整数最適化復号法の性能評価

柿 沼 美 希・高 野 祐 一（専修大学ネットワーク情報学部）

Performance Evaluation of Integer Optimization Decoding

Miki KAKINUMA, Yuichi TAKANO（School of Network and Information, Senshu University）

This paper explores a mathematical optimization approach to error correction. Tanatmis et al. [2010] proposed an integer optimization formulation for the maximum likelihood decoding of binary linear codes. They developed a cutting plane algorithm in which valid inequalities for cutting off non-integral solutions are iteratively added to the continuous relaxation problem. By contrast, we focus on the integer optimization decoding; that is, we solve the integer optimization problem directly by using mathematical optimization software. We evaluate the computational performance of the integer optimization decoding and suggest its potential for practical use. The computational results demonstrate that in a low-noise condition, the integer optimization decoding achieved higher bit error rate performance than did the common sum-product decoding for low-density parity-check codes.

キーワード：誤り訂正，数理最適化，整数最適化復号法，Sum-product 復号法
Key words：Error Correction, Mathematical Optimization, Integer Optimization Decoding, Sum-product Decoding

1. はじめに

音楽再生機，ゲーム機，テレビ，携帯電話などのディジタル機器は，我々の生活において必要不可欠なものとなっている。これらのディジタル機器では，アナログデータはディジタルデータ（0 と 1 の系列）に変換されて扱われる。このようなデータを通信・記録する際には，雑音（ノイズ）による誤りが発生することがある。例えば CD 表面の傷や QR コードの汚れなどが雑音の原因となり，ディジタル機器にとっては僅かな誤りでさえも深刻な問題となる。

近年では通信の高速化と高密度化が進み，通信媒体を通る信号が雑音の中にほとんど埋もれてしまうような状況が生じる。通信路自体の改良によって雑音を減少させることは可能であるが，現実としては物理的・経済的な制限により困難である場合が多い。このような状況下でも通信の高い信頼性を確保するためには，通信で生じた誤りを情報の受信側で訂正する「誤り訂正技術」が重要となる（和田山［2010］）。

図 1 に符号化通信システムの例を示す。送信者が「こんにちは」というメッセージを送信し，符号化器は 11100 という符号語に変換して通信路に送る。しかし，通信路の雑音によって受信語は 11101 となり，最後のビットに誤りが生じている。このような場合でも受信語の誤りを訂正し，「こんにちは」という正しいメッセージを推定して受信者に送ることが復号器の役割である。通信路自体の改良が困難な場合でも，誤り訂正技術では数学的な手法でメッセージを復元し，通信の信頼性を向上させるこ

受付：2017 年 11 月 27 日
受理：2017 年 11 月 27 日

とができる。

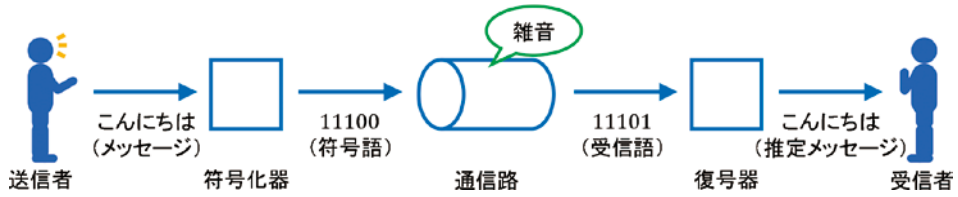

図 1　符号化通信システム

Gallager［1962］により提案された低密度パリティ検査（low-density parity-check : LDPC）符号は，疎なパリティ検査行列により定義される 2 元線形符号であり，高い誤り訂正能力を持ち，高速な復号が可能である。LDPC 符号の復号法としては，メッセージ交換処理に基づく発見的解法である sum-product 復号法（Aji and McEliece［2000］, Kschischang et al.［2001］）や min-sum 復号法（Fossorier et al.［1999］, Wiberg［1995］）が一般的に使用される。しかし，これらのアルゴリズムでは収束性や最尤復号との一致性が保証されず，また高密度のパリティ検査符号に対しては性能が悪化する。これらの欠点を克服できる方法として，数理最適化手法を利用した復号法が近年盛んに研究されている（Feldman et al.［2005］, Helmling et al.［2012］）。特に Tanatmis et al.［2010］は，最尤復号問題を整数最適化問題として定式化し，その整数制約を緩和した問題に対して，妥当不等式を追加しながら繰り返し求解する切除平面法を提案している。

