噴水符号の基礎と構成

野崎隆之

山口大学

高機能暗号とプライバシー保護情報分析の基礎数理
2016/9/6

概要

パケット通信 (特に 1 対多通信) 向けの誤り訂正符号である噴水符号の紹介

- 基礎 (応用先，望まれる条件 etc)
- 構成法 (LT 符号，Raptor 符号)
- 最近の研究 (時間があれば...)

お詫び

- 代数的な話はでてきません (代数的符号理論ではない)
- 暗号・セキュリティで利用している研究は少ない

坂下，古賀，本庄，“LT 符号を用いた高速な線形ランプ型しきい値分散法の検討,” 信学技報 IT2013–85

武井，古賀，“JPEG 画像の消失部分を復元できる電子透かしの提案と性能評価,” 信学技報 IT2015–124

参考文献

1. A. Shokrollahi, M. Luby, “Raptor Codes,” Foundations and Trends in Communications and Information Theory, vol. 6, nos. 3–4, pp. 213–322, 2011
   - 噴水符号についてまとめられた書籍
   - 内容：パケット通信のモデル化，噴水符号の基礎，LT 符号・Raptor 符号の構成，規格
2. J. Byers, M. Luby, and M. Mitzenmacher, “A digital fountain approach to asynchronous reliable multicast,” IEEE Journal on Selected Areas in Communications, vol. 20, no. 8, pp. 1528-1540, 2002.
   - 噴水符号のコンセプトが与えられた論文
3. 萩原学 編著 “進化する符号理論,” 数学評論社 (2016/9/9 発売)
   - 噴水符号の基礎について書かれた章が含まれています

パケット通信のモデル化

パケット

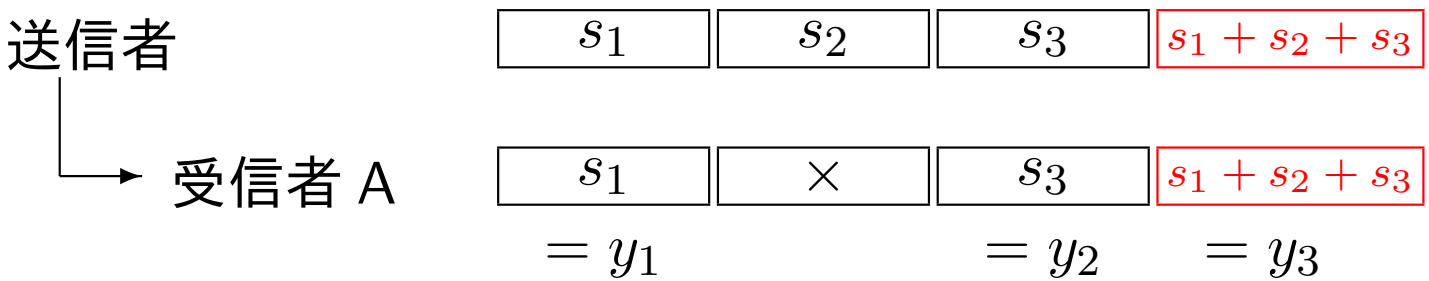
ネットワーク中で伝送されるデータのかたまり

$\boldsymbol{s}_i = (s_{i1}, s_{i2}, \ldots, s_{i\ell}) \in \{0,1\}^{\ell}$

$\boldsymbol{s} = (\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_k)$

- 送信するデータを複数個のパケットに分けて送る (数千〜数万)
- ネットワーク中で一部のパケットが破損 (ロス) する

  $\Rightarrow$ 符号化を利用して復号



$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \\ s_3 \end{pmatrix}$$

マルチキャストで要求される条件

マルチキャスト

1 対多通信 (1 人の送信者，複数の受信者)
(例) データ配信，インターネット放送 etc

符号に要求される条件

1. 全ての受信者が正しくファイルを受け取れる
2. 任意のパケットロス率に対応している
   (受信者ごとに通信状況が異なる)
3. 任意のタイミングで受信を開始しても良い
4. 少ない受信パケットで元のファイルを復号できる
   (理想的には k 個の受信パケットで復号できる)
5. 復号計算量が少ない
   (理想的には $\mathcal{O}(k)$ の計算量で復号できる)

