社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.
IT 2001-19 (2001-7)

ブロックターボ符号の生成行列を用いた一復号法

大島 英明　小笠原 尚徳　小林 学　平澤 茂一

早稲田大学 理工学部 経営システム工学科
〒169-8555 東京都新宿区大久保3-4-1
tel.03-5286-3290
fax.03-5273-7215
E-mail: hideaki@hirasa.mgmt.waseda.ac.jp

あらまし　ターボ符号は，1993年にC.Berrouらによって提案された通信路符号である．この符号はターボ復号を用いてシャノン限界に迫る新しい符号化方式として，近年符号理論などの分野で注目を集めている．本稿ではまず積符号型のブロックターボ符号に対して，置換行列をうまく選択することにより従来の積符号より最小距離が大きくなる符号が存在することを示す．次に積符号型ブロックターボ符号に対してターボ復号を数回繰り返し，その結果得られた符号語を候補にして生成行列を用いた復号法であるGS復号法を用いることにより，復号誤り確率が低減されることを示す．

キーワード ブロックターボ符号，生成行列，最尤復号法，ターボ復号

A Decoding Algorithm Based on Generator Matrix for Block Turbo Codes

Hideaki OHSHIMA　Naonori OGASAHARA　Manabu KOBAYASHI　Shigeichi HIRASAWA

Dep. of Industrial and Management Systems Engineering
Waseda University
3-4-1 Ohkubo, Shinjuku-ku, Tokyo 169-8555
tel.03-5286-3290
fax.03-5273-7215
E-mail: hideaki@hirasa.mgmt.waseda.ac.jp

Abstract　In 1993 turbo codes have been proposed by C.Berrou. In recent year turbo codes attract attentions in the field of coding theory as new channel encoding schemes close to the Shannon limit. In this paper, at first we show that there exist block turbo codes of product-type whose minimum distance is larger than that of conventional product codes. For block turbo codes of product-type we propose a new decoding algorithm based on generator matrices. This algorithm excutes turbo decoding to obtain a candidate codeword. Then it uses GS decoding which achieves maximum likelihood decoding. We show that the proposed decoding algorithm can reduce a bit-error rate (BER) for block turbo codes.

key words　Block turbo codes, Generator matrix, Maximum likelihood decoding , Turbo decoding

1 はじめに

ターボ符号は，1993年にフランスのC.Berrouらによって提案された通信路符号化方式[6]であり，シャノン限界に迫る新しい方式として近年，符号理論などの分野で注目を集めている．ターボ符号の符号化は，2つの要素符号$C^-, C^|$に対し，符号C^-で情報を符号化した後，符号$C^|$の前に一度情報をメモリに記憶させ，その後情報を取り出す順序を変換するインタリーバによって情報を並べ替え，それを符号$C^|$によって符号化するという特徴がある．

ターボ復号法は符号$C^|$に対して，受信系列に基づいて復号化を行い，その際に各情報シンボルに対する信頼度情報を求める．次にインタリーバによる逆の並べ替えを行った後，符号C^-に対してこの信頼度情報と受信系列に基づいて復号化を行い，各シンボルの新しい信頼度情報を求める．この復号過程を1回目の復号とする．2回目以降同様の繰り返しを数回行い，その結果を出力する．またターボ復号の信頼度情報の計算方法として，最大事後確率復号法(MAP復号法[5])がある．中でもこれを実現する復号法として，L.R.Bahlらによって，トレリスの前方からと後方からの繰り返し演算を行い各ノードに対する事後確率を求める復号法[2](以降BCJRアルゴリズムと呼ぶ)などが提案されている．しかし前述の通りターボ復号はシャノン限界に近い復号法であるが，ある程度復号を繰り返すと，ビット誤り確率の値がある値に収束することが知られている．

一方，最尤復号法は各符号語の生起確率が等しいと仮定したときに復号誤り確率が最小になるという意味で最適な復号法である．しかし全符号語の中で尤度が最も大きい符号語を探索し出力する復号法であるため，計算量が膨大となり，これを低減する様々な研究が行われている．中でもD.Gazelleらによって置換生成行列を用いて効率よくテストパターンを生成し，これを符号化することを繰り返す復号法[3](以降GS復号法と呼ぶ)などが提案されている．

