計算時間，信頼性，情報秘匿性を考慮した分散符号化計算方式に関する研究

A Study on Distributed Coded Computation Methods Considering Computation Times, Error-Correcting Capabilities and Information Securities

2022年2月

風間　皐希
Koki KAZAMA

計算時間，信頼性，情報秘匿性を考慮した分散符号化計算方式に関する研究

A Study on Distributed Coded Computation Methods Considering Computation Times, Error-Correcting Capabilities and Information Securities

2022年2月

早稲田大学大学院　基幹理工学研究科
数学応用数理専攻　情報理論研究

風間　皐希
Koki KAZAMA

目次

図目次

第 1 章

序論

1.1 研究背景

大規模データを扱う機会が増えた現代社会において，高速なデータ処理方式の一つである分散計算方式の必要性がいっそう高まっている．この方式は，複数の計算機（ワーカー）にデータの一部を分散し，並列計算を行う方式である．これには，下記の (a) という利点がある一方で，(b),(c) といった欠点もある．

(a) システムの総計算時間を少なくする．
(b) ワーカーの故障等により，マスターが一部のワーカーの計算結果の値を受信しない，あるいは誤った値を受信する可能性が高まる．本論文では，ワーカーの計算結果の値を受信しないことを消失が発生したと言い，誤った値を受信することを誤りが発生したと言う．
(c) データに個人情報などユーザの秘密情報を含む場合，データを分散することで秘密情報が多くのワーカーに漏洩する可能性が高まる．

本論文では，利点 (a) を有しつつ，欠点 (b) や (c) を解消する分散計算方式を論じる．本論文では，(b) のような誤り，消失に対する耐性を信頼性，(c) のような情報漏洩に対する耐性を秘匿性と呼ぶ．ただし，本論文では，分散計算方式をある側面から抽象化して単純にしたうえで，上記のような分散することによる利点，欠点を理論的に論じるため，多くの制約を置いていない抽象化した単純な計算機や方式を用いていることに注意する．本論文で論じる内容を大別すると，第 4 章で論じる (a) と (b) を考慮した方式の提案と，第 5 章で論じる (a)，(b)，(c) をすべて同時に考慮した方式の提案の 2 つに分かれる．(a) と (b) を考慮した方式は，従来から分散符号化計算方式 [1]，[2]，[3]，[4] という方式が提案されているが，本論文では誤り消失訂正の範囲を拡張した方式を提案する．一方，(a)，(b)，(c) をすべて同時に考慮した方式として，

本論文では新たに「グループ型分散秘匿符号化計算方式」を提案する．従来，(b) と (c) を考慮した方式として分散秘匿符号化計算方式 [5] が論じられてきたが，(a) を考慮しない点で本論文と異なる．本論文では，特に，有限体 $\mathbb{F}_q$ 上の巨大な $k_{\mathrm{A}} \times l$ 行列 $\boldsymbol{A}$ と $l \times k_{\mathrm{B}}$ 行列 $\boldsymbol{B}$ の積行列 $\boldsymbol{AB}$ の計算方式を論じる．

本論文は，分散計算方式をある側面から抽象化して単純にしたうえで，一部のデータを複数のワーカーに分散する計算方式における上記の利点 (a)，欠点 (b),(c) を論じることが目標である．例えば，分散することによって，システムの総計算時間が少なくなるという利点が生じる代わりに，誤りが発生する可能性が高まるという欠点が生じる，といった具合である．現実の計算方式に近づけるため，計算機の構造上の制約，例えばメモリ制約や通信量制約[*1]など，より多くの計算機や方式の制約を取り込んだ理論研究もありうる．しかし，その前段階の理論研究である本論文では，抽象化した単純で理論的な計算機や方式を用いるものとし，それらに対し，あまり多くの制約をおいていない．本論文では，計算機には，関数を計算する所定のプログラム等を保存しており，有限体上の行列やベクトルの保存および有限体上の演算の計算が可能であるという仮定を置いている．例えば，実際のコンピュータを複数個接続して，その中の一つをマスター，他をワーカーとした方式を単純化したものを，本論文の方式のモデルとして想定している．そのほかにも，コンピュータ内の CPU や複数個の GPU を接続して，主要な CPU をマスター，GPU をワーカーとした方式を単純化したものを，本論文の方式のモデルとして想定している．本論文にさらに諸制約を加えて，現実の計算機，方式に近いものにしていくことが考えられる．

有限体上の巨大行列の積計算は，例えば誤り訂正符号の符号化において必要になる．例えば，最近の誤り訂正符号でよく利用される LDPC 符号などでは，符号化するための生成行列や，送信する情報ベクトルは巨大であることが多い．具体的には，[6] においては，符号長 8388864，次元 4194304 の LDPC 符号を利用している．この場合の生成行列は，$\mathbb{F}_2$ 上 4194304×8388864 行列である．実用面においては，衛星デジタル放送システムでは，LDPC 符号のほかに，BCH 符号という符号もよく利用される．例えば，DVB-S2 規格においては，符号長 64800，次元 38800 の LDPC 符号が使用され，また，ISDB-S3 規格においては，符号長 65535，次元 65343 の BCH 符号が用いられる [7].

また，有限体上の巨大行列の積計算を目標とする分散符号化計算方式は理論的に単純で扱いやすく，その考え方を発展させることにより，ほかの計算へ応用させることが考えられる．例えば，有限体上の行列でなく，実数体の行列の積計算への方式の考え方を応用することが考えられる．ビッグデータ解析や機械学習などでは，実数体の

[*1] 誤り訂正符号や，それを元にした従来の符号化分散計算方式 [1] などでは，メモリ量や通信量の制限は設けないで議論を行う．

行列積計算をすることが多いため，そのような応用先が考えられる [8]．例えば，[9] では，100×10000000 行列とその転置行列の積を求めている．そのほか，ビッグデータ解析において行列の積計算を利用する文献は多くある．どのような文献があるかは例えば [10] を参照せよ．ほかにも，計算する関数を非線形関数に変えるなどのことも考えられる．しかしながら，上記の通り，本論文は抽象化した方式の理論研究であるため，現実への応用先を一つに限定して具体化しているわけではなく，一般論のみを展開している．

次に，秘匿性の必要性について説明する．分散計算の際に，行列 $\boldsymbol{A}$ がデータベースのように何らかの個人情報を含む数値を並べた行列である場合など，計算するデータに何らかの秘密情報が含まれる場合，そのデータを秘匿しながら計算を行いたいという目標が生じる．例えば，[11] などでは，プライバシー保護を考慮した回帰分析を行っている．

(a) と (b) を考慮した分散符号化計算方式について説明する．第 3 章で論じられている (a) と (b) を考慮した従来方式（例えば [1]）の一般的手順は次の通りである．ここでは，マスターと $n(\geq k_{\mathrm{A}})$ 台のワーカーを用いる．

1. 行列 $\boldsymbol{A}$ が入力されると，マスターは前処理として，行列 $\boldsymbol{A}$ を k_{A} 個の行ベクトルに分割し，$\mathbb{F}_q$ 上 $n \times k_{\mathrm{A}}$ 行列 $\boldsymbol{G}$ を用いて冗長性を付加する変換（符号化）を行い，$n \times l$ 行列 $\boldsymbol{GA}$ に変換する．その後，$\boldsymbol{GA}$ を n 個の行ベクトルに分割し，n 台のワーカーに行ベクトルを 1 個ずつ送信する．i 番目のワーカーは $\boldsymbol{GA}$ の i 番目の行ベクトルを受信して保存する．
2. 行列 $\boldsymbol{B}$ が入力されると，マスターは全ワーカーに行列 $\boldsymbol{B}$ を送信する．ワーカー i は保存していた $\boldsymbol{GA}$ の i 番目の行ベクトルと行列 $\boldsymbol{B}$ の積計算を行い，マスターに計算結果を送信する．この分散計算とマスターへの送信において，一部のワーカーの計算結果に誤りが発生することで，マスターが $\boldsymbol{GAB} + \boldsymbol{E} \in \mathbb{F}_q^{n \times k_{\mathrm{B}}}$ を受信したとする．ここで，$\boldsymbol{E}$ は誤りを表す行列である．
3. マスターは $\boldsymbol{GAB} + \boldsymbol{E}$ を復号する．誤り行列 $\boldsymbol{E}$ がある条件を満たせば，正しく積行列 $\boldsymbol{AB}$ の値を得る．

