(X+\alpha^2)(X+\alpha^3)\cdots(X+\alpha^{2t})S(X) \quad \cdots\cdots (7\text{-}1)$$

2元BCH符号では，生成多項式の係数がGF(2)の元でなければならないため，$S(X)$の中に共役根を含めなければなりませんでした．しかし，リード・ソロモン符号の生成多項式の係数はGF(2^m)ですから，もはや$S(X)$を掛ける必要はなくなります．t個のシンボル誤り訂正を可能にする生成多項式は，αをGF(2^m)の原始元として単

第8章

Golay符号

～情報ビットが12ビット，パリティ・ビットが11ビットの完全符号～

❖

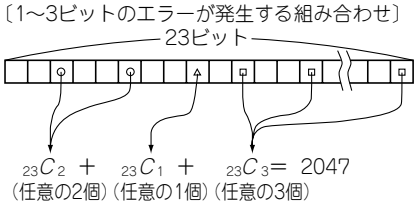
本章では，Golay符号の生成と誤り訂正について説明します．ブロック符号の一つであるGolay符号は，完全符号の一つとして知られています．完全符号である符号化効率を活かして，無線伝送におけるディジタル・システムなどの多くの機器で使われています．Golay符号の符号長は23ビットで，情報ビットは12ビットです．さらに，符号長24ビットの拡張Golay符号もあります．

❖

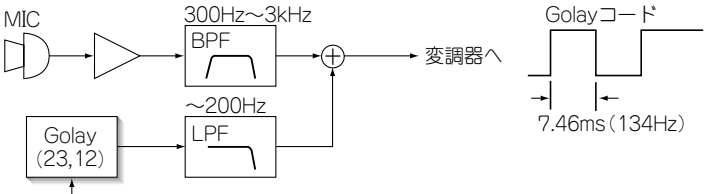
8.1 Golay符号とは

● 多くの機器で採用されているGolay符号

BCH符号の部分集合の巡回符号で，多くの機器で利用されているブロック符号にGolay符号があります．符号長は23ビットで，情報ビットが12ビット，パリティ・ビットが11ビットです(23，12)．BCH符号としての設計符号距離は5で，最大2個の誤りができるはずですが，実際にはそれ以上の3個まで確実に誤り訂正ができる優れものです．しかもパリティ部の11ビットのすべての組み合わせが，すべての3ビットの誤りの組み合わせにユニークに対応しているので，ハミング符号と並んで数少ない完全符号の一つです(図8-1)．



［図8-1］
Golay符号(23，12)



[図8-2]
DCSシステム

Golay符号は，多くの無線伝送におけるディジタル・システムの中で使われています．代表的なものに，**図8-2**のような従来のアナログ・トーン・スケルチをディジタル化したDCS（Digital Code Squelch）システムがあります．DCSは，現在多くの無線機に搭載され標準となっています．送信側のコードと受信側のコードが一致しないと受信しないようなディジタル・スケルチ・システムです．

8.2　Golay符号の生成多項式

● Golay符号の生成多項式の計算

$X^{23}+1=0$の根をβとします．$\beta^{23}=1=\beta^0$となって，23回で巡回します．

ここで，βがどのような有限体GF(2^m)に属するか考えてみましょう．GF(2^m)の原始元をαとします．$n=2^m-1$とすると，$\alpha^n=1=\alpha^0$となり，当然n乗で巡回します．したがって23回で巡回させるためには，

$$2^m-1=23\cdot r \quad \cdots\cdots (8\text{-}1)$$

を計算して，当てはまるmとrを探せばよいのです．

$$2^{11}-1=23\cdot 89 \quad \cdots\cdots (8\text{-}2)$$

と因数分解ができます．すなわち，原始多項式$X^{11}+X^2+1$の根をαとする有限体GF(2^{11})上で，$\beta=\alpha^{89}$とすれば，βはGF(2^{11})上で23回で巡回できます．したがって，βは有限体GF(2^{11})の元と見なせます．GF(2^{11})上の，0および23個のβのべき乗で表せる$\beta^j(0\leqq j<23)$の元は，GF(2^{11})の部分集合を成しています．

Golay符号は，四つの連続する根，β，β^2，β^3，β^4をもつ生成多項式$G(X)$で作られるGF(2^{11})上の巡回符号です．ここで，βの共役根のすべてを考えてみます．