そこで，本研究ではchecks on checksを付加する符号化を行いインタリーバで情報の順序を置換するブロックターボ符号(積符号型ブロックターボ符号と呼ぶことにする)を対象とし新しい復号法を提案する．まずこの積符号型ブロックターボ符号に対し，従来の積符号より最小距離が大きくなる例を示す．一方この符号化に対しターボ復号を行うと，チェックビットが先に符号化した符号の符号語とならないため，最尤復号した時のビット誤り確率の値に比べ劣化した値に収束してしまう．そこで，ビット誤り確率がある程度一定の値になるまでターボ復号を繰り返した後に，最後の繰り返しにより計算された信頼度情報を硬判定し，それを初期符号語とみなしてGS復号を行う．結果的に，ビット誤り確率をさらに低減することが可能であることを示す．

まず2章で通信路のモデルと対数事後確率比について定義し，3章で本研究で用いる積符号型ブロックターボ符号と従来の復号法について説明する．4章で今回用いているターボ符号の生成行列の構造について述べる．5章で生成行列を用いたブロックターボ符号の復号法を提案し，6章においてシミュレーションによる評価と考察を行う．

2 準備

2.1 通信路のモデル

符号長n，情報記号数k，最小距離dの2元線形ブロック符号をCとする．また，送信側の符号器により生成された符号語を$\boldsymbol{c} = (c_1, c_2, \cdots, c_n), c_i \in \mathrm{GF}(2)$，とし，送信信号系列$\boldsymbol{x} = (x_1, x_2, \cdots, x_n)$，$x_i = (-1)^{c_i}$，に変換する．各々のシンボル$x_i$は加法的白色ガウス雑音(Additive White Gausian Noise : AWGN)通信路に送出される．受信側では，受信した信号系列$\boldsymbol{y} = (y_1, y_2, \cdots, y_n)$から信頼度系列$\boldsymbol{\phi} = (\phi_1, \phi_2, \cdots, \phi_n)$，$\phi_i = \ln \frac{p(y_i|0)}{p(y_i|1)}$，を生成する．復号器においてはこの$\boldsymbol{\phi}$及び硬判定受信系列$z = (z_1, z_2, \ldots, z_n)$，$z_i \in \mathrm{GF}(2)$，

$$z_i = \begin{cases} 0, & \phi_i \geq 0, \\ 1, & \phi_i < 0, \end{cases} \tag{1}$$

から送信された符号語を推定する．また，$|\phi_i|$が大きいほどz_iが誤っている確率は小さいので，$|\phi_i|$を信頼度と呼ぶことにする．

2.2 対数事後確率比

事前確率$P(x_t)$と事後確率$p(x_t|y_t)$の対数事後確率比を次のように定義する．

$$L(x_t) = \ln \frac{P(x_t = +1)}{P(x_t = -1)} \tag{2}$$

$$\begin{aligned} L(x_t|y_t) &= \ln \frac{P(x_t = +1|y_t)}{P(x_t = -1|y_t)} \\ &= \ln \left(\frac{p(y_t|x_t = +1)}{p(y_t|x_t = -1)} \cdot \frac{P(x_t = +1)}{P(x_t = -1)} \right) \\ &= \ln \frac{p(y_t|x_t = +1)}{p(y_t|x_t = -1)} + \ln \frac{P(x_t = +1)}{P(x_t = -1)} \\ &= L_c \cdot y_t + L(x_t) \qquad (3) \\ &\quad (t = 1, 2, \cdots, n) \end{aligned}$$

ただしAWGN通信路において$L_c = \phi_t / y_t$は定数である．

3 従来研究

3.1 積符号型ブロックターボ符号

$k_1 k_2$ビットの情報記号，$\boldsymbol{u} = (u_1, u_2, \cdots, u_{k_2})$と$k_1$ビットずつ区切った部分系列$\boldsymbol{u}_i$に分割する．