全ワーカーでベクトルを分散させて並列計算する計算時間と単独のマスターで計算する計算時間の比が $1/n$ 程度になるため，この方式におけるシステム全体の総計算時間は短縮される．また，これにより，従来の分散符号化計算方式は，積行列 $\boldsymbol{AB}$ を列ごとに符号化し $\boldsymbol{GAB}$ を得る方式であることを明確にし，したがって，一部のワーカーの故障による誤り，消失を訂正可能な方式であることを再確認する．この方式は，デジタル情報のストレージ（あるいはデジタル情報の通信）において発生した誤り，消失を訂正する方式である誤り訂正符号の原理を利用しているため，計算途中

で生じた誤り，消失を条件によっては訂正可能である．

(a),(b),(c) をすべて同時に考慮した方式である「グループ型分散秘匿符号化計算方式」の従来研究について説明する．グループ型分散秘匿符号化計算方式に直接関係するとは限らないが，(c) を考慮した（分散計算方式に限らない）分散方式の従来研究としては，秘密分散 (Secret Sharing)[12]，PIR[13]，Robust Byzantine PIR[14][15], Private Computation[16], 分散秘匿符号化計算方式 [5] などがある．ここでは，すべての確率変数は，$\mathbb{F}_q$ の元（スカラー）に実現値をとるか，$\mathbb{F}_q$ 上行列に実現値をとるか，$\mathbb{F}_q$ 上ベクトルに実現値をとるかのいずれかである．

秘密分散は，秘密情報を複数台のストレージに分散して安全に保存するストレージ方式である．秘密分散の性能評価基準は平均保存レートである．秘密分散法は，本論文で論じる分散符号化計算方式のように，符号化から復号までにかかるシステムの総計算時間を評価基準とはしていない点で異なる．

Private Information Retrieval(以降，PIR) は，複数台のデータベース（以降，DB）に保存されている M 個のメッセージをもとに，ユーザが検索したいメッセージのインデックス $m \in [M] \coloneqq \{1, \ldots, M\}$ が何かを各 DB に秘匿しつつ情報検索を行うシステムである．PIR の評価基準は，応答値の長さに対するメッセージの長さの比（ダウンロードレート）である．PIR の派生型の研究として，Byzantine PIR，Robust PIR, Private Computation(以降，PC) などがある．この PIR は誤り，消失訂正能力を仮定しない方式であるが，一定個数以下の DB の応答値に誤り，消失があっても訂正可能な PIR をそれぞれ Byzantine PIR，Robust PIR と呼ぶ．PC は，複数台の DB に保存されている K 個メッセージと m 個の線形関数をもとに，ユーザが検索したいメッセージのインデックス $\theta \in [m]$ が何かを各 DB に秘匿しつつメッセージの θ 番目の線形関数を計算するシステムである．ただし，m 個の線形関数自体は各 DB に公開されている．評価基準は PIR と同様にダウンロードレートである．

分散計算方式の中で秘匿性を有する方式の代表例としては [5] の方式が挙げられる．しかし，これは，(b) 消失訂正能力と (c) 情報秘匿性を評価基準とする分散計算方式であって，本論文のように符号化から復号までにかかるシステムの総計算時間を評価基準とはしていない点で異なる．

1.2 本論文の目的と位置づけ

最初に，第 4 章で論じる (a) と (b) を考慮した分散符号化計算に関する本論文の内容を説明する．4.1 節では，分散符号化計算方式において誤り訂正符号を対応付けることにより，分散符号化計算方式において誤り訂正符号の原理が機能する部分を明確化する．これを明確化にすることもまた本論文の新規性の一部に含まれる．

上記をもとに，続けて 4.2 節では，ワーカーの試みる分散計算の結果が積行列 $\boldsymbol{AB}$ 自体をまとめて符号化した行列となるように各ワーカーの計算する関数を変更した方式を，本論文の方式として提案する．本方式は，$\boldsymbol{AB}$ を $\mathbb{F}_q$ の拡大体上ベクトルに変換したものから拡大体上の符号語へ変換することと同等の処理を，基礎体 $\mathbb{F}_q$ 上の計算に分解して分散計算する方式である．これにより，計算時間の短縮と，列同士が依存する誤り行列の訂正を同時に可能にする．そのため，本論文では，この方式を，グループ型分散符号化計算方式と呼ぶ．本方式では，複数のワーカーを均等にグループ分けして，各グループ内で分散計算を行う．特に，一例として，行列 $\boldsymbol{AB}$ を Gabidulin 符号化するグループ型分散符号化計算方式の構成法を提案する．本論文では，この方式をグループ型 G 方式と呼ぶ.

さらに，4.3 節では，グループ型 G 方式を，(a) 計算時間，(b) 信頼性の 2 つを評価基準として性能評価する．以下，グループ型 G 方式の性能評価の概要を述べる．(a) で論じられているシステムの総計算時間に関しては，ワーカーの並列計算，およびマスターの復号処理における $\mathbb{F}_q$ 上の四則演算回数の合計回数で評価する．その上で，グループ型 G 方式の計算時間と，マスターが単独で積行列 $\boldsymbol{AB}$ を計算する方式の計算時間を比較し，グループ型 G 方式が有利であるようなワーカーの個数のパラメータ $n_{\mathrm{A}}, n_{\mathrm{B}}$ の条件を示す．ここで，Gabidulin 符号を扱うことで拡大体 $\mathbb{F}_{q^m}$ 上の四則演算もかかわってくるが，これも基礎体 $\mathbb{F}_q$ での演算方法を指定して何回必要であるかを数えることにより，$\mathbb{F}_q$ 上の四則演算回数として計上する．(b) で論じられている訂正可能な誤り行列 $\boldsymbol{E}$ の値の条件に関しては，誤り行列 $\boldsymbol{E}$ がランク $t \coloneqq \lfloor (n - k_{\mathrm{A}})/2 \rfloor$ 以下であれば訂正可能であることを示す．従来の符号化分散計算に使われてきた誤り訂正符号がベクトルに対する符号化であるのに対して，Gabidulin 符号は，行列に対する符号化を行う符号であるために，誤りのなす行列のランクが t 以下であるときに訂正可能な符号である．これにより，従来方式では訂正できなかった行列全体の誤り，すなわち，ワーカー全体に及ぶ誤りも，上記の条件を満たせば訂正可能なことが示される．

次に，第 5 章で論じる (a)，(b)，(c) を考慮した分散符号化計算に関する本論文の内容を説明する．本論文では，この方式を，グループ型分散秘匿符号化計算方式と呼ぶ．本方式は，行列に対する $(h_{\mathrm{A}}, n_{\mathrm{A}})$ 秘密分散法（分散秘匿符号化計算方式 [5] と同様）を各グループ内で分散計算することにより，計算時間の短縮と，一部のグループの計算結果の消失に対する訂正と，グループ間での結託に対する $\boldsymbol{A}$ の値の秘匿を可能にする方式である．5.1 節で論じるグループ型分散秘匿符号化計算方式の基本方式の手順は，次の通りである．ここでは，マスターと複数台のワーカーを用いる．さらに，ワーカーを n_{A} 個のグループに分ける．各グループは n_{B} 個のワーカーで構成される．

