確率推論に基づく復号法と疎行列に基づく誤り訂正符号

内川, 浩典
株式会社東芝セミコンダクター＆ストレージ社半導体研究開発センター

出版情報：COE Lecture Note. 46, pp.309-318, 2013-02-28. 九州大学マス・フォア・インダストリ研究所
バージョン：
権利関係：

確率推論に基づく復号法と疎行列に基づく誤り訂正符号

内川　浩典

株式会社東芝　セミコンダクター&ストレージ社
半導体研究開発センター

1　はじめに

現在，身の回りにある機器の多くが情報の蓄積や伝送を行うディジタル機器である．そしてそのようなディジタル機器では，記録データを読み出す際や伝送データを受け取る際，データに生じた誤りを訂正するため誤り訂正符号が用いられている．たとえば音楽 CD などで，盤面に少し傷がついたとしても再生可能であるのは，誤り訂正符号のおかげである．

本稿では，誤り訂正符号の中でも理論限界に迫る訂正能力を実用可能な計算量で実現できることから，近年ハードディスク装置やデジタル放送システムなどで実際に製品化が開始されている LDPC (Low Density Parity Check) 符号と，その復号法である sum-product 復号を紹介する[1]．特に確率推論の観点から最適な復号アルゴリズムが，LDPC 符号により実行可能な形に導かれることを述べる．

2　通信システムモデルと誤り訂正符号

誤り訂正符号を議論する際の通信システムモデルは図 1 のように表される．まず，情報を表す長さ k のメッセージ $\mathbf{m} = m_1 m_2 \dots m_k$ は，通信路での誤りに耐えられるよう，符号化処理によって長さ n の符号語 $\mathbf{x} = x_1 x_2 \dots x_n$ へと変換される．ただし $n > k$ である．符号語 $\mathbf{x}$ は，通信路において誤りが加わり受信語 $\mathbf{y} = y_1 y_2 \dots y_n$ となる．ここで通信路は，送信された符号語 $\mathbf{x}$ に対する条件付き確率 $P_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x})$ でモデル化される．受信語 $\mathbf{y}$ は復号処理により，もとの符号語の推定語 $\hat{\mathbf{x}}$ もしくはメッセージの推定語 $\hat{\mathbf{m}}$ へと変換される[2]．なお本稿では議論を簡単にするため，メッセージのシンボル m_i $(1 \le i \le k)$，符号語のシンボル x_i $(1 \le i \le n)$ および受信語のシンボル y_i $(1 \le i \le n)$ はそれぞれ $\{0, 1\}$ いずれかをとるものとし，通信路はシンボルごとに独立かつ同一の条件付き確率

$$P_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{n} P_{Y|X}(y_i|x_i),$$

[1] 本稿は数学セミナー 2012 年 11 月号に掲載した「確率推論による復号と LDPC 符号」に加筆修正をおこなったものである．

[2] 通信システムとしてはもとのメッセージの推定語 $\hat{\mathbf{m}}$ が得られるまでを通信システムモデルとすべきだが，メッセージ $\mathbf{m}$ と符号語 $\mathbf{x}$ とは 1 対 1 対応の関係があるため，符号語の推定語 $\hat{\mathbf{x}}$ が得られるまでを通信システムモデルとすることが多い．

ただし

$$P_{Y|X}(y_i|x_i) = \begin{cases} 1-p & x_i = y_i \\ p & x_i \neq y_i \end{cases}$$

でモデル化された 2 元対称通信路とする．ただし，p は反転確率と呼び，通信路でシンボルが誤る確率をあらわす．

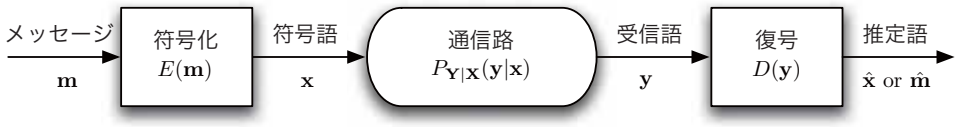

図 1：通信システムモデル

通信の伝送速度を $R = k/n$ と定義したとき，通信路モデルから導出される通信路容量 C を超えない範囲 $R \leq C$ で，誤り率を任意に小さくできる誤り訂正符号が存在することが知られている [1]．そして誤り訂正符号に携わる技術者にとっては，限りなく C に近い伝送速度 R を計算量的に実行可能な符号化復号処理で実現することが，大きな目標となっている．