噴水符号のコンセプト

噴水符号では，条件を満たすために，以下のような手法を採る

- 送信者は k 個のパケットを符号化し，十分大きな数の符号化パケットを作成する
- 受信者は任意の r 個 $(r \geq k)$ の符号化パケットを受信し，復号する

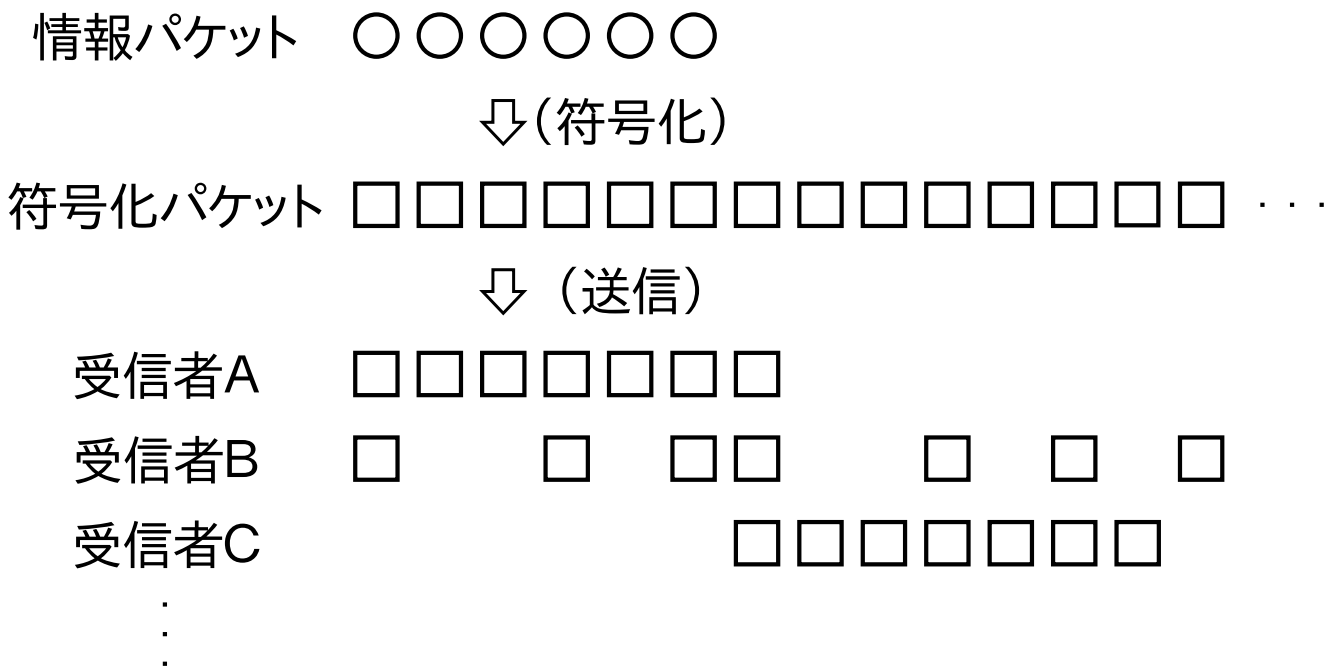


Figure: 噴水符号の符号化と受信の様子

(小話) 噴水符号の名前の由来

噴水符号 (Fountain code) の特徴

どのパケットを受け取るかは重要でなく，一定の個数のパケットを受信すれば良い

噴水でコップに水をためる場合

どの水滴を集めるかは重要ではなく，一定量の水が集められれば良い

噴水符号の符号化

- 情報パケット：$\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_k \in \mathbb{F}_2^\ell$
  (Note: 拡大体 $\mathbb{F}_2$ 上のベクトルと定義しても良い)
- 符号化パケット：$\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots$

符号化

$$\boldsymbol{x}_t = a_{t,1}\boldsymbol{s}_1 + a_{t,2}\boldsymbol{s}_2 + \cdots + a_{t,k}\boldsymbol{s}_k$$

