ターボ符号について。

Kwame Ackah Bohulu

01-12-2017

1 Introduction

ターボ符号は二つの畳み込み符号をインタリーバにより、並列連結をして、作られている。インタリーバにより並列連結することで、低い重みを持っている符号が得られる。ターボ符号は Claude Berrou さんが発明された符号で、1993 年に紹介されました。通常の場合は、同じ畳み込み符号器を使用し、2 番目の符号器の ‘前にインタリーバがあります。そのため、最低レートが 1/3 が、パンクチュアリングすると、1/2 か 2/3 に上がることがでる。この資料の第二章で畳み込み符号をちょっと説明する。第三章でターボ符号の作り方を紹介する。第四章で BCJR アルゴリズムを説明する。最後に、第五章でターボ復号アルゴリズムとシムレーションのシステムモデルを説明する。

2 畳み込み符号

畳み込み符号は線形有限状態シフトレジスタで情報系列を入力して、作られている。一般的に、シフトレジスタが K の k ビット段階と n 個の線形代数関数発生器で構成される。K はシフトレジスタの拘束長を示し、k は、一度に何個のビットをシフトするかを定め、n は何個の出力ビットが出るかを定める。代数関数発生器は Kk の大きさのベクトルで構成され、符号器と mod 2 の加算器の接続を指定する。畳み込み符号器のレートは $\frac{k}{n}$ です。

例 **1** , 図 1
K=3,k=1,n=3。
$g_1 = [100], g_2 = [101], g_3 = [111]$
8 進形式 (octal form) =(4, 5, 7)
$c^{(1)} = u * g_1, c^{(2)} = u * g_2, c^{(3)} = u * g_3$
符号順序 $=(c_1^{(1)}, c_1^{(2)}, c_1^{(3)}, c_2^{(1)}, c_2^{(2)}, c_2^{(3)}, ...)$

D domain
$U(D) = \sum_{i=0}^{\infty} u_i D^i$
$g_1(D) = 1, g_2(D) = 1 + D^2, g_3(D) = 1 + D + D^2$
$C^{(1)}(D) = U(D)g_1(D), C^{(2)}(D) = U(D)g_2(D), C^{(3)}(D) = U(D)g_3(D)$
$C(D) = C^{(1)}(D^3) + DC^{(2)}(D^3) + D^2C^{(3)}(D^3)$

畳み込み符号の構造は trellis グラフで簡単に描ける。trellis のノード数は、畳み込み符号器の可能な状態数とひとしいである。一般的に、レート $\frac{k}{n}$ で、ながさ K の拘束長を持つ畳み込み符号の trellis は $2^{k(K-1)}$ 個のノードとノードに入る 2^k 個の支部とノードに出る 2^k 個の支部。

2.0.1 再帰的系統的と非再帰的系統的畳み込み符号器

情報系列が符号系列に直接に入っている畳み込み符号は系統的畳み込み符号系列という。再帰的畳み込み符号は帰還シフトレジスタで作られている。再帰的符号は常に系統的が、非再帰的符号は非系統的であることが多い。再帰的系統的畳み込み符号の生成行列の D domain 現れは多項式比が入っている。

例 **2** , 図 2
レート =1/2, K=3, $g_1 = [111]$, $g_2 = [101]$
G=[1 g_2/g_1] 1 は系統的出力を示し、分子はフィードフォワード出力を示し、分母は帰還入力を示す。あるレートと拘束長が与え、再帰的系統的畳み込み符号は、非再帰的系統的畳み込み符号の自由距離が得られる。ターボ符号を作るのに、再帰的系統的畳み込み符号は大変重要です。

