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．

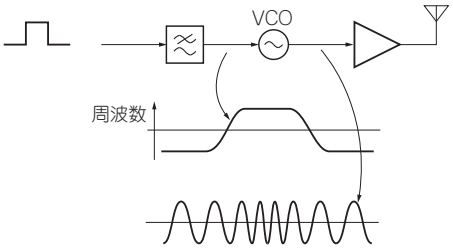
❖

1.1 データの符号化とは

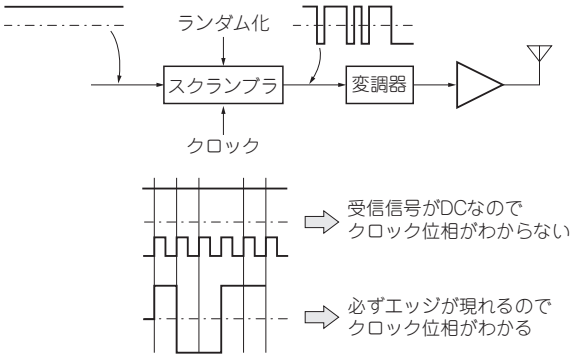
● 通信ではディジタルの情報はアナログ量に変換される

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



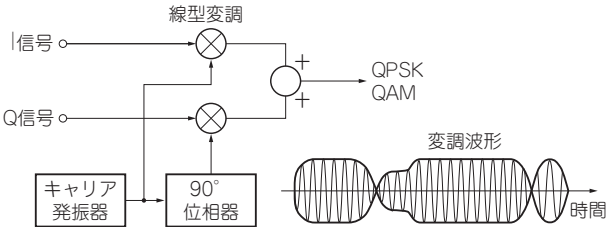
[図1-2]
データのスクランブラ

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＋ΔF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

(constellation)と呼ばれる空間の座標としてデータの符号化が行われています.

たとえば，普段何気なく使っている携帯電話で会話をするときも符号化が行われています．もしかしたら，途中で通話を傍受されるかもしれないという危険を心配する人もいるかもしれませんが，携帯電話による通信では秘話という符号化が行われています．これを使うと，たとえ誰かが傍受して‘1’と‘0’のデータに戻ったとしても，第三者が見ると全く意味のない並びに見えます．ある規則がわかった人だけが元のデータに戻すことができるのです.

これを発展的に考えると，データの符号化で特殊な規則を当てはめれば，途中で一部のデータが欠落したり誤ったりしても，その規則がわかれば元に戻すしくみができます．すなわち，これを誤り訂正のための符号化といいます．ディジタル情報の伝送においては，データの符号化はとても重要な働きをします．優れた符号化を行えば，優れたデータ伝送ができます.

符号化は，数学でいえば関数のようなものです．入力をXとして，出力をYとすれば．変調は$Y=F(X)$と表せます．したがって，データの符号化では，数学をベースに技術論が展開されます．しかも，‘1’と‘0’しか値がない中での数学ですから，これまでの数学とはちょっと違うという予想ができると思います．本書では，データの符号化の中でも特に「誤り訂正」に焦点を当てて解説していきたいと思います.

1.2　誤り訂正とは

● 現代のネットワークを支える誤り訂正技術

インターネットの時代，それを支えているのは通信です．もちろん，アナログの情報ではなく，膨大なディジタルのデータが日々世界中を駆け回っています．たとえば，**図1-4**のように，設計した図面のデータをインターネットを経由して工場まで送るとします．このようなことは，ごく日常の出来事ですが，私たちは相手に全く同じデータが届いていることを疑いません.

しかし，実際にそれがどのような通信回線を通って届いているかはわかりません.