[積符号型ブロックターボ符号の符号化]

① 2元(n_1, k_1, d_1)組織符号C^-を用いて各$\boldsymbol{u}_i$に対して符号化を行いチェックビット$\boldsymbol{p}_i^-$を付加する．すなわち$(\boldsymbol{u}_i, \boldsymbol{p}_i^-) \in C^-$である．また$\boldsymbol{p}^- = (\boldsymbol{p}_1^-, \boldsymbol{p}_2^-, \cdots, \boldsymbol{p}_{k_2}^-)$とする．

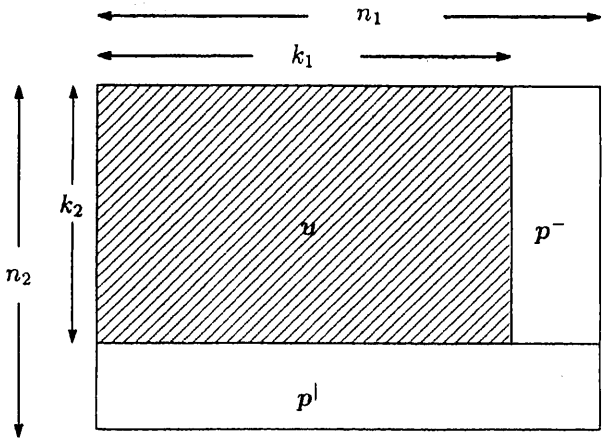

図 1: 積符号型ブロックターボ符号の構成

② 情報 u とパリティビットの p^- をインタリーバを用いて置換する．この系列を $v=(v_1,v_2,\cdots,v_{n_1})$ とする．ただし v_i は長さ k_2 ビットの部分系列である．

③ 2 元 (n_2,k_2,d_2) 組織符号 $C^|$ を用いて各 v_i に対し符号化を行い，チェックビット $p_i^|$ を付加する．すなわち $(v_i,p_i^|)\in C^|$ である．また $p^|=(p_1^|,p_2^|,\cdots,p_{n_1}^|)$ である． □

積符号型ブロックターボ符号の構成を図 1 に示す．

この符号が従来の積符号を含んでいることは容易に分かる．

3.2 BCJR アルゴリズム [2]

符号に対するトレリス線図の M 個の異なる状態を $s\in\{0,1,\cdots,M-1\}$ とする．また時点 t の状態を S_t，送信シンボルを x_t とした時，時点 t から t' の状態系列を $S_t^{t'}=S_t,S_{t+1},\cdots,S_{t'}$，送信記号系列を $x_t^{t'}=x_t,x_{t+1},\cdots,x_{t'}$，受信記号系列を $y_t^{t'}=y_t,y_{t+1},\cdots,y_{t'}$ とする．この時アルゴリズムを示すための準備として以下のように定義する．

$$\begin{aligned}L(\hat{x}_t) &= \ln\frac{P(x_t=+1|y_1^n)}{P(x_t=-1|y_1^n)}\\ &= \ln\frac{\sum_{\substack{(s',s)\\ x_t=+1}} p(S_{t-1}=s',S_t=s,y_1^n)}{\sum_{\substack{(s',s)\\ x_t=-1}} p(S_{t-1}=s',S_t=s,y_1^n)}\end{aligned}\quad(4)$$

$$\begin{aligned}p(s',s,y_1^n) &= p(s',y_1^{t-1})\cdot p(s,y_t|s')\cdot p(y_{t+1}^n|s)\\ &= \alpha_{t-1}(s')\cdot\gamma_t(s',s)\cdot\beta_t(s)\end{aligned}\quad(5)$$

$$\begin{aligned}\alpha_t(s) &= p(S_t=s,y_1^t)\\ &= \sum_{s'}\gamma_t(s',s)\cdot\alpha_{t-1}(s')\end{aligned}\quad(6)$$

$$\begin{aligned}\beta_t(s') &= p(y_{t+1}^n|S_t=s')\\ &= \sum_{s}\gamma_{t+1}(s',s)\cdot\beta_{t+1}(s)\end{aligned}\quad(7)$$

$$\begin{aligned}\gamma_t(s',s) &= p(S_t=s,y_t|S_{t-1}=s')\\ &= p(y_t|s',s)\cdot P(s|s')\\ &= p(y_t|x_t)\cdot P(x_t)\\ &\quad(t=1,2,\cdots,n)\end{aligned}\quad(8)$$