近年では，数理最適化のアルゴリズムとコンピュータの性能が飛躍的に向上し，最適化問題専用のソフトウェアである最適化ソルバーを利用して，実用規模の整数最適化問題を解くことが可能になりつつある（Bixby［2012］，宮代・松井［2006］）。そこで本研究では，最適化ソルバーを用いて整数最適化問題を直接求解する整数最適化復号法に着目し，その性能と実用性を評価することを目的とする。計算機実験の結果，雑音が小さい場合には整数最適化復号法の精度は sum-product 復号法よりも優れていることが分かった。

本論文の構成は以下のようになる。2 節では 2 元線形符号について説明し，最尤復号のための整数最適化問題の定式化を提示する。3 節では計算機実験の結果を報告し，整数最適化復号法の性能を評価する。4 節では本論文のまとめと今後の課題を述べる。

2.　整数最適化復号法

本節では，まず 2 元線形符号とその復号について説明する。その後，最尤復号問題を整数最適化問題として定式化し，その連続緩和問題を提示する。

2.1　2 元線形符号

$\mathbb{F}_2^n$ を 2 元有限体上の長さ n のすべてのベクトルの集合とする。ここでは簡単に，$\mathbb{F}_2^n$ は長さ n の 2 進数のベクトルの集合と考えて良い。m 行 n 列の 2 進数の行列 $\boldsymbol{H}$ をパリティ検査行列とし，2 元線形符号 C はパリティ検査式 $\boldsymbol{Hc} = \boldsymbol{0}$ を満たす符号語 $\boldsymbol{c} \in \mathbb{F}_2^n$ の集合として定義される：

$$C := \left\{ \boldsymbol{c} \in \mathbb{F}_2^n \middle| \boldsymbol{Hc} = \boldsymbol{0} \right\}.$$

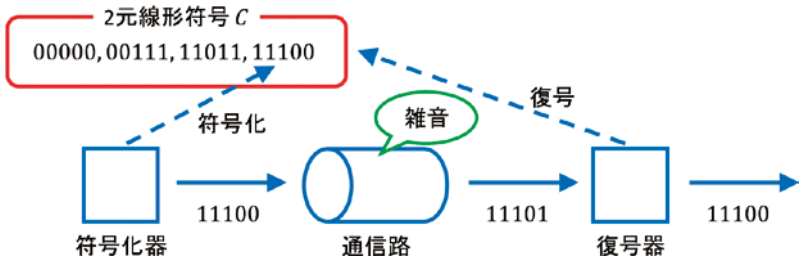

図 2　2 元線形符号の復号

例として $(m, n) = (3, 5)$ とし，以下のパリティ検査行列を考える：

$$\boldsymbol{H} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

このとき，2 元線形符号は以下のようになる：

$$C = \left\{ (0,0,0,0,0)^\top, (0,0,1,1,1)^\top, (1,1,0,1,1)^\top, (1,1,1,0,0)^\top \right\}.$$

2 進数の演算であることに注意すれば，上記の符号語がすべてパリティ検査式を満たすことが確認できる。

図 2 に 2 元線形符号の復号の例を示す。まず符号化器は，送信するメッセージに対応する 2 元線形符号の符号語 11100 を通信路に送る。通信路の雑音によって誤りが生じ，受信語は 11101 となるが，この受信語はパリティ検査式を満たさないために，復号器では受信語に誤りが含まれることを検出できる。さらに，2 元線形符号から受信語に近い符号語を見つけて変換することで，誤りを訂正できる可能性がある。図 2 の例では，受信語 11101 を 1 ビットだけ異なる符号語 11100 に変換することで，元の符号語に復号することができる。