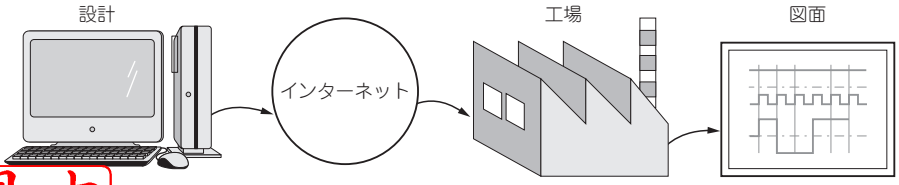


図1-4　インターネットで図面を送る

それがインターネットの本質です．途中で，いくつかの‘1’のデータが‘0’に化けること，あるいはその逆がいかにもありそうです．昔の電話のように，アナログ回線の途中でちょっとノイズが混ざっているというような状況下においては，ディジタル通信では情報が正確に届いているとはいえません．そのようなときのために，受け側で，たとえ間違っても元の完全な形に戻す技術が必要になります．それが誤り訂正です．

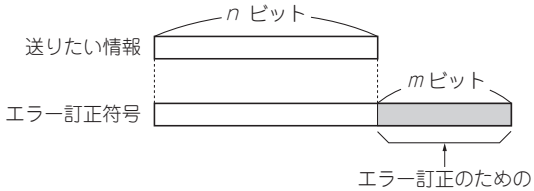
● なぜ誤り訂正が必要なのか

世の中に雑音や外乱というものがなければ，ディジタル符号化において誤り訂正など必要ありません．アナログ・レコードを聞いていると，長年の使用で発生した傷などによりクリック・ノイズなどが発生します．しかし，アナログ・レコードはもともとのレコード溝に刻まれた情報量が多く，ノイズが混ざってもそんなに大きな違和感はありません．レコードが割れでもしない限り，全く音が聞こえなくなるようなことは起こらないでしょう．

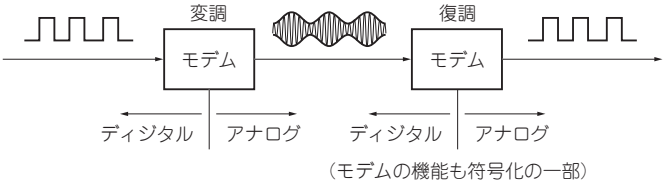
近年，音楽メディアはCD(コンパクト・ディスク)やDVDといったディジタルに完全に切り替わってしまいました．ディジタル処理の場合，‘1’と‘0’の数値だけを使った信号として記録されています．ノイズにより，一個所の‘1’と‘0’を間違えるような誤りが発生すると，全く異なる意味を持つようになり，耳障りな受け入れられないノイズになります．一般的に，ディジタル情報は情報圧縮がかかっている場合も多く，そのまま再生すれば1ビットの誤りでも重大な影響が現れるためです．言い換えれば，アナログのように冗長ではありません．そこで，ディスクに傷が付いても影響がないような技術が要求されます．

● ディジタルの情報に冗長な情報を付加する

ディジタルの特徴を残しつつ，アナログのようなノイズ耐力を付ける優秀な方法を考えなくてはなりません．つまり，アナログ処理のように多少欠陥があっても大きな影響がない，あるいは後で誤り訂正ができるように冗長な情報を付加します(**図**



[図1-5]
冗長な情報としてのエラー訂正符号



［図1-6］
伝送する前後にはモデムが入る

1-5）.

冗長にする方法の一つは，ディジタル・データの圧縮をできるだけやめることです．たとえば，圧縮率が低いデルタ変調では，データが抜けてもそれほど大きな影響はありません．しかし，それではせっかく優れたデータ圧縮の方法があるのに使えないことになります．また，伝送路の帯域幅が広がり，ディジタル化をするメリットが出せません．

この冗長な情報を付加することを符号化と呼び，誤り訂正技術を元に作られます．つまり，受信側で誤り訂正を行うために，本来必要な情報を加工し，そして冗長情報を付加して送ります．また，伝送する前には，図1-6のようにモデムと呼ばれる変調装置が入るのが普通です．送り側の誤り訂正の処理は，この符号化と変調を含みます．