$p(s',s,y_1^n)$ はトレリスの各枝に対する確率を表す．また MAP 復号法は各ビットごとに (4) 式の対数事後確率比を求めることが目的となる．その事後確率をトレリスの構造を用いてその前方からと後方からの繰り返し演算によって効率的に求めるアルゴリズムが BCJR アルゴリズムである．

[BCJR アルゴリズム]

1) $\alpha_0(0)=1$，$\beta_n(0)=1$ とする．

2) y_t を受信した後，(8) 式により $\gamma_t(s',s)$ を計算し，(6) 式により $\alpha_t(s)$ を計算する．またすべての t，s に対して，$\alpha_t(s)$ を記憶しておく．

3) y_1^n を受信した後，(7) 式により $\beta_t(s)$ を計算する．結果的に (5) 式 →(4) 式と順に計算し，$L(\hat{x}_t)\geq 0$ ならば $\hat{x}_t=+1$，さもなくば $\hat{x}_t=-1$ と判定し，$\hat{x}_1^n$ を出力する． □

3.3 ターボ復号法 [1][4]

ターボ復号法は，図 1 のように表されたターボ符号に対して縦方向と横方向のそれぞれに対し復号を行い，その結果を用いて繰り返し演算を行う復号法である．まず最初に縦の方向に復号する．この復号の際に各ビットごとの事後確率を計算する．次に符号化においてインタリーブされた部分をデインタリーブ (デインタリーブはインタリーブの逆変換) し，その後横方向に復号する．その際，縦方向で計算された各ビットごとの事後確率から事前確率 (外部情報) を導出しこれを用いて復号を行う．その後も同様に復号の際に得られる事後確率から次の事前確率を生成する．横方向，縦方向の復号を 1 回の繰り返しとし，数回繰り返すことで復号を終える．以下にこれらの復号の詳細を示す．

(2) 式より事前確率 $P(x_t)$ は

$$\begin{aligned}P(x_t=\pm1) &= \frac{e^{\pm L(x_t)}}{1+e^{\pm L(x_t)}}\\ &= \left(\frac{e^{-L(x_t)/2}}{1+e^{-L(x_t)}}\right)\cdot e^{L(x_t)x_t/2}\\ &= A_t\cdot e^{L(x_t)x_t/2}\end{aligned}\quad(9)$$

となり，同様な式変形から

$$p(y_t|x_t) = B_t \cdot e^{L_c \cdot y_t \cdot x_t/2} \tag{10}$$

となる．(9)，(10) 式を (8) 式に代入すると

$$\gamma_t(s',s) = A_t \cdot B_t \cdot e^{x_t(L_c y_t + L(x_t))/2} \tag{11}$$

となる．また (11) 式を (5) 式に代入することにより次式が導かれる．

$$\begin{aligned} L(\hat{x}_t) &= L_c \cdot y_t + L(x_t) \\ &+ \ln \frac{\sum_{\substack{(s',s) \\ x_t=+1}} \alpha_{t-1}(s') \cdot \beta_t(s)}{\sum_{\substack{(s',s) \\ x_t=-1}} \alpha_{t-1}(s') \cdot \beta_t(s)} \end{aligned} \tag{12}$$

(12) 式の第 1 項は受信系列による尤度，第 2 項が事前確率を表し，第 3 項が次の復号の外部情報を表す．つまり (12) 式を繰り返し計算することで，各ビットごとの事後確率を求め，第 3 項の値を次の復号の事前確率として繰り返し復号を行う．各ビットごとの事後確率の値が収束する一定回数繰り返した時点で終了し，$L(\hat{x}_t) \geq 0$ ならば $\hat{x}_t = +1$，さもなくば $\hat{x}_t = -1$ と硬判定し，その系列を出力する．