但し，$a_{t,1}, a_{t,2}, \ldots, a_{t,k} \in \mathbb{F}_2$

$$\begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_k \\ \vdots \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,k} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,k} \\ \vdots & \vdots & \vdots & \vdots \\ a_{k,1} & a_{k,2} & \cdots & a_{k,k} \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} \boldsymbol{s}_1 \\ \boldsymbol{s}_2 \\ \vdots \\ \boldsymbol{s}_k \end{pmatrix}$$

噴水符号の復号

ある受信者が $i_1, i_2, \ldots, i_r$ 番目の符号化パケットを受信したと仮定
$\Rightarrow (\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_r})$ は既知

$$\begin{pmatrix} \boldsymbol{x}_{i_1} \\ \boldsymbol{x}_{i_2} \\ \vdots \\ \vdots \\ \boldsymbol{x}_{i_r} \end{pmatrix} = \begin{pmatrix} a_{i_1,1} & a_{i_1,2} & \cdots & a_{i_1,k} \\ a_{i_2,1} & a_{i_2,2} & \cdots & a_{i_2,k} \\ \vdots & \vdots & \vdots & \vdots \\ a_{i_r,1} & a_{i_r,2} & \cdots & a_{i_r,k} \end{pmatrix} \begin{pmatrix} \boldsymbol{s}_1 \\ \boldsymbol{s}_2 \\ \vdots \\ \boldsymbol{s}_k \end{pmatrix}$$

線形方程式を解く問題に帰着される

噴水符号の設計の指針

以下を満たすような 行列 $\mathbf{A} = (a_{i,j})$ の設計
(+ 効率的な線形方程式の解法)

(1)' 高い確率でもとの情報を得ることができる

(4) 少ない受信パケットで元のファイルを復号できる
(理想的には k 個の受信パケットで復号できる)

(5) 復号計算量が少ない
(理想的には $\mathcal{O}(k)$ の計算量で復号できる)

本講演で取り扱わない $\mathbf{A}$

Vandermonde 行列 (RS 符号)

- 十分大きな位数の拡大体 $\mathbf{F}_q$ を利用
- k 個の受信パケットで必ず復号できる
- 復号計算量 $\mathcal{O}(k^2)$ (効率のよいアルゴリズムを使うと $\mathcal{O}(q(\log_2 q)^2)$)
- q が大きくなると消費電力大

密なランダム行列

- 高い確率でフルランク $\Rightarrow$ 高い確率で復号可能
- 復号計算量 $\mathcal{O}(k^3)$ (ガウスの消去法)

LT 符号

- $\mathbf{A}$ として疎なランダム行列を利用
- 復号では後退代入のみを利用 (計算量 $\mathcal{O}(k)$, peeling 復号)
- $k(1+\alpha_k)$ 個の受信パケットから高い確率で復号可能

LT 符号の符号化

パラメタ：次数分布 $\Omega(x) = \sum_{i=1}^{D} \Omega_i x^i$
$(\Omega(1) = \sum_i \Omega_i = 1)$

1. 整数 d を確率 Ω_d で選ぶ
2. 非ゼロ要素数が d 個の長さ k の二元ベクトル $(a_{t,1}, \ldots, a_{t,k})$ を作成
   (相異なる d 個のパケットを選び出す)
3. $\boldsymbol{x}_t = a_{t,1}\boldsymbol{s}_1 + \cdots + a_{t,k}\boldsymbol{s}_k$
   (選んだパケットを足し合わせる)

次数分布 $\Omega(x)$ の設計によって性能 (overhead α_k，復号確率) が変化

LT 符号の簡単な例 (符号化)

$k = 5$, $(\Omega_1, \Omega_2, \Omega_3) = (1/3, 1/2, 1/6)$ を仮定

$$\begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \boldsymbol{x}_3 \\ \boldsymbol{x}_4 \\ \boldsymbol{x}_5 \\ \boldsymbol{x}_6 \\ \vdots \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} \boldsymbol{s}_1 \\ \boldsymbol{s}_2 \\ \boldsymbol{s}_3 \\ \boldsymbol{s}_4 \\ \boldsymbol{s}_5 \end{pmatrix}$$

LT 符号の簡単な例 (復号)