3　確率推論による復号

誤り訂正符号の復号処理とは，通信路で誤りが生じた受信語 $\mathbf{y}$ から，もっともらしい符号語 $\mathbf{x}$ を推定する推論問題と考えることができる．そしてこの「もっともらしさ」を定量的に扱う道具として，確率を用いる．

いま，送信者が長さ n の符号語 $\mathbf{x} = x_1 x_2 \cdots x_n$ を一様に選んで送信し，通信路を介して受信者が受信語 $\mathbf{y} = y_1 y_2 \cdots y_n$ を受信することとする．このとき，符号語シンボル x_i の誤り率をもっとも小さくする復号アルゴリズムは次のように与えられる．

$$\hat{x}_i = \underset{x_i \in \{0,1\}}{\operatorname{argmax}} P_{X|\mathbf{Y}}(x_i|\mathbf{y}) \tag{1}$$

ただし $\underset{x_i \in \{0,1\}}{\operatorname{argmax}}$ は右項を最大にする引数 x_i を返す関数で，$P_{X|\mathbf{Y}}(x_i = 0|\mathbf{y})$ と $P_{X|\mathbf{Y}}(x_i = 1|\mathbf{y})$ とを比較し，確率値の大きい x_i を推定シンボル $\hat{x}_i$ とする．事後確率と呼ばれる $P_{X|\mathbf{Y}}(x_i|\mathbf{y})$ を最大化するシンボルを推定シンボルとすることから，この復号アルゴリズムは最大事後確率復号[3]と呼ばれる．

[3]送信された符号語 $\mathbf{X}$ の事後確率を最大化する最大事後確率復号と区別するため，シンボル最大事後確率復号や最大事後周辺確率復号と呼ばれることもある．

式 (1) を通信路の条件付き確率 $P_{Y|X}(y|x)$ で計算できる形に式変形すると，次のようになる．

$$\begin{aligned}
\hat{x}_i &= \operatorname*{argmax}_{x_i \in \{0,1\}} P_{X|\mathbf{Y}}(x_i|\mathbf{y}) \\
&= \operatorname*{argmax}_{x_i \in \{0,1\}} \sum_{\sim x_i} P_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}|\mathbf{y}) && (2) \\
&= \operatorname*{argmax}_{x_i \in \{0,1\}} \sum_{\sim x_i} \frac{P_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) P_{\mathbf{X}}(\mathbf{x})}{P_{\mathbf{Y}}(\mathbf{y})} && (3) \\
&= \operatorname*{argmax}_{x_i \in \{0,1\}} \sum_{\sim x_i} P_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) P_{\mathbf{X}}(\mathbf{x}) && (4) \\
&= \operatorname*{argmax}_{x_i \in \{0,1\}} \sum_{\sim x_i} \left(\prod_{j=1}^{n} P_{Y|X}(y_j|x_j) \right) \mathbb{I}[\mathbf{x} \in C] && (5)
\end{aligned}$$

なお，$\sum_{\sim x_i}$ は x_i を除くすべてのシンボルの，すべての値に対する総和記号で，例えば $\sum_{\sim x_1}$ は

$$\sum_{x_2 \in \{0,1\}} \sum_{x_3 \in \{0,1\}} \cdots \sum_{x_n \in \{0,1\}}$$

を表す．式 (2) は周辺確率の計算式[4]から得られ，式 (2) から式 (3) への変形は

$$\begin{aligned}
P_{\mathbf{Y},\mathbf{X}}(\mathbf{y},\mathbf{x}) &= P_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}|\mathbf{y}) P_{\mathbf{Y}}(\mathbf{y}) \\
&= P_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) P_{\mathbf{X}}(\mathbf{x})
\end{aligned}$$