3.4 GS 復号法 [3]

2 元線形 (n,k,d) ブロック符号に対し，ハミング重み $l(l=1,2,\cdots,k)$ のエラーパターン $e \in \{0,1\}^k$ の集合を Θ_l と定義する．また各 Θ_l の要素を 2 進数表現したとき数の小さい順に $e_i = (e_{i,1}, e_{i,2}, \ldots, e_{i,k})$，$i = 1,2,\ldots,\binom{k}{l}$ とする．また符号語 x に対する z の信頼度損失を $\ell(x) = \sum_{i|x_i \neq z_i} |\phi_i|$ とする．

[GS 復号法]

1) 信頼度 $|\phi_i|$ の高い順に生成行列 G を列置換し，始めの k 列が線形独立になるようにしたものを G' とする．同様な bit 位置の列置換を ϕ，z に対して行い，それぞれ $\tilde{\phi}$，$\tilde{z}$ を得る．また G' の最初の $k \times k$ 行列が単位行列になるように行操作し $\tilde{G}$ を得る．

2) 硬判定系列 $\tilde{z}$ の始めの k ビットを m_0 とし初期符号語 $c_0 = m_0\tilde{G}$ とする．$l = 1$ とする．

3) $e_1 \in \Theta_l$ に対し $\ell(c_{ML}) \leq \sum_{j=1}^{k} e_{1,j}|\phi_j|$ なら c_{ML} を出力して終了．そうでなければ $i = 2,3,\ldots,\binom{k}{l}$ に対し $\ell(c_{ML}) > \sum_{j=1}^{k} e_{i,j}|\phi_j|$ を満たす e_i を符号化し $c^{err} = e_i\tilde{G}$ を得る．$\ell(c_{ML}) > \ell(c_0 \oplus c^{err})$ なら $\ell(c_0 \oplus c^{err}) \to \ell(c_{ML})$，$c_0 \oplus c^{err} \to c_{ML}$ と更新する．ここで c_{ML} は復号途中のある段階で既に得られている最尤の符号語を表す．

4) $l = k$ ならば $c_{ML} = c_0 \oplus c^{err}$ を出力し終了．さもなくば $l \to l+1$ とし 3) へ． □

4 積符号型ブロックターボ符号の生成行列

[C^- と $C^|$ の符号化]

$u=(u_{11}, u_{12}, \cdots, u_{1k_1}, \cdots, u_{k_2 1}, u_{k_2 2}, \cdots, u_{k_2 k_1})$ のように情報部分を表し，C^- の生成行列を G_1，$C^|$ の生成行列を G_2 とすると C^- と $C^|$ の全体の符号化に用いる生成行列 $G^{(1)}$，$G^{(2)}$ はそれぞれ次のように表せる．

$$G^{(1)} = \overbrace{\begin{bmatrix} G_1 & & & \mathbf{0} \\ & G_1 & & \\ & & \ddots & \\ \mathbf{0} & & & G_1 \end{bmatrix}}^{k_2\text{個}}.$$

$$G^{(2)} = \overbrace{\begin{bmatrix} G_2 & & & \mathbf{0} \\ & G_2 & & \\ & & \ddots & \\ \mathbf{0} & & & G_2 \end{bmatrix}}^{n_1\text{個}}.$$

$n_1 k_2 \times n_1 k_2$ 行列において，各行各列に 1 つだけ 1 を含む行列を置換行列 P と呼ぶことにすると，積符号型ターボ符号 C の生成行列 G は

$$G = G^{(1)} P G^{(2)} \tag{13}$$

と書くことができる．このことから積符号型ブロックターボ符号 C は

$$c = uG, \quad (c \in C) \tag{14}$$

により符号化することができる．

この時，置換行列 $P = [p_{a,b}]$ に次式のような条件を付け加える．

$$\sum_{a=(i-1)n_1+1}^{i \cdot n_1} \sum_{b=(j-1)k_2+1}^{j \cdot k_2} p_{a,b} = 1 \tag{15}$$

$$(1 \leq i \leq k_2, \quad 1 \leq j \leq n_1)$$

(15) 式を付け加えることによって積符号型ブロックターボ符号 C の最小距離を D とした時，$D \geq d_1 d_2$ となる [7]．

従来の積符号はその最小距離が厳密に $d_1 d_2$ であるのに対し，上で定義した積符号型ブロックターボ符号の中にはその最小距離が $d_1 d_2$ より大きくなるものが存在する．以下にそのような符号の例を示す．