2.2　整数最適化問題

$\boldsymbol{x} := (x_1, x_1, x_2, \ldots x_n)^\top \in \{0,1\}^n$ を符号語を表す 0−1 決定変数のベクトルとする。$\mathbb{Z}_+^m$ を長さ m のすべての非負整数ベクトルの集合とし，$\boldsymbol{z} := (z_1, z_2, \ldots, z_m)^\top \in \mathbb{Z}_+^m$ を非負整数の決定変数のベクトルとする。このとき符号語 $\boldsymbol{x}$ に対するパリティ検査式は，10 進数として $\boldsymbol{Hx}$ を計算し，要素がすべて 2 の倍数であることと等価なので，以下のように書くことができる：

$$\boldsymbol{Hx} = 2\boldsymbol{z}.$$

ベクトル $\boldsymbol{y} := (y_1, y_2, \ldots, y_n)^\top$ を受信語とし，ベクトル $\boldsymbol{\lambda} := (\lambda_1, \lambda_2, \ldots, \lambda_n)^\top$ を対数尤度比

$$\lambda_j := \log\left(\frac{\Pr(y_j | x_j = 0)}{\Pr(y_j | x_j = 1)} \right) \quad (j = 1, 2, \ldots, n)$$

とする。このとき最尤復号は，$\boldsymbol{\lambda}^\top \boldsymbol{x}$ の最小化と等しいことが知られている（Feldman et al. [2005], Helmling et al. [2012]）。対数尤度比 λ_j は受信語 y_j が 0 に近いとき正値をとり，受信語 y_j が 1 に近いとき負値をとる。さらに $x_j \in \{0,1\}$ を考慮すると，$\boldsymbol{\lambda}^\top \boldsymbol{x}$ の最小化は，受信語 y_j が 0 に近い場合には $x_j = 0$ とし，受信語 y_j が 1 に近い場合には $x_j = 1$ とする，すなわち受信語と近い符号語を求めることに対応

する。

以上をまとめると，パリティ検査式を満たし，受信語に近い符号語を求める最尤復号問題は，以下の整数最適化問題として定式化できる（Tanatmis et al. [2010]）:

$$\text{最小化} \quad \boldsymbol{\lambda}^{\top}\boldsymbol{x} \tag{1}$$

$$\text{制約条件} \quad \boldsymbol{H}\boldsymbol{x} = 2\boldsymbol{z}, \tag{2}$$

$$\boldsymbol{x} \in \{0,1\}^{n}, \quad \boldsymbol{z} \in \mathbb{Z}_{+}^{m}. \tag{3}$$

上記の整数最適化問題の整数制約を緩和すると，以下の連続緩和問題が得られる：

$$\text{最小化} \quad \boldsymbol{\lambda}^{\top}\boldsymbol{x} \tag{4}$$

$$\text{制約条件} \quad \boldsymbol{H}\boldsymbol{x} = 2\boldsymbol{z}, \tag{5}$$

$$\boldsymbol{0} \leq \boldsymbol{x} \leq \boldsymbol{1}, \quad \boldsymbol{z} \geq \boldsymbol{0}. \tag{6}$$

連続緩和問題の最適解はパリティ検査式を満たすことは保証されないが，線形最適化問題であるために効率的に求解することが可能である。

3. 計算機実験

本節では計算機実験を通して，整数最適化復号法のビット誤り率と計算時間を検証する。

3.1 実験の設定

Encyclopedia of Sparse Graph Codes（http://www.inference.phy.cam.ac.uk/mackay/codes/data.html）からパリティ検査行列を取得し，4 種類の LDPC 符号に対して実験を実施した（表 1）。ただし，m と n はそれぞれ検査行列の行数（パリティ検査式の数）と列数（符号語の長さ）を表す。

以降では，以下の手法の性能を比較する：

整数最適化　整数最適化問題（1）-（3）の求解による復号

連続緩和　連続緩和問題（4）-（6）の求解による復号

sum-product　sum-product 復号法

整数最適化問題 (1)-(3) と連続緩和問題 (4)-(6) はモデリング言語 ZIMPL（Koch [2004], 高野[2016]）と最適化ソルバー SCIP（Achterberg [2009]）を使用して求解した。また，sum-product 復号法は LDPC 符号関係の公開プログラム（https://dl.dropboxusercontent.com/u/1820240/supportpages/LDPCpublic.html）から取得した Java プログラムを使用し，最大反復回数は 10 回に設定した。