受信側では，復調装置でまず変調された信号を復調します．この‘1’と‘0’に変換する前のアナログ・データには，アナログ・レコードのような貴重なデータが含まれます．したがって，それらも誤り訂正のための重要な情報源になります．また，その後，‘1’，‘0’のデータに復調されたあとで，さらに送り側の符号化の性質を使って誤り訂正を行います．

このように，実際の誤り訂正の技術は単独の技術としては成り立ちません．この変調と復調の方式も含めて総合的に最適な誤りを低減するための方法が符号化と呼ばれるのが一般的です．

このように誤り訂正技術は，影で現代の通信ネットワークを支えていますが，表には出ない地味な存在です．しかし，なくてはならない技術です．

1.3　シャノン限界

● シャノンの通信理論

現在の誤り訂正技術を支えているのは，半導体技術の進歩といってよいでしょう．誤り訂正には，多くの計算を必要とします．しかし，符号理論が初めて論文として

提案されたときは，現在のようにリアルタイムにデータを処理ができる状況ではありませんでした．その中で符号理論の発展を刺激したのは，1948年に発表されたシャノンの通信理論です．

ある帯域幅の通信路の加法的白色ガウス・ノイズ条件（AWGN，Additive White Gaussian Noise）の中のある決められた信号帯域で，信頼性が良く伝送できる情報量の上限を決めたものです（ここで注意しなければならないのは，完全な誤りフリー伝送をいうわけではない）．

信号の帯域幅W（Hz）と受信した信号のS/Nを使うと，情報量C（ビット／秒）は以下の式で与えられます．

$$C = W \cdot \log_2\left(1 + \frac{S}{N}\right) \quad \cdots\cdots (1\text{-}1)$$

これをシャノン限界といいます．たとえば，3 kHzの信号帯域において$S/N = 6$ dB（$= 10^{0.6} = 3.98$）とするとシャノン限界は，

$$C = 3\ \text{kHz} \cdot \log_2(1 + 3.98) = 6.949\ \text{Kビット／秒} \quad \cdots\cdots (1\text{-}2)$$

となります．つまり，3 kHzの信号帯域で$S/N = 6$ dBの中では最大6.949 Kビット／秒の情報を送れます．

また，シャノン限界を使って別の表現もできます．1ビット当たりの信号エネルギー密度E_b，ノイズ・エネルギー密度N_oとすると，次のようになります．

$$\frac{E_b}{N_o} > \log_e 2 = -1.6\ \text{dB} \quad \cdots\cdots (1\text{-}3)$$

つまり，E_b/N_oが最低でも-1.6 dBを超えていなければ，信頼性のある通信はできないということです．

この理論により，将来の可能性がひらけました．理論的にシャノン限界が証明されてから，すさまじい勢いでさまざまな人が競いあって斬新な誤り訂正の方法を研究しました．

● 進化する符号理論

符号理論においては，抽象的な数学を道具にその理論が展開されました．その理論の美しさにのめりこんでいった人も数多くいます．誤り訂正技術にはそのようなある種の数学的な美しさがありますが，これが実務を抱え込む技術者にとっては高い壁となりました．数学者ではない技術者にとって，実際に使いこなせなければ，何も意味がないのです．