例 4.1 $C^-, C^|$ をそれぞれ (15,7,5)BCH 符号とし，生成行列 G_1, G_2 を次式で定義する．

$G_1 = G_2 =$

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

表 1: 最小距離が $d_1 d_2$ より大きくなる置換行列の例 ($p_{a,b} = 1$ の位置を示す)

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
b	2	10	19	28	35	39	48	53	58	66	76	83	91	95	104	1	13	18	26	30	37
a	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
b	43	51	59	68	74	84	88	97	103	4	9	21	25	34	36	47	55	60	70	72	82
a	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
b	90	92	105	6	8	17	24	33	40	49	52	62	67	71	79	89	94	101	7	11	15
a	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84
b	23	32	38	44	56	61	65	73	80	86	96	100	3	14	20	22	29	42	46	54	63
a	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105
b	69	75	78	85	93	102	5	12	16	27	31	41	45	50	57	64	77	81	87	98	99

上の生成行列は各行のハミング重みがなるべく大きなものを選んである．このとき，以下に示す 18 個の符号語が最小ハミング重みの符号語である．これを C_{min} と表す．

$$C_{min} = \left\{ \begin{array}{l} (000000101010110), (000100010100011), \\ (000001100110001), (000101011000100), \\ (000110001001010), (001000111100000), \\ (001100000010101), (010000110001010), \\ (100001010001001), (100010000000111), \\ (101000001011000), (110000000110010), \\ (001010100001001), (010101000011000), \\ (100011001100000), (111000010000100). \\ (010011100000100), (011110000100000) \end{array} \right\}$$

ここで置換行列 $P = [p_{a,b}]$ を表 1 に示すように定義する．この置換行列は $p_{a_1,b_1} = p_{a_2,b_2} = 1$ なる 位置 $(a_1, b_1), (a_2, b_2)$ に対し

$$a_1 \equiv a_2 \pmod{n_1} \Leftrightarrow \left\lfloor \frac{b_1}{k_2} \right\rfloor = \left\lfloor \frac{b_2}{k_2} \right\rfloor \qquad (16)$$

が成り立つ．すなわち図 1 において同一の列にある要素をまた同一の列に置換している．従って，$\forall \boldsymbol{c} \in C_{min}$ に対し $(\boldsymbol{u}_i, \boldsymbol{p}_i^-) \in \{\boldsymbol{0}, \boldsymbol{c}\}, i = 1, 2, \ldots, k_2$, のときのみ最小ハミング重みが $d_1 d_2$ となる可能性がある．その中のあるパターンに対し $(\boldsymbol{v}_i, \boldsymbol{p}_i^|) \in C_{min}, i = 1, 2, \ldots, n_1$, が成り立つ時のみ最小距離は $d_1 d_2$ であるが，表 1 で定義した置換行列は計算機探索によりこれを満たさないように選ばれている．従ってその最小距離は $d_1 d_2$ より大きい．
□

ここでは最小距離が $d_1 d_2$ より大きくなる例を具体的に示したが，置換に (15) 式の制限を設けた積符号型ブロックターボ符号はその重み分布を考えると，従来の積符号より大きなハミング重みを持つ符号語の数が大きくなることが期待できる．そこで次節において，この (15) 式の制限を設けた置換行列 P を用いた符号の復号法について考える．