1. 行列 $\boldsymbol{A}$ が入力されると，マスターは，前処理として，行列 $\boldsymbol{A}$ とそれとは独立な確率変数行列 $\boldsymbol{R}$ を並べた行列 $(\boldsymbol{A}^\top, \boldsymbol{R}^\top)^\top$ を符号化して行列 $\tilde{\boldsymbol{A}}$ を生成する．その後，$\tilde{\boldsymbol{A}}$ を n_{A} 個の部分行列に分割し，n_{A} 個のグループに部分行列を 1 個ずつ送信する．i 番目のグループは i 番目の部分行列を受信して，グループ内の各ワーカーはその部分行列を保存する．
2. 行列 $\boldsymbol{B}$ が入力されると，マスターは行列 $\boldsymbol{B}$ を n_{B} 個の部分行列に分割し各ワーカーに送信する．i 番目のグループの j 番目のワーカーは，$\tilde{\boldsymbol{A}}$ の i 番目の部分行列と $\boldsymbol{B}$ の j 番目の部分行列の積を計算し，その出力結果をマスターに送信する．この分散計算において，マスターは，一部のグループ，あるいは一部のワーカーの計算結果を受信しない，あるいは誤った結果を受信することを想定する．
3. マスターは，各ワーカーから受信した計算結果を集約し，復号を行う．消失が発生した箇所の個数がある条件を満たせば，積行列 $\boldsymbol{AB}$ の正しい値を得る．

上記 1 において，行列 $\boldsymbol{A}$ と確率変数行列 $\boldsymbol{R}$ を用いて生成した行列 $\tilde{\boldsymbol{A}}$ を部分行列へ分割することによって，行列 $\boldsymbol{A}$ の値が各ワーカーへ秘匿される．具体的には，グループ間で結託するワーカーの個数が一定以下の時に，行列 $\boldsymbol{A}$ の情報が一切漏洩しないことが保証される．さらには，上記 1 における符号化によって，マスターはグループ間で発生した誤り，消失を訂正できる．また，上記 2 において，行列 $\boldsymbol{B}$ が分割されることにより，各ワーカーの計算時間が少なくなる．

5.2 節においては，グループ型分散秘匿符号化計算方式の基本方式を，(a) 計算時間，(b) 信頼性，(c) 秘匿性の 3 つを評価基準として性能評価する．(a) に関しては，本論文の 1 つ目と同様に，提案方式が単独のマスターで計算する方式と比較して有利である条件を導出している．(b) に関しては，$n_{\mathrm{A}} - h_{\mathrm{A}}$ 個 $(h_{\mathrm{A}} \leq n_{\mathrm{A}})$ のグループの計算結果の消失が発生しても正しく復元できることを示している．(c) に関しては，$h_{\mathrm{A}} - 1$ 個以下のグループ結託で $\boldsymbol{A}$ の情報が全く漏洩しないことを示している．

さらに，5.3 節では，グループ内で発生した消失も訂正可能である方式，すなわち情報秘匿性や信頼性を一般化した方式（改良方式）の構成法もまた提案している．5.4 節では，改良方式の (a) 計算時間，(b) 信頼性，(c) 情報秘匿性を評価し，改良方式が優れた性質を有することを示す．

本論文と従来研究の関係は図 1.1 の通りである．

本論文は，6 章から成り立っている．第 1 章では，上記の通り，本論文の位置づけを論じる．第 2 章では，準備として，有限体，確率論，情報理論，誤り訂正符号，秘密分散について概説する．第 3 章は大きく 3 つに分かれる．3.1 節では，本論文で論じる計算の目標と，その計算のための分散計算方式の概要，さらに，本論文で論じる

<table>
<tr><td colspan="3" rowspan="2">分散計算の
着目する点</td><td colspan="2">情報秘匿性(c)</td></tr>
<tr><td>無</td><td>有</td></tr>
<tr><td rowspan="4">計算の速さ(a)</td><td rowspan="2">無</td><td rowspan="2">信頼性(b)</td><td>無</td><td rowspan="2"></td><td rowspan="2"></td></tr>
<tr><td>有</td></tr>
<tr><td rowspan="2">有</td><td rowspan="2">信頼性(b)</td><td>無</td><td>分散計算</td><td rowspan="2"></td></tr>
<tr><td>有</td><td>分散符号化計算
(3.1章：従来[1]等)
(4章：本研究(1))</td></tr>
<tr><td colspan="3"></td><td colspan="2">秘匿分散符号化計算
(3.2章：従来[5]等)</td></tr>
<tr><td colspan="3"></td><td colspan="2">グループ型秘匿分散符号化計算
(5章：本研究(2))</td></tr>
</table>

図 1.1　本論文と従来研究の関係

分散計算方式の評価基準を説明する．3.2 節では，上述の (a) 計算時間と (b) 信頼性を考慮した従来の分散符号化計算方式（例えば [1]）の一般的な手順を説明する．3.3 節では，分散計算方式に限らずに秘匿性を有する分散方式の従来研究を概観し，本論文の立ち位置を明確化する．第 4 章，第 5 章が，本論文の提案の内容の説明である．第 4 章では，(a)(b) の基準で良い性質を有する分散符号化計算方式であるグループ型分散計算方式を提案する．特に，一例として，方式全体で $\boldsymbol{AB}$ を Gabidulin 符号化する方式であるグループ型 G 方式の構成法を提案する．第 5 章では，(a)，(b)，(c) の基準で良い性質を有する分散符号化計算方式であるグループ型分散秘匿符号化計算方式を提案する．最後に，第 6 章では，本論文の結論と今後の課題を説明する．

第 2 章

準備

ノーテーションを定義する．$\mathbb{R}, \mathbb{N}, \mathbb{Z}$ を，それぞれ，実数全体の集合，正整数（0 より大きい整数) 全体の集合，整数全体の集合とおく．2 個の整数 m, n に対し，集合 $[m, n]$ を，$m \leq n$ のときは $[m, n] \coloneqq \{ m, m+1, \ldots, n \}$，$m > n$ のときは $[m, n] \coloneqq \emptyset$ と定義する．整数 n に対し，集合 $[n]$ を，$[n] \coloneqq [1, n]$ と定義する．集合 A と B の差集合を $A \setminus B$，A の冪集合を 2^A と表す．集合 $\mathcal{X}$ と $\mathcal{Y}$ に対し，直積集合を $\mathcal{X} \times \mathcal{Y}$ と表す．任意の正整数 n と集合 $\mathcal{X}_1, \ldots, \mathcal{X}_n$ に対し，これらの直積集合を $\prod_{i \in [n]} \mathcal{X}_i$ と表す．ベクトルは断りがない限り列ベクトルのみ扱う．行列，ベクトルの転置記号を $\top$ と表す．正整数 q に対し，位数 q の有限体を $\mathbb{F}_q$ と表す．本論文では暗黙裡に $q \geq 2$ の場合のみ考える．有限体 $\mathbb{F}_q$ において，加法の単位元を 0，乗法の単位元を 1 と表す．$\mathbb{F}_q$ 上の $n \times b$ 行列全体の集合を $\mathbb{F}_q^{n \times b}$ と表し，$\mathbb{F}_q^n \coloneqq \mathbb{F}_q^{n \times 1}$ と表す．さらに，行列 $\boldsymbol{E} \in \mathbb{F}_q^{n \times b}$ の第 j 列目ベクトルを $\boldsymbol{e}_{\cdot j} \in \mathbb{F}_q^n$，第 i 行目ベクトルの転置を $\boldsymbol{e}_{i \cdot} \in \mathbb{F}_q^b$ と表す．よって行列 $\boldsymbol{E}$ は $(\boldsymbol{e}_{\cdot 1}, \ldots, \boldsymbol{e}_{\cdot b}) = (\boldsymbol{e}_{1 \cdot}, \ldots, \boldsymbol{e}_{n \cdot})^\top$ と表せる．関数 f の x における値を $y = f(x)$ で表すが，関数に入力する変数 x（**説明変数**）の位置を明示するときは関数 f 自体を $f(\cdot)$ と表すことがある．$\boldsymbol{I}_l$ は $\mathbb{F}_q$ 上 l 次単位行列，$\otimes$ は行列のクロネッカー積である．すなわち，k, l, m, n を任意の正整数として，$\mathbb{F}$ を任意の（有限）体として，体 $\mathbb{F}$ 上の $k \times l$ 行列 $\boldsymbol{A}$ と $m \times n$ 行列 $\boldsymbol{B}$ に対して，