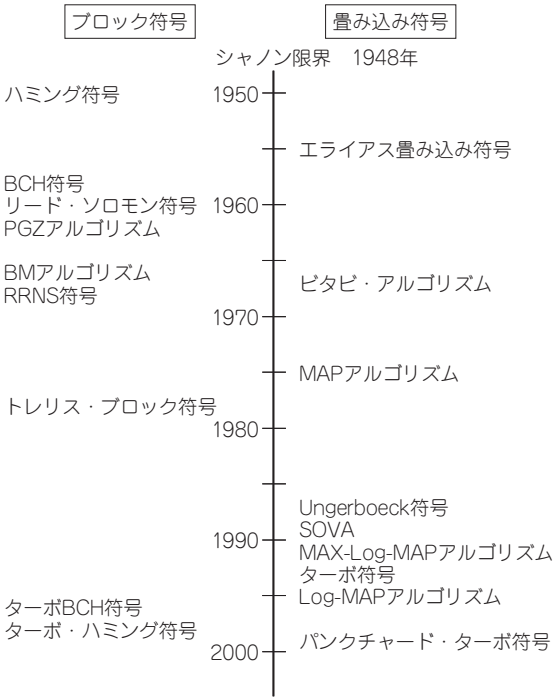
これまでに有益な実用的符号が数多く開発されてきましたが，なかなかシャノン

限界までは到達できませんでした．理論の前提となった，無限長のデータ列が現実と合わないという面も指摘されました．そこで一時期，符号理論はすでに研究が尽くされた技術のように思われていました．しかし，その長い停滞期を過ぎて，近年になってようやく「ターボ原理」を基にしたターボ符号を代表とする新しい符号論が展開されるようになり，ついにシャノン限界に近い性能の符号が現れて注目が集まっています．しかし，今でも基準があくまでシャノン限界にどれくらい近づけるかということは変わっていません．

1.4　符号理論の発展史

● シャノン限界から始まった符号理論

大まかな符号理論の発展の歴史を**図1-7**に示します．後述しますが，符号理論の展開にはブロック符号と畳み込み符号という二つの大きな流れがあります．また，最近はシャノン限界に近づくターボ原理を使った誤り訂正へと大きく発展しています．



[図1-7]
誤り訂正符号の歴史

● ブロック符号として有名なハミング符号

図1-7の最初にあるのは，前述したシャノンの通信理論の論文です．これにより符号理論に弾みがつきました．そして，実用的な1ビットの誤り訂正符号を示したのは有名なハミングです．ハミングは，1950年にブロック符号としてのハミング符号を提案しました．ハミング符号は，実際には当時の信頼性の低いコンピュータの記憶装置の誤り訂正を目的に考えられたものです．また，比較的理解しやすいので，現在でも多くの応用例があります．誤り訂正を勉強するときには，必ず入門として登場します．また，ハミング距離といわれるように，誤り訂正検出符号の幾何学的距離などを使い，その基礎を築きました．

● 畳み込み符号の発展

畳み込み符号は，1955年にエライアスによって提案されましたが，その2年後にはボーゼンクラフトによって逐次復号法が提案されています．1959年には，バースト誤りに強いハーゲルバーガー符号が提案され，今日でもいくつかの無線システムに使われています．しかし，現在の主流となっている，ビタビによって1967年に提案されたビタビ復号法が何といっても重要です．これは実践的な統計的最尤復号法の方法を示したものです．

● 完全符号の一つGolay符号

一方，1949年にハミングより前にブロック符号では応用する際に重要なGolay（ゴレー）符号が発表されています．数少ない完全符号の一つで，符号長23ビット，データ長12ビットという構成で，3個までの誤りを完全に訂正できる符号化です．23ビット長のデータで，3個以下の誤りのすべての組み合わせを計算すると，

$$ {}_{23}C_3 + {}_{23}C_2 + {}_{23}C_1 = 2047 \quad \cdots\cdots (1\text{-}4) $$

の2047通りになります．パリティの長さは11ビットありますから，パリティで表現できる誤りの種類の最大数は，誤りがないゼロを除いて，

$$ 2^{11} - 1 = 2047 \quad \cdots\cdots (1\text{-}5) $$

となります．すなわち，もしパリティが誤りの2047のケースについて1対1という関係を結び付けられれば，全く無駄なくパリティを有効に使い切ることになります．このような符号が完全符号です．