送信語は $\boldsymbol{x} = \boldsymbol{0}$ とし，バイナリ-バイポーラ変換を施して $\boldsymbol{s} = \boldsymbol{1} - 2\boldsymbol{x}$ とした。平均 0，標準偏差 σ の独立な正規乱数による雑音ベクトル $\boldsymbol{\varepsilon}$ を加えて，受信語を $\boldsymbol{y} = \boldsymbol{s} + \boldsymbol{\varepsilon}$ とし，対数尤度比は $\boldsymbol{\lambda} = 2\boldsymbol{y}/\sigma^2$ と

表 1　LDPC 符号

LDPC 符号	m	n	検査行列
48x96a	48	96	96.3.963
48x96b	48	96	96.33.964
102x204a	102	204	204.33.484
102x204b	102	204	204.33.486

設定した。通信の信頼性（雑音の小ささ）を表す SN 比は $1/(2\sigma^2 R)$ と定義され，R は符号化率（$R = 0.5$）である。SN 比は 0.4，0.6，0.8，1.0，1.2，1.4，1.6 の 7 種類の値に設定した。復号した符号語を $\hat{\boldsymbol{x}} := (\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n)^\top \in \{0,1\}^n$ とするとき，ビット誤り率は $\left(\sum_{j=1}^{n} |x_j - \hat{x}_j|\right) / n$ とし，以降の実験結果では，雑音の正規乱数の生成と復号を 5 回繰り返した平均値を示す。

3.2　ビット誤り率

図 3 に各復号法のビット誤り率を示す。整数最適化復号法は，SN 比が 1.0 以上の場合には他の手法と比較してビット誤り率が最も低いが，SN 比が 0.6 以下の場合には他の手法よりもビット誤り率が高