$r = 6$ で $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4, \boldsymbol{x}_5, \boldsymbol{x}_6$ が受信できたと仮定

$$\begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \boldsymbol{x}_3 \\ \boldsymbol{x}_4 \\ \boldsymbol{x}_5 \\ \boldsymbol{x}_6 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{s}_1 \\ \boldsymbol{s}_2 \\ \boldsymbol{s}_3 \\ \boldsymbol{s}_4 \\ \boldsymbol{s}_5 \end{pmatrix}$$

$$\Longleftrightarrow \begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \boldsymbol{x}_3 + \boldsymbol{x}_4 \\ \boldsymbol{x}_4 \\ \boldsymbol{x}_5 + \boldsymbol{x}_4 \\ \boldsymbol{x}_6 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{s}_1 \\ \boldsymbol{s}_2 \\ \boldsymbol{s}_3 \\ \boldsymbol{s}_4 \\ \boldsymbol{s}_5 \end{pmatrix} \quad \vdots$$

LT 符号の復号と性質

- 重み 1 (非ゼロ要素数が 1) の行が重要 (復号開始・途中)
- 重み 1 の行がなくなると復号終了

$\Rightarrow \Omega_1 > 0$ である必要がある
ところが以下の事実が知られている

ガウスの消去法で高い確率で復号可能で $\alpha_k \to 0 (k \to \infty)$ にするには,
Ω_1 を 0 に収束させる必要がある

Overhead を 0 にすると peeling 復号できない
$\Longleftrightarrow$ Overhead と復号計算量の両立が困難

Raptor 符号

- LT 符号の拡張のひとつ (事前符号化と呼ばれる処理が追加される)
- 任意に小さな α を復号計算量 $\mathcal{O}(k)$ で達成

線形符号 (事前符号化の準備)

パリティ検査行列 $\mathbf{H} \in \mathbb{F}^{(n-k)\times n}$ で以下のとおりに定義される

$$\mathcal{C} := \{\boldsymbol{c} = (c_1, c_2, \ldots, c_n) \mid \mathbf{H}\boldsymbol{c} = \mathbf{0}\}$$

符号化

$$\begin{aligned} \phi : \quad & \mathbb{F}_2^k \to \mathcal{C} \\ & (s_1, s_2, \ldots, s_k) \mapsto (c_1, c_2, \ldots, c_n) \end{aligned}$$

符号化率 (レート) : k/n

Raptor 符号の符号化

パラメタ：事前符号 $\mathcal{C}$, 次数分布 $\Omega(x)$

1. (事前符号化) 符号 $\mathcal{C}$ を用いて，情報パケット $(s_1, s_2, \ldots, s_k)$ を事前符号化パケット $(c_1, c_2, \ldots, c_n)$ に符号化
2. $(c_1, c_2, \ldots, c_n)$ を LT 符号で符号化
   1. 整数 d を確率 Ω_d で選ぶ
   2. 非ゼロ要素数が d 個の長さ k の二元ベクトル $(a_{t,1}, \ldots, a_{t,k})$ を作成 (相異なる d 個のパケットを選び出す)
   3. $\boldsymbol{x}_t = a_{t,1}\boldsymbol{s}_1 + \cdots + a_{t,k}\boldsymbol{s}_k$ (選んだパケットを足し合わせる)

- 事前符号化 $+$ LT 符号
- 事前符号化は高符号化率な LDPC 符号がよく用いられる ($\mathcal{O}(n)$ で復号可能で復号性能が良いため)
- 性能 (Overhead，復号確率) は $\mathcal{C}$,$\Omega(x)$ に依存

Raptor 符号の行列表現

$\mathcal{C}$ のパリティ検査行列 $\mathbf{H} = (h_{i,j})$ と書く

$$\begin{pmatrix} 0 \\ \vdots \\ 0 \\ \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_n \\ \vdots \end{pmatrix} = \begin{pmatrix} h_{1,1} & \cdots & h_{1,n} \\ \vdots & \ddots & \vdots \\ h_{n-k,1} & \cdots & h_{n-k,n} \\ a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{n,1} & \cdots & a_{n,n} \\ \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} \boldsymbol{c}_1 \\ \vdots \\ \boldsymbol{c}_n \end{pmatrix}$$