● さまざまな場面で使われる巡回符号

次に，1957年～1960年にかけてプランジらにより巡回符号が示されました．ハ

ミング符号は誤りの訂正はできましたが，1個（ビット）だけなのであまりにも実用するには不十分でした．そこで，1959年ごろに発明されたのが重要なBCH（Bose-Chaudhuri-Hocquenghem）符号です．この符号化はエレガントな理論で複数個の誤り訂正を可能としましたが，使われる道具がとても抽象的な数学であるガロア拡大体（有限体）と呼ばれる特殊な世界で展開されています．さらに，BCH符号は入力は2進数のディジタル・データ列ですが，BCH符号の特殊形としてデータ・シンボル列を複数ビットからなるワード列として拡大した，リード・ソロモン符号が1960年に発明されています．最近では，BCH符号の2次元積符号の復調にも，畳み込み符号のようなターボ原理が応用され，さらなる発展が見られます．

リード・ソロモン符号は，CDで使用されて一躍有名になりました．以来，さまざまな応用に使用されてきています．この証明に使われている数学は美しい代数理論ですが，エンジニアにはとてもとっつきにくいものです．初めての人には，意味不明の“α”が出てきます．現在の誤り訂正の応用では，畳み込み符号とリード・ソロモン符号とを組み合わせて使われる場合が最もポピュラーではないでしょうか．

BCH符号の符号化は比較的簡単ですが，復号はまともに解くと，非線形の連立方程式を有限体の数学空間で扱わなければなりません．この解を実践的に求めるために，1961年ごろPGZ（Peterson-Gorenstein-Zierler）アルゴリズムが発明されました．非線形連立方程式を，誤り位置を求める一つの高次方程式の係数を求める形に展開し，線形行列とベクトルの問題として扱えるようになりました．これで実際の応用が見えてきました．

PGZ法ではBCH符号やリード・ソロモン符号を復号するために，充分見通しの良い方法を提示しましたが，計算量の多いことが難点です．具体的には，リード・ソロモン符号の場合，PGZ法を用いても二つの逆行列を求めなければならないからです．そこで，1965年に逆行列の計算を行わないBM（Berlekamp-Massey）アルゴリズムが発明されました．これにより計算量がぐんと減り，誤り訂正個数が増える場合には特に有効になりました．これらの具体的な計算方法が提案され，リード・ソロモンやBCH符号の実際の応用例が次第に増えていきました．

● シャノン限界に近づくターボ原理

再び畳み込み符号に戻ると，ビタビ・アルゴリズムは確かに強力です．基本は，符号距離的に最も送信系列に近いデータ列を探し出して復号するものです．これにより受信したデータ列に近い，もっともらしい送信データ列は得られますが，必ずしもデータ列の誤り確率自体を小さく抑えることを目指した復号法ではありませ

ん．そこで，誤り確率を最低に抑えることを目指したMAP（Maximum A-Posteriori）アルゴリズムが1974年に発明されました．しかし，従来の畳み込み符号の応用におけるビタビ・アルゴリズムとMAPアルゴリズムの結果の差はわずかでした．その割には，MAPアルゴリズムはとても複雑で，実際の応用で使われることはありませんでした．しかし，この考え方が後のターボ符号で生かされることになります．

QAMのような位相振幅変調は，帯域を広げずに伝送速度が上げられます．このQAMなどの変調に対して有効なトレリス符号化変調の考え方がアンガーベックによって1987年に示され，現在主流の考え方となっています．畳み込み符号は，トレリス符号の中で線形条件を満たすものをいいます．

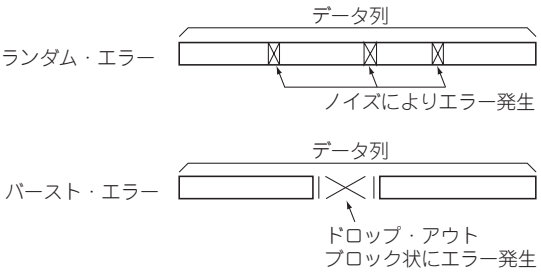
これまで符号理論を引っ張ってきたのは，シャノン限界の理論です．しかし，実際にはこの限界近くまで達成できた符号化はありませんでした．そんな中，1993年にターボ符号がシャノン限界に近づいた符号化として発表されました．同じデータが二つ以上の等価的にお互いに統計的無相関の畳み込み符号器を通ります．受信側では，その無相関の関係を利用し，お互いの誤り率を下げるように繰り返し再帰演算（これが車のターボ・チャージャに似ていたことからの命名）が行われます．演算を繰り返すうちに，誤り率がどんどん下がっていくものです．