3 ターボ符号器

一般のターボ符号器は、図 3 にしめされて、再帰的系統的符号器である。並列連結された二つの再帰的系統的畳み込み符号器（要素符号器と言う。）で構成され、2 番目の要素符号器の前にインタリーバがある。通常の場合は両方の要素符号器は同じである。それぞれの要素符号器はパリティビット系列を生成する。インタリーバと要素符号器の組み合わせは、数が少ない低い重みをもつ符号語のふごうが生成できる。図 3 で、出力のレートは 1/3 ということがわかる。レートを上がるために、パリティビット系列をパンクチュアリングする。ターボ符号器の可な能状態が巨大ため、最尤復号が不可能である。そのかわりに、ターボ復号アルゴリズムを使用する。そのアルゴリズムは BCJR アルゴリズムに基づいている。

4 BCJR アルゴリズム

BCJR アルゴリズムは Bahl さん, Cocke さん,Jelinek さんと Raviv さんが 1974 に発明されたアルゴリズムである。このアルゴリズムは最も可能性の高い入力系列を探すより、MAP アルゴリズムを使用し、それぞれの入力記号を復号する。BCJR アルゴリズムを説明するために、以下の条件を想定する。
1. 情報系列 **u** の長さは、N であり,**u**=$\{u_1, u_2, ..., u_N\}$
k=1 の場合 $u_i \in \{0, 1\}$
2. 符号系列 **c** の長さは、N であり,**c**=$\{c_1, c_2, ..., c_N\}$
c_i の長さは、n である。
3. 符号系列は AWGN チャネルで送信し、復号器で受信された系列 **y** は実数で、長さは nN であり、**y**=$\{y_1, y_2, ..., y_N\}$

BCJR アルゴリズムは事後対数尤度比 (a posteriori LLR),$L(u_i|y)$ を計算し、元の情報系列を推定する。

$$L(u_i|y) = \ln \frac{P(u_i = 1|\boldsymbol{y})}{P(u_i = 0|\boldsymbol{y})} \tag{1}$$

BCJR アルゴリズムの計算をもっと便利にするために、trellis グラフを使う。時間が i のとき、現在状態 $\sigma_i = s$ であり, 過去の状態は $\sigma_{i-1} = s'$ である。復号器で受信された系列は y_i である。時間が i になる前に、i-1 個の系列が送信され、その後は N-i の系列が受信する。そのうえ、時間が i のとき、完全系列 $\mathbf{y}$ は、過去、現在と未来の三つの系列に分けられる。

$$\boldsymbol{y} = \boldsymbol{y}_{<i}\boldsymbol{y}_i\boldsymbol{y}_{>i}$$

trellis グラフでわかることは、$\sigma_{i-1} = s'$ から $\sigma_i = s$ の遷移は、u_i の値で決められる。遷移が互いに排他的ため、ある遷移が起こる確率はそれぞれの確率の和である。

$$\therefore P(u_i = 1|\boldsymbol{y}) = \sum_{R_1} P(s', s|\boldsymbol{y})$$

$$P(u_i = 0|\boldsymbol{y}) = \sum_{R_0} P(s', s|\boldsymbol{y})$$

$R_0 = u_i = 0$ での $\sigma_{i-1} = s'$ から $\sigma_i = s$ の遷移の組
$R_1 = u_i = 1$ での $\sigma_{i-1} = s'$ から $\sigma_i = s$ の遷移の組

$\mathbf{y}$,$P(u_i = 1|\boldsymbol{y})$ と $P(u_i = 0|\boldsymbol{y})$ を式 1 に入力すると、

$$\begin{aligned} L(u_i|y) &= \ln \frac{\sum_{R_1} P(s', s|\boldsymbol{y})}{\sum_{R_0} P(s', s|\boldsymbol{y})} \\ &= \ln \frac{\sum_{R_1} P(s', s, \boldsymbol{y})}{\sum_{R_0} P(s', s, \boldsymbol{y})} \\ &= \ln \frac{\sum_{R_1} P(s', s, \boldsymbol{y}_{<i}\boldsymbol{y}_i\boldsymbol{y}_{>i})}{\sum_{R_0} P(s', s, \boldsymbol{y}_{<i}\boldsymbol{y}_i\boldsymbol{y}_{>i})} \end{aligned} \tag{2}$$