第9章

畳み込み符号

～優れた誤り訂正能力を示すビタビ・アルゴリズム～

❖

畳み込み符号は1955年に提案され，逐次復号法やハーゲルバーガー符号などさまざまな方法が開発されて以来，現在でも無線システムなどに広く利用されています．中でも，1967年にビタビによって発明された実践的な統計的最尤復号法は，優れた誤り訂正能力を示すビタビ・アルゴリズムとして使われています．本章では，このビタビ復号法の符号化と計算方法について解説します．

❖

9.1　畳み込み符号とは

● ブロック符号の限界

前章まではデータ列をブロックに区切り，その区切ったフレーム単位に誤り訂正を行ってきました．ブロックに区切ることで有限体上で計算でき，数学的にはとてもスマートで扱いやすいものでした．理論的には簡単ではないにしても比較的理解しやすいものです．特にどれくらいの誤り耐力があるかは，符号語間の距離で厳密に計算できます．

一方で，数学的に扱いやすい形で誤り訂正をすることでブロック符号の限界も見えてきます．つまり，受信した連続的なアナログ信号を‘1’と‘0’に識別してから，誤り処理を行っている点です．**図9-1**のように識別の段階で，実際はノイズなどで‘1’のところを‘0’に誤ったりします．しかしその状況が，‘1’に近いほんのちょっとした違いで‘0’と間違えたのか，あるいは‘0’の近くでまるっきり間違えたのかは，本当は違う評価をされるべきです．そのような誤りの確率的な扱いが考慮されていないので，直感的にもまだ誤り率を下げられる可能性がありそうです．

言い換えれば受信されるすべての情報をまだ使い切っていないということです．このような統計的な情報を最大に使った復号法を，従来の‘1’，‘0’に判定する硬判

定復号法との比較で『軟判定復号法』と呼ばれます.

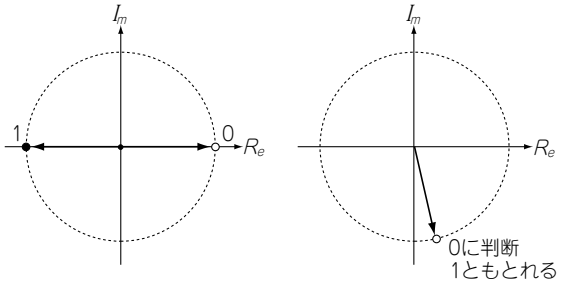
● 統計的な方法で復号する畳み込み符号

ブロック符号とは異なり，データの区切りがなく，1個所のデータをその周辺にばらまき，その情報を使って統計的な方法で復号するのが畳み込み符号です．受信側では，考えられる統計的受信系列の道筋のうち，確率的に最も送信側の符号に距離が近いものを復調データとします．畳み込み符号は，一般的に優れた軟判定復号法と併せて使われます(もちろん，硬判定復号法と組み合わせることも可能).

そこで，近年のデータ通信の多くでは畳み込み符号に注目が集まり，応用されるようになりました．畳み込み符号は，ブロック符号のように構造的に特別の区切りはありません．新しいデータは，符号化器の内部状態の新しい変化の結果として出力されるものです．そのため，過去のデータの並びが現在の受信されるデータに影響します．また，それにより生じる時系列の相関を利用して誤り訂正を行うものです.

特に，復調した信号を '1', '0' に識別する前の連続値をそのまま使うソフト・ディシジョン(軟判定)の復号が主に使われています．もちろん，ブロック符号と同じく '1', '0' に識別したあとに処理を行うハード・ディシジョン(硬判定)の復号器もありますが，性能が落ちるため実用ではあまり使われません．おおよそ2 dBくらいの符号化利得の違いがあると言われています．しかし，ソフト・ディシジョンの復号化器では連続量を扱うため，かなり高い計算能力を必要とします．多くのメモリも必要です.