このような考え方は「ターボ原理」と呼ばれ，その考え方によりターボ符号だけではなくそのほかのLDPC符号などでも，シャノン限界に近付くような誤り訂正の研究と応用がとても盛んになってきています．

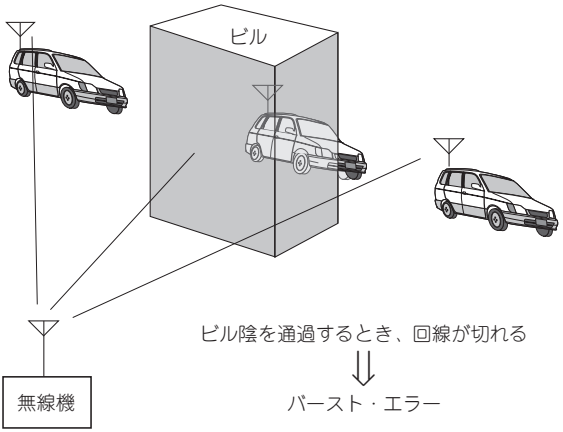
1.5　ランダム誤りとバースト誤り

● ノイズで発生するランダム誤りと障害で発生するバースト誤り

誤り訂正の対象となる誤りは，大きく2種類に分けられます．**図1-8**に示すよう



[図1-8]
ランダム・エラーとバースト・エラー



[図1-9]
バースト・エラー

にランダム誤りとバースト誤りです.

ランダム誤りは白色ガウス・ノイズが伝送中に加わり，ランダムにビットが反転し誤りが発生することが予想されます．すなわち，誤りの発生確率がランダムです．通常の無線によるデータ通信の場合はそれが顕著で，このようなノイズ環境はAWGNと呼ばれています.

一方，図1-9のようにデータの伝送路の途中を障害物が横切った場合などは，データの塊として抜け落ち，バースト誤りが発生します．情報を磁気テープやディスクに記録する場合もドロップ・アウトなどでバースト誤りが頻繁に発生します.

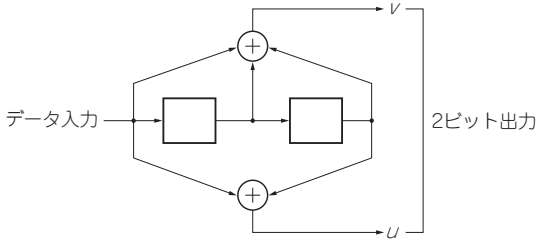
誤り訂正技術を使うときは，これらの誤りの性質と種類にも気を付けて，その方法を選択する必要があります．ランダム誤りに向いた誤り訂正技術をバースト誤りが頻発するところに用いても，信頼性の高い通信を確保できません.

そのほかにもさまざまな性質の外乱がありますが，まずはノイズの種類とその統計的な性質を知ることが肝心です．ただ一般的には，どちらかはっきり分けることは難しく，多かれ少なかれ両方が合わさった誤りが現れるのが普通でしょう.

1.6　ブロック符号と畳み込み符号

● 符号化率を考えることが重要

誤り訂正符号を考えるとき，符号化率というのが重要な要素になります．すなわち，必要なデータに対して，それに関係のない誤り訂正のためのデータをどれだけ付加するかを表す比率です.