5 積符号型ブロックターボ符号の復号法

5.1 生成行列を用いた提案復号法

通常の積符号の場合，図 1 において任意の行と列がそれぞれ C^- と $C^|$ の符号語となっているため，ターボ復号を行った時，すべての行と列に対し信頼度を求めることが可能である．各行と列に MAP 復号を用いた場合，ほぼ最尤復号と同程度のビット誤り確率になる．しかし 3.1 節及び 4 節で定義した積符号型ブロックターボ符号に関しては，図 1 の $\boldsymbol{p}^|$ 部分を横方向に見た時 C^- 符号語となっていない．そのためこの部分に対し C^- の復号を用いることができない．これを考慮した復号法を以下に示す．

[生成行列を用いた復号法]

1) 図 1 の縦方向に，各ビットの事前確率と受信系列の尤度から各ビットに対し (12) 式を計算し，(12) 式の外部情報にあたる第 3 項の値から次の横の復号の各ビットの事前確率を求める．

2) 図 1 の横方向に，1) で得られた事前確率と受信系列の尤度から各ビットごとに (12) 式を計算し，(12) 式の外部情報にあたる第 3 項の値から次の縦の復号の各ビットの事前確率を求める．

3) 1),2) を一回の繰り返しとし，一定回数繰り返す．

4) 情報部分である $\boldsymbol{u}$ に対応する x_t に対し，$L(\hat{x}_t) \geq 0$ ならば $\hat{c}_t = 0$，さもなくば $\hat{c}_t = 1$ と硬判定し，その系列を生成行列 G を用いて符号化した系列を候補符号とする．

5) 候補符号を初期符号語とし，GS 復号法を用いて復号する．
□

この復号法の性能は次章においてシミュレーションにより評価する．

6 シミュレーション結果による評価

符号は C^-，$C^|$ 共に (8,4,4) 符号を用いた．置換行列は (15) 式を満たす P を用いた．

以下にシミュレーション結果を示す．

ターボ復号を 10 回繰り返した後，その結果から得られる符号語を候補として GS 復号法を行う．このシミュレーション結果を図 2 に示す．

6.1 ビット誤り確率についての考察

図 2 からターボ復号は 1 回目から 4 回目程度の復号において，ビット誤り確率が除々に小さくなっていることから，ターボ復号がある程度効率良く復号ができていることは明らかである．ここで 5 回目から 10 回目までを見てみるとわずかにビット誤り確率は小さくなっているが，ほぼ一定の値と考えてよい．つまりこれ以上繰り返