というベイズ則から得られる．そして $P_{\mathbf{Y}}(\mathbf{y})$ が $\operatorname*{argmax}_{x_i \in \{0,1\}}$ の操作に寄与しないことから，式 (4) が得られる．さらに各シンボルの誤り率が独立なことと，各符号語の送信される確率が一様であるという仮定から式 (5) を得た．ただし $\mathbb{I}$[条件] は指示関数で，条件を満たすときには 1 を返し，そうでない場合は 0 を返す．

式 (5) であれば，通信路の条件付き確率から直接計算できるものの，$\sum_{\sim x_i}$ で総和をとる項数が符号の長さ n に対して指数的に増加するため，一定以上の長さの符号に対してこの計算をこのまま実行することは現実的ではない．では効率的に計算するには，どのようにすればよいだろうか？

4　分配則による演算数の削減

前節の式 (5) を見るとわかるように，事後確率の計算は積和演算の形をとる．もし，和の項に共通する因子を和記号の外にくくり出す，つまり分配則が適用できれば，演算数を削減できる（図 2）．

[4]多変数の同時確率から計算される単一変数の確率を，周辺確率と呼ぶ．たとえば 2 変数の同時確率 $P_{AB}(a,b)$ が与えられたとき，確率変数 A の周辺確率は $P_A(a) = \sum_{b \in \mathcal{B}} P_{AB}(a,b)$ により得られる．ただし $\mathcal{B}$ は確率変数 B の定義域を表す．

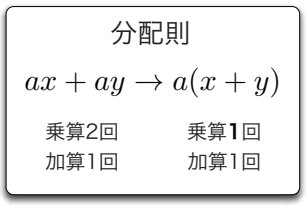

図 2：分配則により，乗算数を 1 つ減らせる例.

分配則により事後確率の計算式が簡単になることを示すため，以下ではパリティ検査行列

$$H_1 = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

で定義される符号

$$C_1 = \{\mathbf{x} \in \{0,1\}^7 | H_1 \mathbf{x}^{\mathsf{T}} = \mathbf{0}\}$$

を用いて，送信シンボル x_1 に対する事後確率の計算式を導く.

まず，式 (5) より，送信シンボル x_1 に対する事後確率は次のように与えられる.

$$P_{X|\mathbf{Y}}(x_1|\mathbf{y}) = \sum_{\sim x_1} \mathbb{I}[H_1 \mathbf{x}^{\mathsf{T}} = \mathbf{0}] \prod_{j=1}^{7} P_{Y|X}(y_j|x_j) \tag{7}$$

なお，$\mathbf{x}^{\mathsf{T}}$ はベクトル $\mathbf{x}$ の転置を表す．ここで，指示関数の条件となっている $H_1 \mathbf{x}^{\mathsf{T}} = \mathbf{0}$ がパリティ検査行列中の各検査式の充足条件の積で記述できることを利用すると，式 (7) は

$$P_{X|\mathbf{Y}}(x_1|\mathbf{y}) = \sum_{\sim x_1} \mathbb{I}_1 \mathbb{I}_2 \mathbb{I}_3 \prod_{j=1}^{7} P_{Y|X}(y_j|x_j)$$

と変形できる．ただし

$$\begin{aligned} \mathbb{I}_1 &= \mathbb{I}[x_1 + x_4 + x_5 = 0], \\ \mathbb{I}_2 &= \mathbb{I}[x_1 + x_2 + x_6 = 0], \\ \mathbb{I}_3 &= \mathbb{I}[x_2 + x_3 + x_7 = 0] \end{aligned}$$

で，指示関数の条件はそれぞれ 1 行目から 3 行目のパリティ検査式に対応している．さらに，

総和に含まれる共通因子と指示関数の条件に着目すると，次のように変形できる．

$$P_{X|\mathbf{Y}}(x_1|\mathbf{y}) = P_{Y|X}(y_1|x_1) \sum_{\sim x_1} \mathbb{I}_1\mathbb{I}_2\mathbb{I}_3 \prod_{j=2}^{7} P_{Y|X}(y_j|x_j) \tag{8}$$