$$\boldsymbol{A} \otimes \boldsymbol{B} \coloneqq \begin{pmatrix} a_{11}\boldsymbol{B} & \ldots & a_{1l}\boldsymbol{B} \\ \vdots & \ddots & \vdots \\ a_{k1}\boldsymbol{B} & \ldots & a_{kl}\boldsymbol{B} \end{pmatrix} \in \mathbb{F}^{km \times ln}. \tag{2.1}$$

ただし，$a_{ij} \in \mathbb{F}$ は $\boldsymbol{A}$ の第 $(i, j) \in [k] \times [l]$ 成分を表す．本論文では，? はシンボルの消失を表す形式的記号，もしくは復号不能を表す形式的記号である．? と有限体 $\mathbb{F}_q$ の元は異なるものとして扱う．ここで，各 $a \in \mathbb{F}_q$ に対し，a と ? の和，差を ? であると形式的に定義しておく．変数が x である $\mathbb{F}_q$ 上多項式環を $\mathbb{F}_q[x]$ と表す．

2.1 有限体に関する基礎事項

本論文では，有限体の行列の積計算を扱うので，本節で有限体の基礎事項について説明する．

2.1.1 有限体に関する基礎事項

本小節では，有限体に関する基礎事項，特に代数的性質について説明する．

最初に，体に関する事項を説明する．これらに関しては例えば [17] を参照せよ．以降，体の定義，部分体，体の準同型，体の同型に関しては既知とする．

定義 2.1.1（体の拡大） 体 $\mathbb{F}, \mathbb{K}$ が $\mathbb{F} \subset \mathbb{K}$ を満たすとき，$\mathbb{K}$ は $\mathbb{F}$ の**拡大体**，$\mathbb{F}$ は $\mathbb{K}$ の**基礎体**と呼ぶ．$\mathbb{F} \subset \mathbb{K}$ であることを**体の拡大**という．

定義 2.1.2（有限次拡大体） 体 $\mathbb{F}$ が $\mathbb{K}$ の基礎体であるとき，$\mathbb{K}$ は $\mathbb{F}$ 上線形空間をなす．線形空間の次数が正整数 m であるとき，$\mathbb{K}$ は $\mathbb{F}$ の m **次拡大体**であるという．

定義 2.1.3（標数） 体 $\mathbb{F}$ が与えられたとする．その乗法の単位元を 1 と表す．ある正整数 p が存在し，任意の元 1 を p 回足したら 0 になる（$1 + \cdots + 1 = 0$）とする．このような正整数 p の最小値を**標数** (character) と呼び，$\mathrm{char}(\mathbb{F})$ と表す．また，そのような正整数 p が存在しない場合，標数 $\mathrm{char}(\mathbb{F})$ を 0 と定義する．

命題 2.1.1（素体） 体 $\mathbb{F}$ の標数が正である場合，その標数は素数である．また，$\mathbb{F}_q$ の部分集合 $\{ 0, 1, 1+1, 1+1+1, \dots \}$ は $\mathbb{F}$ の二項演算によって部分体を成し，それは $\mathbb{Z}/p\mathbb{Z}$ と同型な体である．

この体 $\{ 0, 1, 1+1, 1+1+1, \dots \}$ を**素体**と呼ぶ．

次に，有限体に関する事項を説明する．これ以降に関しては，例えば [18] を参照せよ．対応する訳語に関しては，例えば [17] を参照せよ．

定義 2.1.4 体の濃度（**位数**）が有限であるとき，その体を**有限体**と呼ぶ．

本論文では，体の位数は常に 2 以上であると仮定しておく．

命題 2.1.2 位数が正整数 q である有限体 $\mathbb{F}$ の標数 p は正の素数である．有限体 $\mathbb{F}$ は素体 $\mathbb{K}$ の線形空間である．特に，$\log_p q$ は素体 $\mathbb{K}$ 上線形空間 $\mathbb{F}$ の次元である．

このことから，有限体の位数 q は常に素数冪である．すなわち，ある素数 p が存在して $\log_p q$ が正整数である．

命題 2.1.3　任意の正整数 q が与えられているとする．位数 q の有限体が存在する必要十分条件は，q が素数冪である，すなわち，ある素数 p が存在して $\log_p q$ が正整数であることである．

q が素数である場合，$\mathbb{Z}/q\mathbb{Z}$ は通常の和，積により体を成す．q が素数冪である場合の $\mathbb{F}_q$ の構造については [18] を参照せよ．

また，任意の有限体は，以下の命題から同型を除きただ一つに定まることが示される．

命題 2.1.4　q を任意の素数冪とする．位数が q である有限体が存在し，さらに，2 個の任意の有限体 $\mathbb{F}$ と $\mathbb{F}'$ がともに位数 q であれば，$\mathbb{F}$ と $\mathbb{F}'$ の間には常に体同型関数が存在する．

位数が q である有限体を $\mathbb{F}_q$ と表す．例えば $\mathbb{F}_2$ は $\mathbb{Z}/2\mathbb{Z}$ と同型であるので，これを $\{0,1\}$ と表す．

例えば，位数が 2 の冪である有限体においては，$-1 = 1$ が成立するので，加算と減算が同等の演算になる．

これによって，$\mathbb{F}_p \subset \mathbb{F}_q$ とみなす．また，一般には次のことが示される．

補題 2.1.1　任意の素数冪 q, q' をとる．$\mathbb{F}_q \subset \mathbb{F}_{q'}$ である[*1]必要十分条件は，$q' = q^m$ を満たす正整数 m が存在することである．また，このとき，体 $\mathbb{F}_{q'}$ は体 $\mathbb{F}_q$ の m 次拡大体である．

特に，$\mathbb{F}_q$ 上線形独立な元 $v_1, \ldots, v_m \in \mathbb{F}_{q^m}$ が存在する．これらは基底を成す．特に，以下で定義する正規基底 (normal element)[18] を本論文では主に取り扱う．これは符号理論でも頻繁に取り扱われるクラスである [19].