Raptor 符号の復号

ある受信者が $i_1, i_2, \ldots, i_r$ 番目の符号化パケットを受信したと仮定

$$\begin{pmatrix} 0 \\ \vdots \\ 0 \\ \boldsymbol{x}_{i_1} \\ \vdots \\ \boldsymbol{x}_{i_r} \end{pmatrix} = \begin{pmatrix} h_{1,1} & \cdots & h_{1,n} \\ \vdots & \ddots & \vdots \\ h_{n-k,1} & \cdots & h_{n-k,n} \\ a_{i_1,1} & \cdots & a_{i_1,n} \\ \vdots & \ddots & \vdots \\ a_{i_r,1} & \cdots & a_{i_r,n} \end{pmatrix} \begin{pmatrix} \boldsymbol{c}_1 \\ \vdots \\ \boldsymbol{c}_n \end{pmatrix}$$

後退代入 (peeling 復号法) で復号 (疎行列なので $\mathcal{O}(n)$)

次数分布の最適化

漸近解析

Overhead α が与えられた時，次式を最大化する $\Omega(x)$ が最適

$$\sup\{x \in [0,1) \mid 1 - x - \exp[-(1+\alpha)\Omega'(x)] > 0\}$$

(復号中に発生する重み 1 の行数を解析すると導出できる)

k が有限の時

ある $\gamma > 0$ に対して，次を満たす $\Omega(x)$ を利用

$$\forall x \in [0, 1-\delta] \qquad 1 - x - \exp[-(1+\alpha)\Omega'(x)] > \gamma\sqrt{\frac{1-x}{k}}$$

(漸近解析の結果 + ヒューリスティック)

Inactivation 復号 (1)

復号問題： $\boldsymbol{x}$ が既知, $\mathbf{A}$ が疎

$$\boldsymbol{x} = \mathbf{A}\boldsymbol{s}$$

Inactivation decoding

- $\mathbf{A}$ が疎であることを利用した線形方程式の解法
- $\mathbf{A}$ に列置換・行置換を施すことで解きやすい形に変形
  (疎性は変化しない)

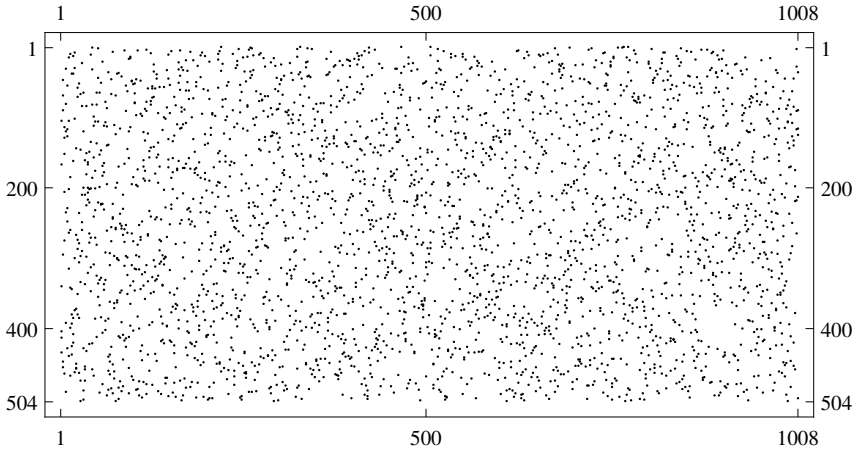
$$\begin{aligned} & \mathbf{P}\boldsymbol{x} = (\mathbf{PAQ})[\mathbf{Q}^{-1}\boldsymbol{s}] \\ \Longleftrightarrow\ & \tilde{\boldsymbol{x}} = (\mathbf{PAQ})\tilde{\boldsymbol{s}} \end{aligned}$$