[図1-10]
1/2畳み込み符号化器

できるだけ少ない付加データで，できるだけ高い誤り訂正能力を持った符号化が優れた符号化となります．図1-10の畳み込み符号では，1ビットを送るために2ビット使っています．したがって，最終の送信データ列に対して符号化率は1/2です．

● ブロック符号と畳み込み符号の違いは符号長にある

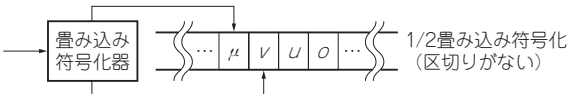
また，実際にどのように付加データを作るかで，ブロック符号と畳み込み符号に大きく分けられます．ブロック符号と畳み込み符号を見て，すぐにわかる大きな違いは符号長です．

- ブロック符号：決まった長さのデータ列(パケット)を基本に誤り訂正する
- 畳み込み符号：長さは決まっておらず，過去からのデータ列の変化過程を元に誤り訂正する

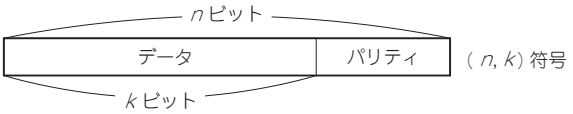
代表的なブロック符号と畳み込み符号を**図1-11**と**図1-12**に示します．

● データ部分とパリティ部分からなるブロック符号

特にブロック符号の場合は，**図1-12**のようにデータ部分と，誤りを訂正するために使う付加データ(パリティ)とに分かれた構造が一般的です(組織的符号と呼ばれます)．極端な言い方をすれば，パリティを無視してデータだけを取り込んでも，



[図1-11]
代表的なブロック符号と畳み込み符号



[図1-12]
組織化ブロック符号

誤りがなければそのまま使えます．ブロック符号の場合，符号化されたデータ・ビット長をn，情報ビット長をkとしたとき(n, k)符号と一般的に表現します．したがって，パリティ・ビット長は$n-k$ビットとなります．

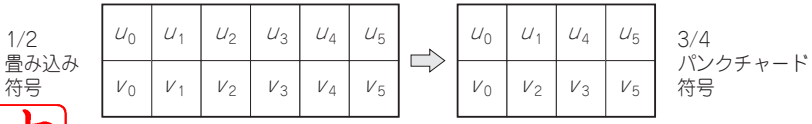
ブロック符号の場合はデータ長が決まっているので，たとえば15ビットの固定長の符号に対して誤り訂正を考えればよいのです．決まった長さを使うということは，符号化に数学を導入するときにとても都合がよい構造です．有限体(ガロア体)と呼ばれる強力な道具を用いることができます．したがって，ブロック符号化では有限体を利用するのが一般的です．

● 過去のデータ列の変化を元に実際に送られる符号が決まる畳み込み符号

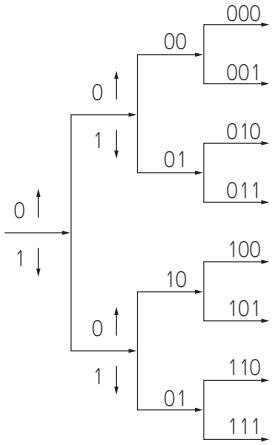
一方，畳み込み符号はデータそのものの痕跡が符号化されたデータ列に残ることは少なく(組織化符号も当然ある)，多くの場合，誤り訂正をして復号しない限りは元のデータはわかりません．言い換えれば，符号そのものではなく符号化器の状態変数を送っているのです．したがって，組織的なブロック符号のように受信したデータ列から元のデータを簡単に推測することは難しいのです．また，**図1-13**に示すように，符号化率を制御するために，受信側で誤り訂正をするのを期待して，定期的にデータを間引きして送ることもあります(パンクチャード符号と呼ばれる)．

畳み込み符号は，過去のデータ列の変化を元に実際に送られる符号が決まります．すなわち，入力されるデータ'1'，'0'によって場合分けがそのつど発生し，**図1-14**に示すように木の枝別れのような道筋をたどります．したがって，畳み込み符号は木符号と呼ばれます．受信側は，受信したデータ列を元に枝分かれしたどのルートを通ってくるのが確率的に一番高いのかを計数し，推定によって復号します．