$$= P_{Y|X}(y_1|x_1) \left(\sum_{x_4x_5} \mathbb{I}_1 \prod_{j'\in\{4,5\}} P_{Y|X}(y_{j'}|x_{j'}) \right) \left(\sum_{x_2x_3x_6x_7} \mathbb{I}_2\mathbb{I}_3 \prod_{j''\in\{2,3,6,7\}} P_{Y|X}(y_{j''}|x_{j''}) \right) \tag{9}$$

$$= P_{Y|X}(y_1|x_1) \left(\sum_{x_4x_5} \mathbb{I}_1 \prod_{j'\in\{4,5\}} P_{Y|X}(y_{j'}|x_{j'}) \right) \times \left(\sum_{x_2x_6} \mathbb{I}_2 \prod_{j''\in\{2,6\}} P_{Y|X}(y_{j''}|x_{j''}) \left(\sum_{x_3x_7} \mathbb{I}_3 \prod_{j'''\in\{3,7\}} P_{Y|X}(y_{j'''}|x_{j'''}) \right) \right) \tag{10}$$

なお $\sum_{x_ax_b}$ はシンボル x_a と x_b のすべての要素についての総和記号を表す．例えば $\sum_{x_4x_5}$ は $\sum_{x_4\in\{0,1\}} \sum_{x_5\in\{0,1\}}$ を表す．

まず，すべての和の項に含まれる $P_{Y|X}(y_1|x_1)$ を和記号の外へくくり出すことで，式 (8) が得られる．次に，検査式 $x_1 + x_4 + x_5 = 0$ で変数の組が決まる x_4x_5 をくくり出し，式 (9) が得られる．最後に，残りの検査式に対応する指示関数 $\mathbb{I}_2, \mathbb{I}_3$ ごとに積和をまとめることで，式 (10) を得た．変形前の式 (7) では，乗算の数が 96，加算の数が 14 必要であったのに対し，変形後の式 (10) では，乗算の数が 20，加算の数が 6 となり，演算数の少ない計算式を得ていることがわかる．この例ではもとの式 (7) の演算数がもともと大きくないため，分配則による演算数削減の恩恵が少ないように感じるかもしれないが，変形前の式は符号長が長くなると演算数が指数的に増加し，直接計算することが難しくなる．一方，式 (10) は次節以降で述べる sum-product アルゴリズムを用いることで，非常に効率よく計算することができる．

5　木と sum-product アルゴリズム

前節では，事後確率の計算式が分配則により指示関数ごとの積和式の積へ簡略化できることを，例を用いて説明した．どのようなパリティ検査行列に対しても，上述したような簡略化が適用できれば好ましいのだが，残念ながらこのような式変形はパリティ検査式を 2 部グラフで表現した際に，そのグラフ構造が木になっている場合のみに限られる．

図 3 に式 (6) のパリティ検査式の 2 部グラフ表現を示す．2 部グラフは，符号語シンボルに対応する変数ノード（○）と，パリティ検査式に対応する検査ノード（□）の 2 種類のノードから構成される無向グラフである．変数ノード v と検査ノード c とを接続するエッジ (v, c) は，符号シンボルとパリティ検査式との関係を表す．例えば x_1 に対応する変数ノード v_1 は，1 行目と 2 行目のパリティ検査式に対応する検査ノード c_1 と c_2 に接続されていることがわかる．

グラフ構造が木とは，任意のノード a からその他の任意のノード b への経路が 1 つしかないということを表し，閉路を持たないグラフを意味する．図 3 を見ると，グラフに閉路がなく，

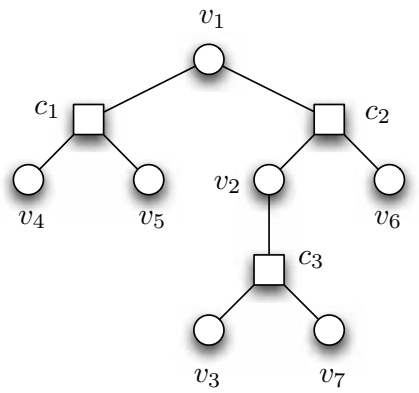

図 3：式 (6) のパリティ検査式の 2 部グラフ表現

木になっていることがわかるだろう．グラフが木の場合，sum-product アルゴリズムにより組織的に事後確率を計算することができる．

sum-product アルゴリズムでは，事後確率を計算したいノード（図 3 では v_1）をルートノードにし，ルートノードからの経路の末端にある葉ノード（図 3 では $v_2, v_3, \ldots, v_7$）からルートノードに向かって，ノードごとに式 (11), (12) の処理を行い，生成したメッセージをルートノードの方向へ送る．そしてルートノードは接続しているすべてのエッジからメッセージを受け取った後，式 (13) の一時推定処理をすることで，事後確率を得る．