定義 2.1.5（正規基底）　$\mathbb{F}_q$ 上線形独立な元 $v_1, \ldots, v_m \in \mathbb{F}_{q^m} \setminus \{0\}$ が, 任意の $i \in [m]$ に対し $v_i = v_1^{q^{i-1}}$ を満たすとき，集合 $\{v_1, \ldots, v_m\}$ を**正規基底** (normal basis) と呼ぶ．また，v_1 を**正規要素** (normal element) と呼ぶ．

本論文では，正規基底を $\{v_1, \ldots, v_m\}$ と表した時に，常に各 $i \in [m]$ に対し $v_i = v_1^{q^{i-1}}$ が成立するものとする．

命題 2.1.5（正規基底の存在定理）　任意の有限体 $\mathbb{F}_q$ と任意の正整数 m に対し，正規基底 $\{v_1, \ldots, v_m\} \subset \mathbb{F}_{q^m}$ が存在する．

次に，線形化多項式 [18] に関わる諸定義，性質を説明する．これは付録で論じる

[*1] 厳密には，これは $\mathbb{F}_q$ から $\mathbb{F}_q$ への単射な体準同型が存在することを表す．

Gabidulin 符号の復号アルゴリズムの説明で現れる概念である．

定義 2.1.6（線形化多項式）　変数が x の $\mathbb{F}_{q^m}$ 上 $F(x)$ が，$F(x) = \sum_{i\in[0,u]} f_i x^{q^i}$，$f_u \neq 0$ という形であるとき，$F(x)$ を**線形化多項式** (linearized polynomial) と呼ぶ．また，この線形化多項式の **q-次数** $\deg_q F(x)$ を u と定義する．

多項式環 $\mathbb{F}_{q^m}[x]$ における線形化多項式全体の集合 $\mathbf{L}[x]$ は，次の和 $+$，積 $*$ によって非可換環を成す [18]．特に積が多項式の通常の積と異なる．

定義 2.1.7（線形化多項式の和と積）　変数が x の $\mathbb{F}_{q^m}$ 上線形化多項式 $F(x) = \sum_{i\in[0,u]} f_i x^{q^i}$，$G(x) = \sum_{i\in[0,v]} g_i x^{q^i}$，$f_u g_v \neq 0$ が与えられているとする．線形化多項式 $F(x)$，$G(x)$ の和 $F(x)+G(x)$ を，通常の多項式の和と定義する．また，線形化多項式の積 $F(x) * G(x)$ を $F(G(x))$ と定義する．

$F(x) * G(x) = \sum_{i\in[0,u+v]} h_i x^{q^i}$ の各係数 h_i は，$\sum_{j\in[0,i]} f_i g_{i-j}^{q^j}$ である．

定義 2.1.8（線形化多項式の q-次数）　変数が x の $\mathbb{F}_{q^m}$ 上線形化多項式 $F(x) = \sum_{i\in[0,u]} f_i x^{q^i} \in \mathbb{F}_{q^m}[x]$，$f_u \neq 0$ の **q-次数** $\deg_q F(x)$ を u と定義する．

線形化多項式の根全体の集合が $\mathbb{F}_q$ 上線形空間を成すことを示す．

命題 2.1.6　標数が正である任意の体 $\mathbb{F}$ の任意の元 $a, b \in \mathbb{F}$ に対して，

$$(a+b)^{\mathrm{char}(\mathbb{F})} = a^{\mathrm{char}(\mathbb{F})} + b^{\mathrm{char}(\mathbb{F})}. \tag{2.2}$$

系 2.1.1　任意の素数冪 q と任意の正整数 l, m, n および $\mathbb{F}_{q^m}$ の n 個の任意の元 $a_1, \ldots, a_n \in \mathbb{F}_{q^m}$ に対して，

$$(a_1 + \cdots + a_n)^{q^l} = a^{q^l} + \cdots + a_n^{q^l}. \tag{2.3}$$

命題 2.1.7（[18]）　$\mathbb{F}_{q^m}$ 上線形化多項式 $\Lambda(x) = \sum_{j\in[0,t]} \lambda_j x^{q^j}$ の根全体の集合は $\mathbb{F}_q$ 上線形空間をなし，その次元 r は t 以下である．

証明　系 2.1.1 から，任意の $a, b \in \mathbb{F}_q$, $x, y \in \mathbb{F}_{q^m}$ に対し，$\Lambda(ax+by) = a\Lambda(x)+b\Lambda(y)$ が成立することが示される．よって，根全体の集合は $\mathbb{F}_q$ 上線形空間をなすことが示される．もしその次元 r が $t+1$ 以上だと，根の個数が q^{t+1} 以上になり，$\Lambda(x)$ の多項式としての次数が q^t であることに矛盾する．□

次に，$\mathbb{F}_q$ の原始元について説明する．

命題 2.1.8　どのような位数 q の有限体 $\mathbb{F}_q$ とその元 $\alpha \in \mathbb{F}_q$ に対しても，$\alpha^{q-1} = 1$ が成立する．

定義 2.1.9（$\mathbb{F}_q$ の原始元）　位数 q の有限体 $\mathbb{F}_q$ と正整数 n が与えられているとする．$\alpha \in \mathbb{F}_q \setminus \{0\}$ が $\alpha^n = 1$ かつ任意の $i \in [n-1]$ に対し $\alpha^i \neq 1$ を満たすとき，α を $\mathbb{F}_q$ における**位数 n の元** (an element of order n) と呼ぶ．特に，位数 $q-1$ の元 α を $\mathbb{F}_q$ の**原始元** (primitive element) と呼ぶ．

位数 n の元という定義は一般的に可換環にも拡張可能．

命題 2.1.9　どのような位数 q の有限体 $\mathbb{F}_q$ に対しても，$\mathbb{F}_q$ の原始元 α は存在する．

有限体 $\mathbb{F}_q$ の原始元の値は，ただ一つに定まるとは限らない．

2.1.2　$\mathbb{F}_{q^m}$ 上四則演算の計算時間

拡大体 $\mathbb{F}_{q^m}$ 上四則演算は，$\mathbb{F}_q$ 上四則演算を何回行うことで得られるかを論じる．

定義 2.1.10（乗法テーブル [20]）　$\{v_1, \ldots, v_m\}$ を $\mathbb{F}_{q^m}$ の $\mathbb{F}_q$ 上正規基底とする．各 $i \in [m]$ に対し，$T_{i1}, \ldots, T_{im} \in \mathbb{F}_q$ を $v_1 v_i = \sum_{j \in [m]} T_{ij} v_j$ が成立するような元と定める．$\boldsymbol{T} \coloneqq (T_{ij})_{(i,j) \in [m]^2} \in \mathbb{F}_q^{m \times m}$ に対し，$C(\boldsymbol{T}) \in \mathbb{Z}$ を $\boldsymbol{T}$ の非零成分の個数とする．$\boldsymbol{T}$ を**乗法テーブル** (multiplication table)，$C(\boldsymbol{T})$ を**正規基底の複雑度** (complexity of the normal basis) と呼ぶ．

$\mathbb{F}_{q^m}$ 上乗算は，正規基底 $\{v_1, \ldots, v_m\} \subset \mathbb{F}_{q^m}$ を用いることで計算時間を測ることができる．

定義 2.1.11（$\boldsymbol{v}$）　$\mathbb{F}_q$ 上線形独立な元 $v_1, \ldots, v_m \subset \mathbb{F}_{q^m}$ が各 $i \in [m]$ に対し $v_i = v_1^{q^{i-1}}$ を満たすとする．すなわち，$\{v_1, \ldots, v_m\}$ は $\mathbb{F}_{q^m}$ の正規基底を成す．このとき，ベクトル $(v_1, \ldots, v_m)$ を $\boldsymbol{v}$ と表す．

以降，本節では，ベクトル $\boldsymbol{v}$ を一つ固定して議論を進める．

任意の $a \in \mathbb{F}_{q^m}$ は，$v_1, \ldots, v_m$ の $\mathbb{F}_q$ 上線形結合 $a = a_1 v_1 + \ldots + a_m v_m$ の係数を並べたベクトル $\boldsymbol{a} \coloneqq (a_1, \ldots, a_m) \in \mathbb{F}_q^m$ と 1:1 上への対応がなされる．