Bayes 方程式で

$$\begin{aligned} P(s', s, \boldsymbol{y}_{<i}\boldsymbol{y}_i\boldsymbol{y}_{>i}) &= P(\boldsymbol{y}_{>i}|s', s, \boldsymbol{y}_{<i}\boldsymbol{y}_i)P(s', s, \boldsymbol{y}_{<i}\boldsymbol{y}_i) \\ &= P(\boldsymbol{y}_{>i}|s)P(\boldsymbol{y}_i, s|s', \boldsymbol{y}_{<i})P(s', \boldsymbol{y}_{<i}) \\ &= P(\boldsymbol{y}_{>i}|s)P(\boldsymbol{y}_i, s|s')P(s', \boldsymbol{y}_{<i}) \\ &= \alpha_{i-1}(s')\gamma_i(s', s)\beta_i(s) \end{aligned} \tag{3}$$

$\alpha_{i-1}(s') = P(s', \boldsymbol{y}_{<i})$ を定義し、時間が i-1 のとき、状態が s' で今まで受信された系列は $(\boldsymbol{y}_{<i}$ の共同確率を表す。
$\gamma_i(s', s) = P(\boldsymbol{y}_i, s|s')$ を定義し、過去の状態が s' の条件で未来の状態が s で

あり、受信された系列が $\boldsymbol{y}_i$ である確率を表す。
$\beta_i(s) = P(\boldsymbol{y}_{>i}|s)$ を定義し、現在の状態が s の条件で未来の系列は $\boldsymbol{y}_{>i}$ になるの条件付確率を表す。式 1 に式 3 を代入すると、以下になる。

$$L(u_i|y) = \ln \frac{\sum_{R_1} \alpha_{i-1}(s')\gamma_i(s',s)\beta_i(s)}{\sum_{R_0} \alpha_{i-1}(s')\gamma_i(s',s)\beta_i(s)} \tag{4}$$

4.1 $\gamma_i(s',s)$ の計算方法

$$\begin{aligned} \gamma_i(s',s) &= P(\boldsymbol{y}_i, s|s') \\ &= P(\boldsymbol{y}_i|s',s)P(s|s') \\ &= P(\boldsymbol{y}_i|c_i)P(u_i) \end{aligned} \tag{5}$$

AWGN チャネルの場合、

$$\gamma_i(s',s) = \frac{P(u_i)}{(\pi N_o)^{n/2}} exp(-\frac{\|y_i - c_i\|^2}{N_o}) \tag{6}$$

4.2 $\alpha_{i-1}(s')$ の計算方法

$\alpha_{i-1}(s') = P(s', \boldsymbol{y}_{<i})$。書き換えると、

$$\begin{aligned} \alpha_i(s) &= P(s, \boldsymbol{y}_{<i+1}) \\ &= P(s, \boldsymbol{y}_{<i}, \boldsymbol{y}_i) \end{aligned} \tag{7}$$

確率論で

$$P(A) = \sum_B P(A,B) \tag{8}$$

$$\begin{aligned} \therefore \alpha_i(s) &= \sum_{s'} P(s, \boldsymbol{y}_i|s', \boldsymbol{y}_{<i})P(s', \boldsymbol{y}_{<i}) \\ &= \sum_{s'} P(s, \boldsymbol{y}_i|s')P(s', \boldsymbol{y}_{<i}) \\ &= \gamma_i(s',s)\alpha_{i-1}(s') \end{aligned} \tag{9}$$

trellis がすべてゼロ状態から始まる場合、$\alpha_i(s)$ の初期条件は

$$\alpha_i(s) = \begin{cases} 1, & s = 0 \\ 0, & s \neq 0 \end{cases}$$

4.3 $\beta_i(s)$ の計算方法

$\beta_i(s) = P(\boldsymbol{y}_{>i}|s)$ 書き換えると

$$\begin{aligned}\beta_{i-1}(s') &= P(\boldsymbol{y}_{>i-1}|s') \\ &= \sum_s P(\boldsymbol{y}_{>i}|s', s, \boldsymbol{y}_i) P(s, \boldsymbol{y}_i|s') \\ &= \sum_s P(\boldsymbol{y}_{>i}|s) P(s, \boldsymbol{y}_i|s') \\ &= \beta_i(s)\gamma_i(s', s)\end{aligned} \tag{10}$$