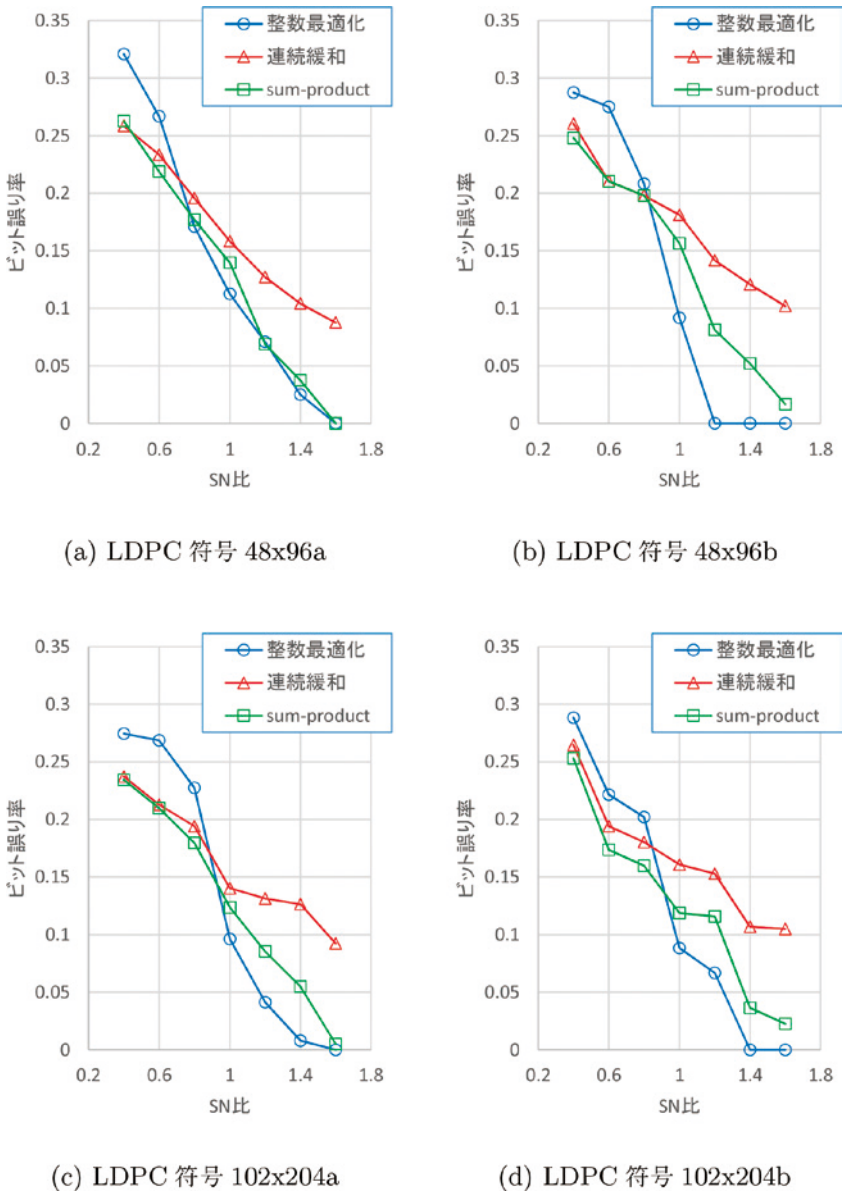

図 3　各復号法のビット誤り率

いという結果になった。連続緩和による復号は，SN 比が大きい場合には他の手法と比較してビット誤り率が高かった。sum-product 復号法は，SN 比が大きい場合にはビット誤り率は整数最適化復号法に劣るが，SN 比が小さい場合には他の手法よりもビット誤り率が低かった。

以上の結果をまとめると，雑音が小さい場合には整数最適化復号法の精度が最も優れており，雑音が大きい場合には連続緩和による復号や sum-product 復号法の精度が優れていると言える。雑音が小さい場合には，最尤復号問題を正確に求解することで高精度の復号が可能であるために，整数最適化復号法の精度が優れていたと考えられる。一方で雑音が大きい場合には，元の符号語と受信語が大きく異なる。このために，パリティ検査式を満たすように復号しても誤りを訂正できず，むしろ誤りを大きくしてしまう可能性があるために，整数最適化復号法の精度が悪化したと考えられる。

3.3 計算時間

表 2 に各復号法の計算時間を示す。整数最適化復号法は SN 比が小さくなるにつれて計算時間が増大した。(m, n) = (48, 96) の場合には数秒程度の計算時間であるが，(m, n) = (102, 204) の場合には

表 2　各復号法の計算時間（秒）

LDPC 符号	SN 比	整数最適化	連続緩和	sum-product
48x96a	0.4	5.328	0.010	1.642
	0.6	4.136	0.002	1.750
	0.8	3.818	0.004	1.640
	1.0	2.430	0.008	1.392
	1.2	1.600	0.012	1.126
	1.4	0.108	0.006	1.195
	1.6	0.030	0.012	0.983
48x96b	0.4	6.330	0.012	1.305
	0.6	9.272	0.014	1.795
	0.8	16.310	0.002	1.611
	1.0	7.654	0.004	1.506
	1.2	0.888	0.004	1.451
	1.4	1.586	0.008	1.439
	1.6	0.070	0.016	1.132
102x204a	0.4	2907.608	0.004	2.381
	0.6	3274.004	0.010	3.433
	0.8	2310.262	0.020	2.342
	1.0	819.942	0.004	2.372
	1.2	395.410	0.012	2.211
	1.4	17.494	0.014	2.291
	1.6	0.436	0.004	2.032
102x204b	0.4	2641.390	0.006	2.542
	0.6	3457.798	0.008	3.034
	0.8	2862.886	0.014	2.564
	1.0	332.084	0.000	2.339
	1.2	222.588	0.006	2.258
	1.4	3.400	0.006	1.776
	1.6	0.114	0.014	2.249

最長で 1 時間程度の計算時間を要した。対照的に連続緩和による復号は，どの場合でも 0.02 秒以下の計算時間であり，他の手法と比較して計算時間が非常に短かった。sum-product 復号法は，$(m, n) = (48, 96)$ の場合には 1〜2 秒程度，$(m, n) = (102, 204)$ の場合には 2〜3 秒程度の計算時間であった。