変数ノード処理

$$M_{v_j \to c_i}(x_j) = P_{Y|X}(y_j|x_j) \prod_{i' \in \mathcal{I}(j) \backslash \{i\}} M_{c_{i'} \to v_j}(x_j) \tag{11}$$

検査ノード処理

$$M_{c_i \to v_j}(x_j) = \sum_{\sim x_j} \mathbb{I}_i \prod_{j' \in \mathcal{J}(i) \backslash \{j\}} M_{v_{j'} \to c_i}(x_{j'}) \tag{12}$$

一時推定処理

$$g(x_j) = P_{Y|X}(y_j|x_j) \prod_{i \in \mathcal{I}(j)} M_{c_i \to v_j}(x_j) \tag{13}$$

なお，$\mathcal{I}(j)$ は変数ノード v_j に接続する検査ノードのインデックス（シンボル x_j が含まれるパリティ検査式の行インデックス）を表し，$\mathcal{J}(i)$ は検査ノード c_i に接続する変数ノードのインデックス（i 行目のパリティ検査式に含まれるシンボル x_j のインデックス）を表す．また $\mathcal{S} \backslash \{i\}$ は，集合 $\mathcal{S}$ から集合 $\{i\}$ を除いた差集合を表す．

たとえば葉ノード v_3 は変数ノードなので，式 (11) よりメッセージ

$$M_{v_3 \to c_3}(x_3) = P_{Y|X}(y_3|x_3)$$

を生成し，検査ノード c_3 へ送る．すべての葉ノードはエッジが 1 本のため，式 (11) 中のメッセージの積の項を持たない．ここでは紙面の都合によりノード v_1 の計算例のみを示したが，他のノードに対しても同様に計算できる．

6 同時更新型 sum-product 復号

復号の際には，すべてのシンボル x_j $(1 \le j \le n)$ に対して事後確率を計算するが，各シンボルに対応する変数ノードをルートノードとして計算を繰り返すことは，シンボル間に共通するメッセージが存在するため，効率がよくない．そこで通常は，各ノードの処理を同時に行い，収束するまで繰り返し処理をする同時更新型 sum-product アルゴリズムが用いられる．以下では，同時更新型 sum-product アルゴリズムを使った復号法の手順を紹介する．

同時更新型 sum-product 復号アルゴリズム

手順 1 初期化
K に最大繰り返し回数を設定し，k に 1 を代入する．そしてすべての $M_{c_i \to v_{j''}}(x_{j''})$ $(1 \le i \le m,\ j'' \in \mathcal{J}(i),\ x_{j''} \in \{0,1\})$ に 1 を代入する．

手順 2 変数ノード処理
すべての変数ノード v_j $(1 \le j \le n)$ において，変数ノード処理（式 (11)）を実行する．

手順 3 検査ノード処理
すべての検査ノード c_i $(1 \le i \le m)$ において，検査ノード処理（式 (12)）を実行する．ただし，m はパリティ検査行列の行数を表す．

手順 4 パリティ検査処理
すべての変数ノード v_j $(1 \le j \le n)$ において，一時推定処理（式 (13)）を実行する．そしてすべてのシンボル x_j $(1 \le j \le n)$ に対して一時推定シンボル

$$\tilde{x}_j = \begin{cases} 0 & (g(x_j = 0) \ge g(x_j = 1)), \\ 1 & (\text{その他}), \end{cases}$$

を得て，一時推定語 $\tilde{\mathbf{x}} = \tilde{x}_1 \tilde{x}_2 \ldots \tilde{x}_n$ を得る．ここで，$H\tilde{\mathbf{x}}^{\mathsf{T}} = \mathbf{0}$ であれば $\tilde{\mathbf{x}}$ を推定語 $\hat{\mathbf{x}}$ として出力し終了する．その他の場合は**手順 5** へ．