trellis がすべてゼロ状態に終了する場合、$\beta_i(s)$ の初期条件は

$$\beta_N(s) = \begin{cases} 1, & s = 0 \\ 0, & s \neq 0 \end{cases}$$

4.4 Log-MAP と Max-Log-MAP

前に紹介した BCJR アルゴリズムは、trellis が長い場合、数値的に不安定である。そのかわりに、対数領域の BCJR アルゴリズム、Log-MAP と Max-Log-MAP アルゴリズムを使用する。Log-MAP の場合、以下の定義が必要である。

$$\begin{aligned}\widetilde{\alpha}_i(s) &= \ln(\alpha(s)) \\ \widetilde{\beta}_i(s) &= \ln(\beta(s)) \\ \widetilde{\gamma}_i(s', s) &= \ln(\gamma(s', s))\end{aligned} \tag{11}$$

前方再帰, 後方再帰と LLR は、以下の式で計算する。

$$\widetilde{\alpha}_i(s) = \ln \sum_{s'} \exp(\widetilde{\alpha}_{i-1}(s') + \widetilde{\gamma}(s', s))$$

$$\widetilde{\beta}_{i-1}(s) = \ln \sum_{s} \exp(\widetilde{\beta}_{i}(s') + \widetilde{\gamma}(s', s))$$

$$L(u_i) = \ln \Big[\sum_{R_1} \exp\ \widetilde{\alpha}_{i-1}(s') + \widetilde{\gamma}(s', s) + (\widetilde{\beta}_i(s')) \Big] - \ln \Big[\sum_{R_0} \exp(\widetilde{\alpha}_{i-1}(s') + \widetilde{\gamma}(s', s) + (\widetilde{\beta}_i(s')) \Big] \tag{12}$$

計算効率をよくするために、Max-Log-MAP アルゴリズムは以下の記法が使われている。

$$\begin{aligned}\max*\{x, y\} &\triangleq \ln(e^x + e^y) \\ \max*\{x, y, z\} &\triangleq \ln(e^x + e^y + e^z)\end{aligned} \tag{13}$$

上記の記法を使用し、前方再帰, 後方再帰と LLR は、以下の式で計算する。

$$\begin{aligned}
&\widetilde{\alpha}_i(s) = \max_{s'} *\{\widetilde{\alpha}_{i-1}(s') + \widetilde{\gamma}(s', s)\} \\
&\widetilde{\beta}_{i-1}(s) = \max_{s} *\{\widetilde{\beta}_i(s') + \widetilde{\gamma}(s', s)\} \\
&L(u_i) = \max_{R_1} *\Big\{\widetilde{\alpha}_{i-1}(s') + \widetilde{\gamma}(s', s) + (\widetilde{\beta}_i(s')\Big\} - \max_{R_0} *\Big\{\widetilde{\alpha}_{i-1}(s') + \widetilde{\gamma}(s', s) + (\widetilde{\beta}_i(s')\Big\}
\end{aligned} \tag{14}$$

両方の場合は、以下の初期条件は以下のようになる。

$$\widetilde{\alpha}_0(s) = \begin{cases} 0, & s = 0 \\ -\infty, & s \neq 0 \end{cases}$$

$$\widetilde{\beta}_N(s) = \begin{cases} 0, & s = 0 \\ -\infty, & s \neq 0 \end{cases}$$