ビット誤り率の結果も考慮すると，雑音が小さい場合には整数最適化復号法の精度が最も優れており，雑音が小さくなるほどその計算時間は減少する。一方で雑音が大きい場合には，整数最適化復号法の精度は悪化し，計算時間も増大する。雑音が大きい場合には，計算時間に関しては連続緩和による復号が，復号の精度に関しては sum-product 復号法が優れていると言える。

4. おわりに

本論文では，最適化ソルバーを用いて整数最適化問題を直接求解する整数最適化復号法に着目し，計算機実験を実施してその性能と実用性を評価した。実験結果から，雑音が小さい場合には整数最適化復号法の精度が優れており，その計算時間は雑音が小さくなるほど減少することが分かった。

LDPC 符号に対して一般的に使用される sum-product 復号法と比較すると，整数最適化復号法は計算に時間がかかり，現時点では実用規模の復号問題を短時間で求解することは困難であると考えられる。しかしながら本研究で検証したように，雑音がある程度小さい場合には整数最適化復号法の精度は sum-product 復号法よりも優れており，今後も数理最適化のアルゴリズムとコンピュータの性能が向上していけば，整数最適化復号法は実用的かつ高性能な復号法となることが期待される。

本研究では最適化ソルバーを用いて整数最適化問題を求解したが，今後の課題としては，整数最適化問題を効率的に求解するためのアルゴリズムの開発が挙げられる。

参考文献

Achterberg, T., “SCIP: Solving Constraint Integer Programs”, *Mathematical Programming Computation*, Vol. 1, No. 1, 2009, pp. 1-41.

Aji, S.M. and McEliece, R.J., “The Generalized Distributive Law”, *IEEE Transactions on Information Theory*, Vol. 46, No. 2, 2000, pp. 325-343.

Bixby, R.E., “A Brief History of Linear and Mixed-integer Programming Computation”, *Documenta Mathematica, Extra Volume: Optimization Stories*, 2012, pp. 107-121.

Feldman, J., Wainwright, M.J. and Karger, D.R., “Using Linear Programming to Decode Binary Linear Codes”, *IEEE Transactions on Information Theory*, Vol. 51, No. 3, 2005, pp. 954-972.

Fossorier, M.P., Mihaljevic, M. and Imai, H., “Reduced Complexity Iterative Decoding of Low-density Parity Check Codes Based on Belief Propagation”, *IEEE Transactions on Communications*, Vol. 47, No. 5, 1999, pp. 673-680.

Gallager, R., “Low-density Parity-check Codes”, *IRE Transactions on Information Theory*, Vol. 8, No. 1, 1962, pp. 21-28.

Helmling, M., Ruzika, S. and Tanatmis, A., “Mathematical Programming Decoding of Binary Linear Codes: Theory and Algorithms”, *IEEE Transactions on Information Theory*, Vol. 58, No. 7, 2012, pp. 4753-4769.

Koch, T., “Rapid Mathematical Programming”, ZIB Report 04-58, Konrad-Zuse-Zentrum für Informationstechnik Berlin, 2004.

Kschischang, F.R., Frey, B.J. and Loeliger, H.A., “Factor Graphs and the Sum-product Algorithm”, *IEEE Transactions on Information Theory*, Vol. 47, No. 2, 2001, pp. 498-519.

Tanatmis, A., Ruzika, S., Hamacher, H.W., Punekar, M., Kienle, F. and Wehn, N., “A Separation Algorithm for Improved LP-decoding of Linear Block Codes”, *IEEE Transactions on Information Theory*, Vol. 56, No. 7, 2010, pp. 3277-3289.

Wiberg, N., “Codes and Decoding on General Graphs”, *European Transactions on Telecommunications*, Vol. 6, No. 5, 1995, pp. 513-526.

高野祐一,「ZIMPL 言語と SCIP による数理最適化」,『専修ネットワーク＆インフォメーション』, 第 24 号, 2016 年 3 月, 9-14 ページ.
宮代隆平, 松井知己,「ここまで解ける整数計画」,『システム / 制御 / 情報』, 第 50 巻, 第 9 号, 2006 年 9 月, 363-368 ページ.
和田山正,『誤り訂正技術の基礎』, 森北出版, 2010 年.