定義 2.1.12（f^1）　q を任意の素数冪，m を任意の正整数とする．任意の $a \in \mathbb{F}_{q^m}$ に対し，$a_1 v_1 + \cdots + a_m v_m$ を満たすベクトル $\boldsymbol{a} \coloneqq (a_1, \ldots, a_m) \in \mathbb{F}_q^{1 \times m}$ を対応させる $\mathbb{F}_q$ 上線形同型関数を $f^1 \colon \mathbb{F}_{q^m} \to \mathbb{F}_q^{1 \times m}$ と表す．

定義 2.1.13（$\boldsymbol{f}^n$）　q を任意の素数冪，m, n を任意の正整数とする．$\mathbb{F}_q$ 上線形同型関数 $\boldsymbol{f}^n \colon \mathbb{F}_{q^m}^n \to \mathbb{F}_q^{n \times m}$ は，入力 $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$ に対し $\boldsymbol{x} = \boldsymbol{X}\boldsymbol{v}$ を満たす行列 $\boldsymbol{X} \in \mathbb{F}_q^{n \times m}$ を出力する関数である．

$f^1, \boldsymbol{f}^n$ は $v_1, \ldots, v_m$ に依存して定まる関数である．また，任意の $\mathbb{F}_{q^m}$ 上 n 次元ベクトル $\boldsymbol{x} = (x_1, \ldots, x_n)^\top \in \mathbb{F}_{q^m}^n$ に対して，次式が成立する．

$$\boldsymbol{f}^n(\boldsymbol{x}) = \begin{pmatrix} f^1(x_1) \\ \vdots \\ f^1(x_n) \end{pmatrix}. \tag{2.4}$$

命題 2.1.10　$\boldsymbol{X} \in \mathbb{F}_{q^m}^{n\times m}$ が与えられたとする．$\boldsymbol{f}^n(\boldsymbol{X}\boldsymbol{v}) = \boldsymbol{X}$ が成立する必要十分条件は $\boldsymbol{X} \in \mathbb{F}_q^{n\times m}$ である[*2]．

拡大体上ベクトル $\mathbb{F}_{q^m}^n$ の元 $\boldsymbol{a}$ の四則演算の演算回数を $\mathbb{F}_q$ 上の演算回数で数えるときは，最初から $\boldsymbol{a}$ を行列 $\boldsymbol{f}^n(\boldsymbol{a})$ に対応させて議論する．そのため，次の仮定を置く．

仮定 2.1.1　任意の正整数 m, n および素数冪 q に対して，任意の $\boldsymbol{a} \in \mathbb{F}_{q^m}^n$ を入力した下で $\boldsymbol{f}^n(\boldsymbol{a}) \in \mathbb{F}_q^{n\times m}$ を出力する計算時間は無視する．また，任意の $\boldsymbol{A} \in \mathbb{F}_q^{n\times m}$ を入力した下で $(\boldsymbol{f}^n)^{-1}(\boldsymbol{A}) \in \mathbb{F}_{q^m}^n$ を出力する計算時間は無視する．

仮定 2.1.1 は [20][21][22] においても暗黙裡に置かれている．

命題 2.1.11　任意の素数冪 q と任意の正整数 m と任意の $a \in \mathbb{F}_{q^m}$ をとる．$f^1(a)$ を $(a_1, \ldots, a_m) \in \mathbb{F}_q^{1\times m}$ と表すと，

$$f^1(a^{q^i}) = (a_{m-i+1}, \ldots, a_m, a_1, a_2, \ldots, a_{m-i}). \tag{2.5}$$

証明　次式を示せば本命題は直ちに示される．

$$f^1(a^q) = (a_{m-1}, a_1, \ldots, a_{m-1}). \tag{2.6}$$

$f^1, v_1, \ldots, v_m$ の定義から，次式が成立する．

$$a = a_1v_1 + a_2v_2 + \cdots + a_mv_m = a_1v_1^{q^0} + a_2v_1^{q^1} + \cdots + a_mv_1^{q^{m-1}}. \tag{2.7}$$

命題 2.1.9 と系 2.1.1 から，

$$\begin{aligned} a^q &= (a_1v_1^{q^0} + a_2v_1^{q^1} + \cdots + a_{m-1}v_1^{q^{m-2}} + a_mv_1^{q^{m-1}})^q && (2.8)\\ &= a_1^qv_1^{q^1} + a_2^qv_1^{q^2} + \cdots + a_{m-1}^qv_1^{q^{m-1}} + a_m^qv_1^{q^m} && (2.9)\\ &= a_mv_1 + a_1v_1^{q^1} + a_2v_1^{q^2} + \cdots + a_{m-1}v_1^{q^{m-1}} && (2.10)\\ &= a_mv_1 + a_1v_2 + a_2v_3 + \cdots + a_{m-1}v_{m-1}. && (2.11) \end{aligned}$$

よって式 (2.6) が示される．□

[*2] 後に，4 章で，$\boldsymbol{f}^n(\boldsymbol{G}\boldsymbol{A}\boldsymbol{B}'\boldsymbol{v})$ という式が現れる．ここで，$\boldsymbol{G} \in \mathbb{F}_{q^m}^{n\times k_A} \setminus \mathbb{F}_q^{n\times m}$ である．このとき，$\boldsymbol{f}^n(\boldsymbol{G}\boldsymbol{A}\boldsymbol{B}'\boldsymbol{v})$ は $\boldsymbol{G}\boldsymbol{A}\boldsymbol{B}$ に一致しない．

定義 2.1.14（巡回上変換，巡回下変換 [20]）　任意の素数冪 q，正整数 m に対して，$\boldsymbol{a} \in \mathbb{F}_q^m$ の $i \in [m]$ 回の**巡回上変換** (cyclic shift up) と**巡回下変換** (cyclic shift down) をそれぞれ以下で定義する．

$$\boldsymbol{a}^{\uparrow i} \coloneqq f(a^{q^i}) = (a_i, \ldots, a_{m-1}, a_0, a_{i-1})^\top \tag{2.12}$$

$$\boldsymbol{a}^{\downarrow i} \coloneqq \boldsymbol{a}^{\uparrow(m-i)} = (a_{m-i}, \ldots, a_{m-1}, a_0, a_{m-i-1})^\top \tag{2.13}$$

仮定 2.1.2　上記の巡回上変換，巡回下変換の計算時間は無視できるほど小さい．すなわち，$a \in \mathbb{F}_{q^m}$ における a^{q^i} の計算時間は無視できるほど小さい．

仮定 2.1.2 は，[20][21][22] においても置かれている．

命題 2.1.9 から明らかな通り，任意の $a \in \mathbb{F}_{q^m}$ に対し，$a^{q^m} = a$ が成立する．これは $\boldsymbol{a}^{\uparrow m} = \boldsymbol{a}^{\downarrow m} = \boldsymbol{a}$ を意味する．

補題 2.1.2（[22]）　a と b を $\mathbb{F}_{q^m}$ の任意の元とする．$f^1(a) = (a_1, \ldots, a_m)^\top (\in \mathbb{F}_q^m)$ を $\boldsymbol{a}$ と表し，また，$f^1(b) = (b_1, \ldots, b_m)^\top (\in \mathbb{F}_q^m)$ を $\boldsymbol{b}$ と表す．また，v_j は上記の最適な正規基底の $j \in [m]$ 番目の元であるとする．このとき，次式が成立する．

$$f^1(av_j) = (\boldsymbol{T}^\top \boldsymbol{a}^{\uparrow(j-1)})^{\downarrow(j-1)} \in \mathbb{F}_q^{1\times m}. \tag{2.14}$$

$$f^1(ab) = \sum_{i\in[m]} b_i (\boldsymbol{T}^\top \boldsymbol{a}^{\uparrow(i-1)})^{\downarrow(i-1)} \in \mathbb{F}_q^{1\times m}. \tag{2.15}$$