手順 5 終了判定
$k = K$ の場合は $\tilde{\mathbf{x}}$ を推定語 $\hat{\mathbf{x}}$ として出力し終了する．その他の場合には k へ $k+1$ を代入し，**手順 2** へ．

ここまで，2 部グラフが木の場合に限って説明してきたが，実用化されている誤り訂正符号のパリティ検査行列の 2 部グラフは木とはなっていない．2 部グラフが木でない場合，sum-product アルゴリズムで計算した値は事後確率の近似値となるが，任意のノードに対して深さ

6から8程度までの部分グラフ（以下，近傍グラフと呼ぶ）が木になっていれば，良好な復号特性を得られることが経験的に知られている．なお深さとは，任意のノードaをルートにして2部グラフを木のように展開したとき，経路に含まれるエッジの数のことを指す．

では，近傍グラフがなるべく木になるようにパリティ検査行列を構成するには，どうすればよいのだろうか？ 実はその答えが，LDPC符号のパリティ検査行列を特徴づける“疎”の部分にあることを次節で紹介する．

7 LDPC符号のパリティ検査行列

LDPC符号は「疎（低密度）なパリティ検査行列で定義される符号」と定義される．疎なパリティ検査行列とは，パリティ検査行列中に含まれる非零要素，2元の場合には“1”の数が少ないパリティ検査行列を表す．どの程度少ないのかというと，1列に含まれる非零要素の平均個数lが符号長nに依存せず，$2 \leq l \leq 8$が一般的に用いられる．1ビット訂正符号として知られるハミング符号のパリティ検査行列に含まれる非零要素数が，平均で少なくとも$\frac{\log_2 n}{2}$個になることと比較しても，LDPC符号のパリティ検査行列が疎であることがわかる．パリティ検査行列が疎であるということは，対応する2部グラフに含まれるエッジの数が少なくなるため，閉路が生成されにくくなる．つまりLDPC符号のパリティ検査行列は，疎にすることで近傍グラフが木になりやすくなり，sum-productアルゴリズムで得られる事後確率の近似精度を上げることに成功しているのである．

以下では，LDPC符号の発明者であるGallager教授が，彼の博士論文[2]で提案したパリティ検査行列の構成法を紹介する．この構成法は，1列に含まれる非零要素の数がlで，1行に含まれる非零要素の数がrとなるような長さnのパリティ検査行列$H^{(n,l,r)}$を生成する．ここでは説明を簡単にするため，nはrの倍数であるものとする．

まず，サブ行列$H_0^{(n,l,r)}$をつぎのように定義する．

$$H_0^{(n,l,r)} := \begin{bmatrix} \mathbf{h}(r) \\ s^{(r)}(\mathbf{h}(r)) \\ s^{(2r)}(\mathbf{h}(r)) \\ \vdots \\ s^{(n-r)}(\mathbf{h}(r)) \end{bmatrix}$$

ただし$\mathbf{h}(r)$は先頭に非零要素がr個連続でならび，残りはすべて0となる長さnのベクトルである．また$s^{(r)}(\mathbf{h})$は，ベクトル$\mathbf{h}$を右方向にr要素分巡回シフトしたベクトルを返す関数とする．

このサブ行列$H_0^{(n,l,r)}$を使って，パリティ検査行列$H^{(n,l,r)}$が次のように得られる．

$$H^{(n,l,r)} = \begin{bmatrix} H_0^{(n,l,r)} \\ \pi_1(H_0^{(n,l,r)}) \\ \vdots \\ \pi_{l-1}(H_0^{(n,l,r)}) \end{bmatrix} \tag{14}$$

なお，$\pi_i(H_0^{(n,l,r)})$ $(1 \leq i \leq l-1)$ は，サブ行列 $H_0^{(n,l,r)}$ の列ベクトルをランダムに並べ替える関数とする．