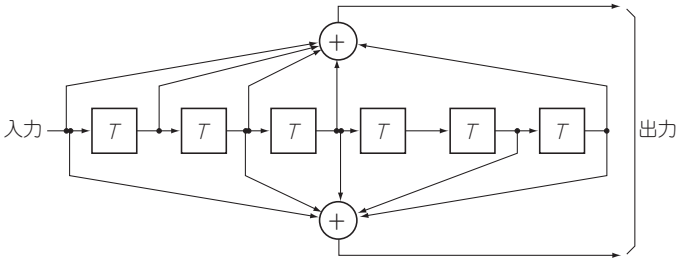
畳み込み符号では，符号化器がどれくらい過去の情報データを使って符号化しているかが，誤り訂正能力の目安の一つになります．図1-15の場合，過去の7ビットを符号化に使っています．したがって，拘束長7の畳み込み符号と呼ばれます．もちろん，現在のデータ列は無数に枝分かれした結果で，その拘束長だけの長さのデータで現時点の符号が決まっているわけではありません．ただし，拘束長で現在の符号が過去のデータにどれくらい影響されているかがわかります．



[図1-13] **パンクチャード符号**(畳み込み符号)



[図1-14]
1/2畳み込み符号化器で生成される木符号

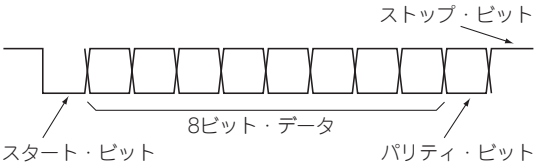


[図1-15]
拘束長7の畳み込み符号

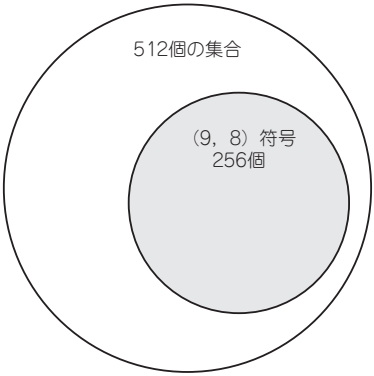
1.7　つまり，符号化とは何か

● RS-232-Cでは1ビットのパリティを加えて符号化する

最も簡単でわかりやすい符号が，**図1-16**に示す情報ビットに1ビット付加する符号化です．たとえば，一般的に使われているRS-232-Cの場合，8ビットのデータの後に，オプションとして1ビットのパリティ・ビットを付加できます．すなわち，符号長は9ビットで(9，8)のブロック符号です．



[図1-16]
RS-232-Cの非同期通信



［図1-17］
(9，8)符号

9ビットの中で必ず1の数が偶数個になるようにパリティ・ビットを決めるのが偶数パリティ，奇数個になるように決めるのが奇数パリティです．言い換えれば，有限な9ビットの1/0のすべての組み合わせ512通りの中で，送るデータを256個の半分のグループ内のデータに限定した符号化です．このようすを図1-17に示します．この符号化は，「送るデータはこの限定されたグループ内のデータだけですよ」と規定していることになります．このようなある規則を持った符号化情報を，一般的に符号語と呼びます．

受信側では，もし受けた符号データが本来あるはずのグループ内の符号語でなければ，伝送途中で，何らかの原因で誤りが発生したと推測できます．ただし，(9, 8)の符号の場合は，誤りがあるかどうかはわかりますが，誤り訂正はできません．この考え方を拡張し，送る符号語をもっと限定されたグループにすれば，誤りまで訂正ができるのではないかと容易に想像できます．

1.8　最少限度の数学は必要

● 符号化の働きを理解するために必要な数学

実践的に符号化をするときは，あまり深く数学を意識する必要はありません．しかし，その働きを理解するためには数学の基礎知識は必須です．でも恐れることはありません．単純な四則演算のみしか使いません．三角関数もなければ，微分積分もありません．ただし，後述しますが，実数の演算とは違って『有限体』と呼ばれる数学体系の中で計算します．

有限体を簡単に説明すれば，決められた有限個の数のグループの中で，四則演算