補題 2.1.2 は計算により示される．次のことが示される．

補題 2.1.3（[23]）　$\mathbb{F}_{q^m}$ の加算は，$\mathbb{F}_q$ 上加算 m 回行うことによって求めることができる．$\mathbb{F}_{q^m}$ の乗算は，$\mathbb{F}_q$ 上加算を $m(C(\boldsymbol{T})+1) - m^2 - 1$ 回，$\mathbb{F}_q$ 上乗算を $m(C(\boldsymbol{T})+m)$ 回行うことによって求めることができる．

補題 2.1.3 は式 (2.15) から示される．

定義 2.1.15（[24]）　任意の正規基底 $\{v_1, \ldots, v_m\}$ とその乗法テーブル $\boldsymbol{T}$ に対し，$C(\boldsymbol{T}) \geq 2m - 1$ が成立する．この等号を達成する正規基底 $\{v_1, \ldots, v_m\}$ を**最適な正規基底**と呼ぶ．

補題 2.1.4（[24]）　最適な正規基底が存在する素数冪 q と正整数 m の必要十分条件は，次の 1 もしくは 2 が成立することである．

1. $m+1$ が素数であり，q が $\mathbb{Z}/m\mathbb{Z}$ における原始元であること．
2. q は 2 の冪乗であり，かつ $\log_2 q$ と m は互いに素であり，かつ $2m+1$ は素数であり，かつ乗法群 $(\mathbb{Z}/(2m+1)\mathbb{Z})^\star$ は 2 と -1 から生成される．ただし，$(\mathbb{Z}/(2m+1)\mathbb{Z})^\star$ を $\mathbb{Z}/(2m+1)\mathbb{Z}$ から 0 の同値類を除いた集合と定義する．

以降，素数冪 q と正整数 m が補題 2.1.4 の条件を満たすとする．また，正規基底 $\{v_1, \ldots, v_m\}$ を最適な正規基底とする．これを用いることで，命題 2.1.12 が示される．

命題 2.1.12　素数冪 q と正整数 m が補題 2.1.4 の条件を満たすとき，$\mathbb{F}_{q^m}$ の加算は，$\mathbb{F}_q$ 上加算を m 回行うことによって求めることができる．$\mathbb{F}_{q^m}$ の乗算は，$\mathbb{F}_q$ 上加算を $m^2 - 1$ 回，$\mathbb{F}_q$ 上乗算を $3m^2 - 1$ 回行うことによって求めることができる．

また，乗法テーブル $\boldsymbol{T}$ の各成分は $\mathbb{F}_q$ の素体の元である [24]．これを用いると，q が 2 の冪乗であるとき，$\boldsymbol{T}$ は $\mathbb{F}_2 = \{0, 1\}$ 上の行列であることが示される．これと式 (2.15) を合わせて，命題 2.1.13 が示される．

命題 2.1.13　2 の冪 q と正整数 m が補題 2.1.4 の条件を満たすとき，$\mathbb{F}_{q^m}$ の加算は，$\mathbb{F}_q$ 上加算を m 回行うことによって計算できる．また，$\mathbb{F}_{q^m}$ の乗算は $\mathbb{F}_q$ 上加算を $m^2 - 1$ 回，$\mathbb{F}_q$ 上乗算を m^2 回行うことによって計算できる．

命題 2.1.14　2 の冪 q と正整数 m が補題 2.1.4 の条件を満たすとする．任意の $a \in \mathbb{F}_{q^m}$ に対し，$a^{-1} \in \mathbb{F}_{q^m}$ は，$\mathbb{F}_q$ 上加算を $(m^2 - 1)(2\left\lfloor \log_2(m \log_2 q - 1) \right\rfloor + 2)$ 回，$\mathbb{F}_q$ 上乗算を $m^2(2\left\lfloor \log_2(m \log_2 q - 1) \right\rfloor)$ 回行うことによって求めることができる．

証明　$a \in \mathbb{F}_{q^m}$ の逆元 $a^{-1} = a^{q^m - 2}$ は，q が 2 の冪である場合は，$\mathbb{F}_{q^m}$ 上高々 $2\left\lfloor \log_2(m \log_2 q - 1) \right\rfloor$ 回の乗算で求められる [25][26][27]．これと命題 2.1.13 から，本命題が直ちに示される．□

これと，q が 2 の冪であるときに，$\mathbb{F}_q$ 上の加算と減算は同じ操作になることから，次が示される．

命題 2.1.15　2 の冪 q と正整数 m が補題 2.1.4 の条件を満たすとする．$\mathbb{F}_{q^m}$ の加算は，$\mathbb{F}_q$ 上加算を m 回行うことによって計算できる．$\mathbb{F}_{q^m}$ の減算は，$\mathbb{F}_q$ 上加算を m 回行うことによって計算できる．また，$\mathbb{F}_{q^m}$ の乗算は $\mathbb{F}_q$ 上加算を $m^2 - 1$ 回，$\mathbb{F}_q$ 上乗算を m^2 回行うことによって計算できる．$\mathbb{F}_{q^m}$ 上の逆元を求める演算は，$\mathbb{F}_q$ 上加算を $(m^2 - 1)(2\left\lfloor \log_2(m \log_2 q - 1) \right\rfloor + 2)$ 回，$\mathbb{F}_q$ 上乗算を $m^2(2\left\lfloor \log_2(m \log_2 q - 1) \right\rfloor + 2)$ 回行うことによって計算できる．

注意 2.1.1　楕円曲線を用いたある正規基底を用いて，別のアルゴリズムを適用することにより，$\mathbb{F}_{q^m}$ 上の四則演算を高々 $Km(\log m)^4 |\log(\log m)|^3$（$K$ はある定数）で行うことができる [28]．

2.2 確率論における基礎事項

本節では，確率論の基礎事項について説明する．これらは後の情報理論，秘密分散法の説明で用いる．本節の確率論に関する事項の説明は [29] を参照せよ．

本論文では，有限集合上に値をとる離散確率変数のみ用いて議論を進める．

定義 2.2.1（可測空間） 確率空間，確率変数，確率質量関数 空でない任意の有限集合 Ω について，集合 $\mathfrak{A}(\subset 2^{\Omega})$ が，次の条件をすべて満たすとき，$\mathfrak{A}$ を **σ 加法族**と呼ぶ．

- $\Omega \in \mathfrak{A}$.
- 任意の $\mathcal{A} \in \mathfrak{A}$ に対し，$\Omega \setminus \mathcal{A} \in \mathfrak{A}$ が成立する．
- 任意の可算無限個の集合 $\mathcal{A}_1, \mathcal{A}_2, \dots \in 2^{\Omega}$ に対し $\bigcup_{i\in\mathbb{N}} \mathcal{A}_i \in \mathfrak{A}$ が成立する．

さらに，集合 Ω を**標本空間**と呼び，集合 Ω とその上の σ 加法族 $\mathcal{A}$ の組 $(\Omega, \mathfrak{A})$ を**可測空間**と呼ぶ．