例として，上記構成法で生成した $H^{(8,2,4)}$ を示す．

$$H^{(8,2,4)} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$

最後に，2 元対称通信路における LDPC 符号の復号誤り率に関する数値実験結果を図 4 に示す．LDPC 符号のパリティ検査行列は，式 (14) の構成法で生成した 510×1020 のパリティ検査行列で，列と行の非零要素数はそれぞれ $l = 3, r = 6$ である．復号処理は最大繰り返し回数を 500 回とした同時更新型 sum-product 復号を用いた．また比較のため，ランダム誤りに対する誤り訂正符号として一般に用いられる BCH (Bose-Chaudhuri-Hocquenghem) 符号 [3] の結果も図 4 に示す．BCH 符号は長さが 1023 ビットで設計距離 103（訂正可能ビット数 51 ビット）の符号を用い，符号長および伝送速度が評価する LDPC 符号と同等になるようにした．

図 4 をみると，シンボル反転確率が 0.06 のとき，BCH 符号の復号誤り率はほぼ 1 となってしまうのに対し，LDPC 符号の復号誤り率は $\dfrac{1}{100}$ を達成していることがわかる．

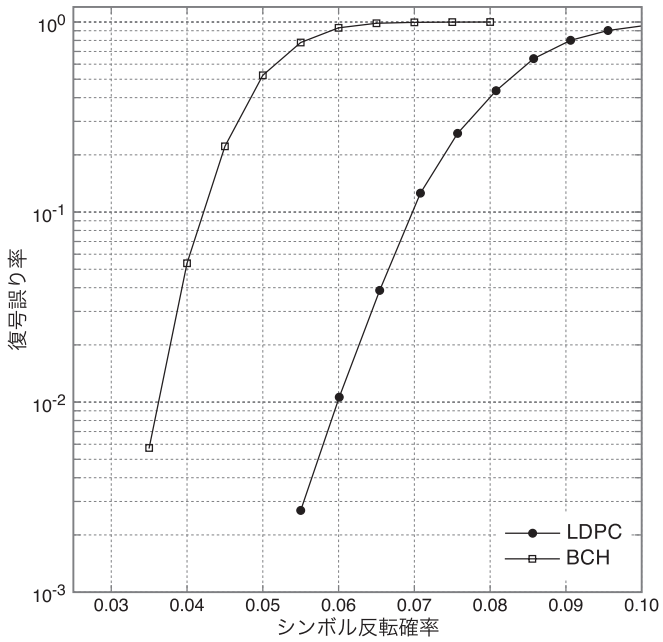

図 4：2 元対称通信路における復号誤り率に関する数値実験結果

8　むすび

本稿では，確率推論の観点から最適な最大事後確率復号が，分配則を用いることで効率良く実行できることを紹介し，具体的な復号アルゴリズムである sum-product 復号を与えた．さらに sum-product 復号によって精度の高い最大事後確率が計算できるパリティ検査行列の構成として，LDPC 符号のような疎なパリティ検査行列が与えられることを述べた．

本稿は直感的な理解が得られるよう，厳密な定理や証明を与えるよりは，例を通してそのエッセンスを紹介した．本稿を読んで LDPC 符号に興味を持たれた方は文献 [4, 5] を参照することで，より深い知識を得ることができるだろう．また，LDPC 符号に関わる深い数理に興味がある方は文献 [6] を参照することをおすすめする．一方，数理的な内容よりも符号の設計手法や復号アルゴリズムなど，実用的なトピックに興味がある方は，文献 [7] が参考になるであろう．

参考文献

[1] C. E. Shannon, “A mathematical theory of communication,” *Bell System Technical Journal*, 1948.

[2] R. G. Gallager, *Low-density parity-check codes*. MIT Press, 1963.

[3] S. Lin and D. J. Costello, *Error Control Coding*. Prentice Hall, 2nd ed., 2004.

[4] 和田山 正，**誤り訂正技術の基礎**．森北出版，2010.

[5] 萩原 学，**符号理論 デジタルコミュニケーションにおける数学**．日本評論社，2012.

[6] T. J. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge University Press, 2008.

[7] W. E. Ryan and S. Lin, *Channel Codes: Classical and Modern*. Cambridge University Press, 2009.