  $\mathbf{P}, \mathbf{Q}$：置換行列

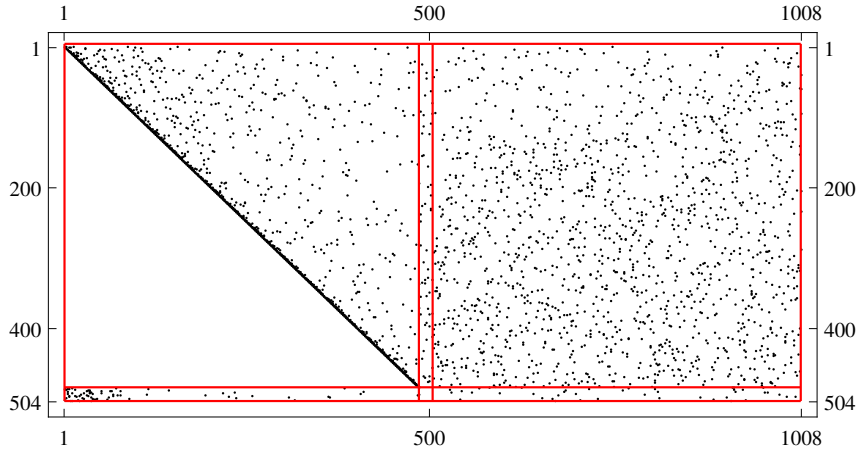
- $\mathbf{PAQ}$ を Approximate 三角行列にする

Inactivation 復号 (2:Approximate 三角行列)

(図がやや不適切ですが...)



$\Rightarrow$



$$\boldsymbol{x} = (\mathbf{PAQ})\boldsymbol{s}$$

$$\iff \begin{pmatrix} \boldsymbol{x}_u \\ \boldsymbol{x}_l \end{pmatrix} = \begin{pmatrix} \mathbf{T} & \mathbf{C} \\ \mathbf{D} & \mathbf{E} \end{pmatrix} \begin{pmatrix} \boldsymbol{s}_u \\ \boldsymbol{s}_l \end{pmatrix}$$

$$\iff \begin{pmatrix} \mathbf{T}^{-1} & \mathbf{O} \\ \mathbf{D}\mathbf{T}^{-1} & -\mathbf{I} \end{pmatrix} \begin{pmatrix} \boldsymbol{x}_u \\ \boldsymbol{x}_l \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \mathbf{T}^{-1}\mathbf{C} \\ \mathbf{O} & \mathbf{D}\mathbf{T}^{-1}\mathbf{C} - \mathbf{E} \end{pmatrix} \begin{pmatrix} \boldsymbol{s}_u \\ \boldsymbol{s}_l \end{pmatrix} \quad \text{(前方消去)}$$

Inactivation 復号 (3: 復号の手続き)

$$\begin{pmatrix} \boldsymbol{y}_u \\ \boldsymbol{y}_l \end{pmatrix} := \begin{pmatrix} \mathbf{T}^{-1} & \mathbf{O} \\ \mathbf{D}\mathbf{T}^{-1} & -\mathbf{I} \end{pmatrix} \begin{pmatrix} \boldsymbol{x}_u \\ \boldsymbol{x}_l \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \mathbf{T}^{-1}\mathbf{C} \\ \mathbf{O} & \mathbf{D}\mathbf{T}^{-1}\mathbf{C} - \mathbf{E} \end{pmatrix} \begin{pmatrix} \boldsymbol{s}_u \\ \boldsymbol{s}_l \end{pmatrix}$$

1. 左辺の計算 (前方消去と一緒にできる)
   1. $\boldsymbol{y}_u$ を計算 (後退代入を利用)

$$\boldsymbol{y}_u = \mathbf{T}^{-1}\boldsymbol{x}_u \quad \Longleftrightarrow \quad \mathbf{T}\boldsymbol{y}_u = \boldsymbol{x}_u$$

   2. $\boldsymbol{y}_l$ を計算

$$\boldsymbol{y}_l = \mathbf{D}\boldsymbol{y}_u - \boldsymbol{x}_l$$

2. $\boldsymbol{s}$ の計算

$$\begin{aligned} \boldsymbol{s}_u &= \boldsymbol{y}_u \\ \boldsymbol{s}_l &= (\mathbf{D}\mathbf{T}^{-1}\mathbf{C} - \mathbf{E})^{-1}\boldsymbol{y}_l \end{aligned}$$

ここまでのまとめ

話の内容

- 噴水符号の目的・コンセプト・符号化・復号
- LT 符号の符号化・復号
- Raptor 符号の符号化・復号
- Inactivation 復号

議論したい内容

$$\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$$

- 早く解けるアルゴリズムは?
- どのように $\mathbf{A}$ を構成すればよいか?

時間があれば...

- 最近やった研究