畳み込み符号自体は，昔から考えられてきました．しかし，解析的な扱いが難しいことと実践的な復号法がないのが課題でした．そのため，優れた特性は期待されましたが，実用化されませんでした．状況を変えたのは，無限に枝分かれする受信



[図9-1]
符号化でのノイズの影響　(**a**) BPSK　ノイズがないとき　(**b**) BPSK　ノイズが多いとき

第10章

符号化に付随する技術

～誤り訂正でよく使われるグレー・コード変換とインターリーバ～

❖

誤り訂正のための符号化技術とは直接関係ありませんが，併せて使うととても効果的な技術があります．本章では，その中のグレー・コード変換とインターリーバを紹介します．実際の応用では，二つともよく使われる重要な技術です．

変調方式と直接かかわっていますが，伝送途中で受けるノイズの種類によって受ける誤りの現れ方に違いがあります．ホワイト・ノイズの場合に，誤り率を下げる方法としてグレー・コード変換が使われます．また，フェージングやブロックひずみの影響を和らげるために，インターリーバが使われます．

❖

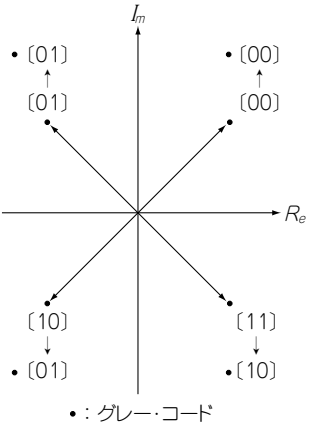
10.1 グレー・コード変換

● ハミング距離的に有効なグレー・コード変換

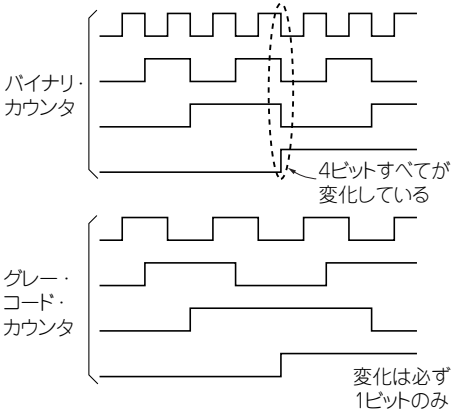
すでに前章の硬判定ビタビ・アルゴリズムでも少し述べました．たとえば，**図10-1**のようなQPSKの変調で，通信路でホワイト・ノイズの影響を受けたとします．コンスタレーションの隣同士の境界までは±45°の余裕があります．ノイズ・レベルが小さいうちは，この範囲に収まるので，間違ってデコードされません．しかしノイズが多くなるとその範囲を超え，隣のデータと間違ってデコードされてしまいます．すなわち，誤りが発生するとすれば，隣のコードと間違う確率が高くなるときといえます．

QSPKの場合，コンスタレーションの位置を図の±45°，±135°の四つの位置とします．これまでの説明では，0°，90°，－90°，180°としてきました．$\pi/4$の位相オフセットが付いていますが，内容はいずれも同じです．それぞれ2ビットの2の補数で表すと，

45°→00，135°→01，－45°→11，－135°→10 ……（10-1）



［図10-1］QPSK変調でのグレー・コード変換



［図10-2］バイナリ・カウンタとグレー・コード・カウンタ

となります.

たとえば，45°のシンボルを送ったとします．ホワイト・ノイズでたまに誤りが発生して，135°または－45°に間違ってデコードされる場合が起こると思われます．その場合に，復調された2ビットの2の補数を考えてみましょう.

送信　　受信

45°　→45°(00)　　135°(01)　－45°(11) ……(10-2)

135°に間違えた場合は00→01となり，2ビットのうち1ビットの伝送誤りが発生しています．ところが－45°に間違われた場合，00→11となり，隣と間違えただけでも2ビットとも間違ってデコードされてしまいます．誤り訂正は，ハミング距離を基準に計算されます．メトリックス計算の際，コンスタレーションの隣同士のハミング距離は1になるべきです.

45°と135°，－45°は隣同士なのでハミング距離は1になるべきです．また，－135°のときに2となるべきです．しかし，バイナリの2の補数表現では，距離が遠いはずの－135°の距離が1で，－45°が2になってしまっています．これを避けるためにグレー・コード(Gray Code)変換が用いられます.