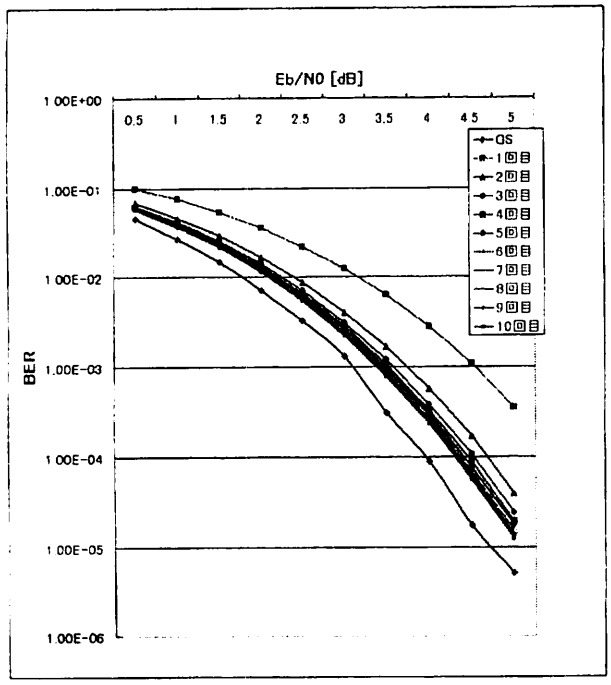

図 2: ターボ復号+GS 復号のビット誤り確率

し回数を増やしてもこの値はほぼ収束してしまっているため，これ以上の改善を望むことはできない．そこで今回の提案では計算量は増えるがビット誤り確率をさらに改善することを考えている．本稿での提案復号法は図 2 からもわかるとおり，広い範囲の SN 比の値に対して改善されている．従来の積符号をターボ復号を用いて復号した場合，繰り返し復号していくと最尤復号したときの値に収束していく．このことは C^-，$C^{\mid}$ のどちらも符号語となっているためであると考えられる．一方積符号型ブロックターボ符号は $C^{\mid}$ のパリティビット部分 $(p^{\mid})$ に対して横方向に見てみると符号語となっていない．そのためこの部分での復号が行えないので，ターボ復号によって繰り返し復号を行っても最尤復号していったときの値に収束していかないと考えられる．このことから最後の繰り返しの結果を候補として GS 復号法を用いた時ビット誤り確率がさらに小さくできる．

6.2 計算量に関する考察

今回の提案ではターボ復号をした後に GS 復号を行うことによってターボ復号のみでは実現できないビット誤り確率を実現できるようにした．これに伴い計算量は増加してしまうため計算量とビット誤り確率が trade off の関係にある．ただし今回の提案ではターボ復号を用いて候補符号語を導出し，それを用いて GS 復号を行うことにより，GS 復号法においてテストエラーパターンを少なくすることができる．このことはターボ復号を用いて導出した候補符号語がある程度良い符号語であるので，その符号語の尤度が大きな値になっているためであると考えられる．したがって単独で GS 復号法を行うよりも計算量をある程度に抑え，そのもとでビット誤り確率を低減することができる．4 章に示した最小距離が増加する場合にはこの効果がさらに期待できる．

7 まとめと今後の課題

本論文では，積符号型ブロックターボ符号に対して，ターボ復号により大域的探索を行い GS 復号法により局所的探索を行う 2 つの方法を効率良く組み合わせる復号法を提案した．すなわちターボ復号を行った際，復号誤り確率の収束値がそれほど良くないために最後の繰り返しの結果から候補符号語求め，最尤復号法である GS 復号法を用いる方法でこれによって，復号誤り確率が低減されることを示した．最尤復号は積符号型ターボ符号の生成行列を求めることにより可能となる．今回は従来の積符号と最小距離が変わらない符号でしかシミュレーションを行うことができなかったが，今後の課題として C^-，$C^{\mid}$ それぞれの符号長をさらに大きくし，従来の積符号との比較が必要である．また最小距離が積符号よりも大きくなる場合の理論的保証や解析も行う必要がある．

謝辞 著者の大島は日頃より有益な助言をいただいた早稲田大学 竹内公二氏，八木秀樹氏及び，平澤研究室の方々に感謝いたします．

参考文献

[1] J.Hagenauer,E.Offer,L.Papke "Iterative Decoding of Binary Block and Convolutional Codes," *IEEE Trans. Inform. Theory*, vol.IT-42, No.2,pp.429-445, Mar. 1996.

[2] L.R.Bahl,J.Coche,F.Jelinek,J.Raviv "Optimal Decoding of Linear Codes for Minimizing SymbolError Rate," *IEEE Trans. Inform. Theory*, vol.IT-5, pp.284-287, Mar. 1974.

[3] D.Gazelle and J.Snyders, "Reliability-based code-search algorithm for maximum likelihood decoding of block codes," *IEEE Trans. Inform. Theory*, vol.43, pp.239-249, Jan. 1997.

[4] R.M.Pyndiah, "Near-optimum decoding of product code:Block turbo codes." *IEEE Trans. Commun.*, vol.46, pp.1003-1010, Aug. 1998.

[5] Y.Kaji, R.Shibuya, T.Fujiwara, T.Kasami, and S.Lin, "MAP and LogMAP Decoding Algorithms for Linear Block Codes Using a Code Structure," *IEICE Trans. Fundamentals.*, vol.E83-A. Oct. 2000.

[6] C.Berrow, A.Glavieux, and P.Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo code(1)," *IEEE Int. Conf. Commun ICC'93*, vol2/3, pp.1064-1071, May, 1993.

[7] 小林 学，松嶋 敏泰，平澤 茂一，"ブロックターボ符号の生成行列と性能評価," 信学技報, July, 2001.