5 ターボ復号アルゴリズム

ターボ符号器の可能状態が巨大なため、最尤復号が不可能である。そのかわりに、Claude Berrou さんが提案された準最適なターボ復号アルゴリズムを使用する。そのアルゴリズムは、Log-MAP か Max-Log-MAP アルゴリズムの反復使用に基づいている。Max-Log-MAP アルゴリズムに集中させる。$n = 2$, AWGN チャネルと BPSK 変調を仮定すると、

$$\boldsymbol{c}_i = (c_i^s, c_i^p), \boldsymbol{y}_i = (y_i^s, y_i^p)$$

式 6 は以下のようになる。

$$\gamma_i(s', s) = \frac{1}{(\pi N_o)} \exp(-\frac{(y_i^s)^2 + (y_i^p)^2 + 2(c_i^s)^2}{N_o}) P(u_i) \exp(\frac{2y_i^s c_i^s + 2y_i^p c_i^p}{N_o})$$

$\frac{1}{(\pi N_o)} \exp(-\frac{(y_i^s)^2 + (y_i^p)^2 + 2(c_i^s)^2}{N_o})$ は u_i と関係ないからむししてもかまいません。すると、以下の式が出る。

$$\begin{aligned}
\gamma_i(s', s) &= P(u_i) \exp(\frac{2y_i^s c_i^s + 2y_i^p c_i^p}{N_o}) \\
&P(u_i) \exp(\frac{2y_i^s c_i^s}{N_o}) \exp(\frac{2y_i^p c_i^p}{N_o}) \\
\therefore \widetilde{\gamma}(s', s) &= \ln P(u_i) + \exp(\frac{2y_i^s c_i^s}{N_o}) + \exp(\frac{2y_i^p c_i^p}{N_o})
\end{aligned} \tag{15}$$

上記の $\widetilde{\gamma}(s',s)$ を式 14 の $L(u_i)$ に入力すると、以下の式が出る。

$$L(u_i) = \max_{R_1} * \Big\{ \widetilde{\alpha}_{i-1}(s') + [\ln P(u_i) + \exp(\frac{2y_i^s c_i^s}{N_o}) + \exp(\frac{2y_i^p c_i^p}{N_o})] + (\widetilde{\beta}_i(s') \Big\} - \max_{R_0} * \Big\{ \widetilde{\alpha}_{i-1}(s') + [\ln P(u_i) + \exp(\frac{2y_i^s c_i^s}{N_o}) + \exp(\frac{2y_i^p c_i^p}{N_o})] + (\widetilde{\beta}_i(s') \Big\} \tag{16}$$

$$c_i^s = \begin{cases} \sqrt{\varepsilon_c}, & u_i = 1 \\ -\sqrt{\varepsilon_c}, & u_i = 0 \end{cases}$$

を仮定すると、

$$\begin{aligned} L(u_i) =& \frac{4\sqrt{\varepsilon_c} y_i^s}{N_o} + \ln \frac{P(u_i)=1}{P(u_i)=0} + \max_{R_1} * \Big\{ \widetilde{\alpha}_{i-1}(s') + \exp(\frac{2y_i^p c_i^p}{N_o}) + (\widetilde{\beta}_i(s') \Big\} - \\ & \max_{R_0} * \Big\{ \widetilde{\alpha}_{i-1}(s') + \exp(\frac{2y_i^p c_i^p}{N_o}) + (\widetilde{\beta}_i(s') \Big\} \\ & = L_c y_i^s + L^{(a)}(u_i) + L^{(e)}(u_i) \end{aligned} \tag{17}$$

$L_c y_i^s = \frac{4\sqrt{\varepsilon_c} y_i^s}{N_o}$ はチャネルの $L(u_i)$ 値を示し、組織的なビットの影響に関するチャネルの出力を表れる。
$L^{(a)}(u_i) = \ln \frac{P(u_i)=1}{P(u_i)=0}$ は先天的な (a priori)$L(u_i)$ 値である。

$$L^{(e)}(u_i) = \max_{R_1} * \Big\{ \widetilde{\alpha}_{i-1}(s') + \exp(\frac{2y_i^p c_i^p}{N_o}) + (\widetilde{\beta}_i(s') \Big\} - \max_{R_0} * \Big\{ \widetilde{\alpha}_{i-1}(s') + \exp(\frac{2y_i^p c_i^p}{N_o}) + (\widetilde{\beta}_i(s') \Big\}$$