● グレー・コード変換の具体例

バイナリ・カウンタは，図のようにクロック・アップするときに，カウンタの各けたビットの状態の変化数は定まりません．これは，CMOS回路などでは大きな

第11章

ターボ符号

～再帰原理に基づく符号とその計算方法～

❖

データ伝送の理論上の限界値はシャノンによって計算され，それに基づいて誤り訂正が研究されてきました．この理論限界に挑む符号化法はなかなか発見されませんでしたが，近年になって再帰原理に基づく符号化に注目が集まり，今後主流になると予測されています．シャノン限界ではデータの拘束長は無限長として計算されていますが，現実には有限長の短い拘束長の符号が使われてきました．これが限界に近付けなかった原因の一つです．一方，再帰原理に基づく符号は，いずれも拘束長が非常に長くなっています．計算量は増えますが，簡単な符号化とインターリーバの組み合わせで計算量を増やさずに応用が可能となってきました．

❖

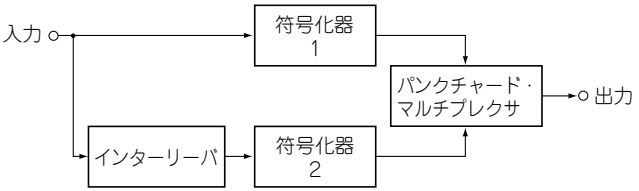
11.1　ターボ符号とは

● データの誤り確率を下げるアルゴリズムとは

畳み込み符号の復号に使われてきたビタビ・アルゴリズムでは，最もありそうな送信パスを探す最尤推定を用います．しかし，最もありそうなパスと，最も誤り確率が低いデータ列とは異なっている場合が考えられます．ビタビ・アルゴリズムでは，途中に誤りが入るとそれによって最尤パスの経路が変わることがあり，真のデータ列パスとの確率的な距離がかえって大きくなる場合が考えられます．この場合は，当然最尤パスを復調データとします．つまり，誤り率が高い方を選択します．

シャノン限界に近付くには，データの誤り確率を下げるようなアルゴリズムを使う必要があると考えられています．そのためには，伝送路の状態によらない最もありそうなデータ列を見つける手段が必要です．すなわち，できるだけデータを広範囲の時系列の中にばら撒き，ランダム化することが求められます．

さらに誤り確率を下げるためには，ほとんど統計的相関のない複数の符号化が有



[図11-1]
ターボ符号化の原理

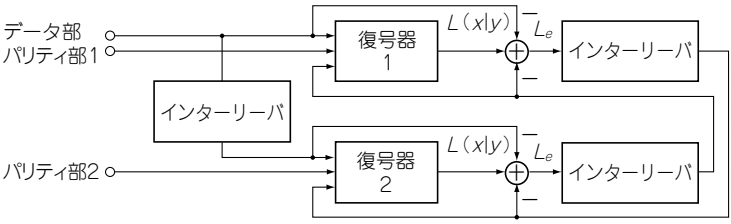
効です．ほとんど相関のない状態で符号化を同時に行い，同じ伝送路を通すと，伝送路で受ける誤りの影響は違う形で現れます．一方で誤り率が低く，もう一方の評価では誤り率が高くなる場合があります．お互いに，それぞれ誤りの確率を下げるように逐次処理を繰り返すと，徐々に合議制で誤り率を下げられます．1回受信されたデータ列はこの繰り返し作業に掛けられ，自動車のターボ・チャージャのようにぐるぐる回って再利用処理されるので，ターボ符号と呼ばれています．

そのために，**図11-1**のように二つ以上のお互いに相関のない推定方法を可能とする符号化方法を行い，それらを同時に送ります．相関をなくす働きは，主に図のインターリーバが担当しています．

● ターボ符号は再帰的な構造で符号化利得を上げる

ターボ符号で用いられるビタビ・アルゴリズムは，畳み込み符号の復号に用いられる普通のビタビ・アルゴリズムと異なり，最終的に推定された最尤パス自体を計算しデコードすることが目的ではありません．一つの復号で得られた結果を使って，繰り返し演算をするための別の復号器の入力データとしなければなりません．そのため，最尤パスに至る各状態での確率に関連した量を計算します．その点が大きく異なります．

軟判定ビタビ・アルゴリズムの結果もデータの確からしさの値で出力するので，すぐにはデコードしません．これをソフトIN-ソフトOUTのビタビ・アルゴリズムと呼んでいます．**図11-2**のように，一つのソフトOUTの結果を別の評価に掛け，



[図11-2]
ターボ復号化の原理