は、外因性 $L(u_i)$ の値を示し、パリティビットに関する $L(u_i)$ の値である。

ターボ復号器は図 4 に描いてある。反復復号処理を説明するために、以下の仮定が必要。

1 ターボ符号器の要素符号器は RSC であり、レートは 1/2 である。

2 情報系列 $\boldsymbol{u}_i = (u_1, u_2, ..., u_N)$ は、一番目の要素符号器に入力し、パリティービット $\boldsymbol{c}^p = (c_1^p, c_2^p, ..., c_N^p)$ が出る。

3 情報系列はインタリーバに入力し、$\boldsymbol{u'}_i = (u'_1, u'_2, ..., u'_N)$ が出てきて、2 番目の要素符号器に入力し、$\boldsymbol{c'}^p = (c'^p_1, c'^p_2, ..., c'^p_N)$ が出る。

4 $\boldsymbol{u}_i, \boldsymbol{c}^p, \boldsymbol{c'}^p$ は BPSK 変調し、AWGN チャネルで送信する。

5 受信された系列は、$\boldsymbol{y}_i^s, \boldsymbol{y}^p, \boldsymbol{y'}_i$ であり、ターボ復号器の入力になる。

ターボ復号アルゴリズムは以下の順番で説明される。

1 1 番目の復号器の入力は $(\boldsymbol{y}^s, \boldsymbol{y}^p)$ であり、1 番目の要素符号器の出力と関係がある。

2 式 17 を使用し、$L(u_i)$ を計算する。最初の反復のとき、u_i は等確率を持つと仮定し、$L^{(a)}(u_i)$ の値は 0 になる。

3 1 番目の復号器の出力で $L(u_i)$ から $L_c y_i^s$ を引き、$L_{12}^{(e)}(u_i)$ を計算する。$L_{12}^{(e)}(u_i)$ はインタリーバで並び替えて、2 番目の復号器の $L^{(a)}(u_i)$ として、入力する。

4 2 番目の復号器の入力は $(\boldsymbol{y'}^s, \boldsymbol{y'}^p)$ であり、$L_{21}^{(e)}(u_i)$ を計算するのに使える。

5 $L_{21}^{(e)}(u_i)$ はインタリーバで並び替えて、次の反復の $L^{(a)}(u_i)$ として、1 番目の復号器に帰還する。

反復復号処理は、ある条件が当たるまで、あるいは、何回の反復まで続ける。最後の反復になったら、$L(u_i)$ を使用し、u_i の値を選択をする。

5.1 システムモデル

シムレーションで使用するシステムモデルは、図 5 に描かれている。それぞれのシステムブロックに以下の条件を使用する。

1. ターボ符号器

a ターボ符号器の入力は二進 (バイナリ) である。

b ターボ符号器の要素符号器は、拘束長 $K = 3$, レートは 1/2 の RSC 符号器である。

c 2 次インタリーバ、S-ランダムインタリーバと置換多項式に基づいてインタリーバを使用する。インタリーバの長さは,2^m であり、m の値は $3 - 10$ である。

d ターボ符号器のレートは 1/3 である。(パンクチュアリングはしない)

2. BPSK 変調器

入力ビットが 1 の場合、出力が +1 になり、入力ビットが 0 の場合、出力が −1 になります。

3. AWGN チャネル

ゼロ平均白色ガウス雑音を使用し、出力の SNR は 1 – 2 であり、ステップサイズ 0.1 である。

4. BPSK 復調器

入力が 0 より大きいの場合、出力ビットが 1 になり、入力が 0 より小さいの場合、出力ビットが 0 になります。

5. ターボ復号器

2 次インタリーバ、S-ランダムインタリーバと置換多項式に基づいてインタリーバを使用する。

6 参照

1. John G. Proakis, Masoud Salehi. Digital Communications Fifth Edition, McGraw-Hill
. 2. Silvio A. Abrantes,"From BCJR to turbo decoding: "MAP algorithms made easier", April 2004