例えば，$\mathcal{A} = 2^{\Omega}$（冪集合）と取れば，それは Ω の σ-加法族である．

定義 2.2.2（確率空間） 可測空間 $(\Omega, \mathfrak{A})$ が与えられているとする．$\Pr(\Omega) = 1$, $\Pr(\emptyset) = 0$ が成立し，さらに，可算無限個の集合 $\mathcal{A}_1, \mathcal{A}_2, \dots \in \mathfrak{A}$ に対し，$\Pr(\bigcup_{i\in\mathbb{N}} \mathcal{A}_i) = \sum_{i\in\mathbb{N}} \Pr(\mathcal{A}_i)$ が成立する関数 $\Pr\colon 2^{\Omega} \to \{\, r \in \mathbb{R} \mid 0 \le r \le 1 \,\}$ を**確率測度**と呼び，組 $(\Omega, \mathfrak{A}, \Pr)$ を**確率空間**と呼ぶ．

定義 2.2.3（確率変数） 確率空間 $(\Omega, \mathfrak{A}, \Pr)$ と，非空集合 $\mathcal{X}$ とその上の σ-加法族の組（可測空間）$(\mathcal{X}, \mathfrak{X})$ が与えられているとする．任意の $A \in \mathfrak{X}$ に対して $X^{-1}(A) \in \mathfrak{A}$ が成立するような関数 $X\colon \Omega \to \mathcal{X}$ を，($\mathcal{X}$-値) **可測関数**と呼ぶ．可測関数 $X\colon \Omega \to \mathcal{X}$ を，($\mathcal{X}$-値）**確率変数**と呼ぶ．また，任意の集合 $A \in \mathfrak{A}$ に対し，$\Pr(X^{-1}(A))$ を $\Pr(X \in A)$ と表す[*3]．また，任意の $x \in \mathcal{X}$ に対し，$\Pr(X^{-1}(\{x\}))$ を $\Pr(X = x)$ と表す．各 $x \in \mathcal{X}$ を確率変数 X の**実現値**と呼ぶ．関数 $\Pr(X^{-1}(\cdot))$ を確率変数 X の**確率分布**と呼び，**確率変数 X は（確率）分布 $\Pr(X^{-1}(\cdot))$ に従う**という．

以降，標本空間 Ω と実現値全体の集合 $\mathcal{X}$ は有限集合であるとする．さらに，以降，

[*3] 慣例的に，任意の正整数 n と n 個の任意の確率変数 $X_1, \dots, X_n$，および各 $x_1 \in \mathcal{X}_1, \dots, x_n \in \mathcal{X}_n$ に対して定義される命題 $\mathrm{T}(x_1, \dots, x_n)$ に対し，$\Pr(\{\, \omega \in \Omega \mid \mathrm{T}(X_1(\omega), \dots, X_n(\Omega)) \text{ is true} \,\})$ を $\Pr(\mathrm{T}(X_1, \dots, X_n))$ と表す．さらに，上記の確率変数 $X_1, \dots, X_n$ が問題設定に出てくる場合，「任意の実現値 $x_1 \in \mathcal{X}_1, \dots, x_n \in \mathcal{X}_n$ に対して $\mathrm{T}(x_1, \dots, x_n)$ が真である」ことを，「$\mathrm{T}(X_1, \dots, X_n)$ が真である」（あるいは「成立する」）と表記する．命題に限らず各アルゴリズム等も同様に実現値で書くべき個所を確率変数で記述する．

Ω の上の σ-加法族 $\mathfrak{A}$ を 2^{Ω}（冪集合）とする．$\mathcal{X}$ の上の σ-加法族 $\mathfrak{X}$ を $2^{\mathcal{X}}$（冪集合）とする．本節では，確率空間 $(\Omega, \mathfrak{A}, \Pr)$ と可測空間 $(\mathcal{X}, \mathfrak{X})$ を一つに固定して議論を進める．

定義 2.2.4（一様分布） 任意の実現値 $x \in \mathcal{X}$ に対し，$\Pr(X = x) = 1/|\mathcal{X}|$ が成立するとき，関数 $\Pr(X^{-1}(\cdot))$ を確率変数 X の**一様分布**と呼び，**確率変数 X は $\mathcal{X}$ 上一様分布に従う**という．

定義 2.2.5（確率変数ベクトル，確率変数行列） n を任意の正整数とする．n 個の確率変数 $X_1, \ldots, X_n$ から構成される全射な関数 $(X_1, \ldots, X_n)$ を，次式で定義する．

$$(X_1, \ldots, X_n) \colon \Omega \to \prod_{i \in [n]} \mathcal{X}_i \,;\, \omega \mapsto (X_1(\omega), \ldots, X_n(\omega)). \tag{2.16}$$

これは直積集合 $\prod_{i \in [n]} \mathcal{X}_i$ に値をとる確率変数となる．ただし，有限集合 $\mathcal{X}_i$ は確率変数 X_i の値域，すなわち実現値全体の集合．$(X_1, \ldots, X_n)$ がベクトル関数であることを強調したい場合は，$(X_1, \ldots, X_n)$ を**確率変数ベクトル**と呼ぶ．**確率変数行列**も同様に定義する．

定義 2.2.6（確率質量関数） 確率変数 X に対し，関数

$$p_X \colon \mathcal{X} \to \{\, r \in \mathbb{R} \mid 0 \leq r \leq 1 \,\} \tag{2.17}$$

を，任意の $x \in \mathcal{X}$ に対し，$p_X(x) = \Pr(X = x)$ となるように定義する．これを**確率質量関数**と呼ぶ．

確率変数 X が明らかなときは，関数 p_X を p と表す．

定義 2.2.7（同時確率質量関数，条件付確率質量関数） 確率変数 X_1, X_2 に対し，確率変数ベクトル (X_1, X_2) の確率質量関数 p_{X_1,X_2} を確率変数 X_1, X_2 の**同時確率質量関数**と呼ぶ．任意の実現値 $x_2 \in \mathcal{X}_2$ に対し，確率 X_2 の確率質量関数の値 $p_{X_2}(x_2)$ が正であると仮定する．関数 $p_{X_1|X_2} \colon \mathcal{X}_1 \times \mathcal{X}_2 \to \{\, r \in \mathbb{R} \mid 0 \leq r \leq 1 \,\}$ を，任意の $x_1 \in \mathcal{X}_1, x_2 \in \mathcal{X}_2$ に対し，$p_{X_1|X_2}(x_1|x_2)$ を $p_{X_1,X_2}(x_1, x_2)/p_{X_2}(x_2)$ で定義する．この関数 $p_{X_1|X_2}$ は X_1 に関する確率質量関数であり，X_2 が与えられた下での X_1 の**条件付確率質量関数**と呼ぶ．

定義 2.2.8（独立） 確率変数 X_1, X_2 に対し，任意の実現値 $x_1 \in \mathcal{X}_1, x_2 \in \mathcal{X}_2$ に対し $p_{X_1,X_2}(x_1, x_2) = p_{X_1}(x_1) p_{X_2}(x_2)$ が成立するとき，**確率変数 X_1, X_2 は (統計的に) 独立**であるという．

命題 2.2.1 X_i を Ω 上で定義された $\mathcal{X}_i$-値確率変数と定義する．有限集合 $\mathcal{Y}$ 上に値

をとる可測関数 $f\colon \prod_{i\in[n]} \mathcal{X}_i \to \mathcal{Y}$ に対し，Ω 上定義された $\mathcal{Y}$-値関数

$$f(X_1,\dots,X_n)\colon \Omega \to \mathcal{Y}\ ;\ \omega \mapsto f(X_1,\dots,X_n)(\omega) = f(X_1(\omega),\dots,X_n(\omega)) \tag{2.18}$$

もまた確率変数となる．

命題 2.2.2　X_1 を $\mathcal{X}_1$-値確率変数，X_2 を $\mathcal{X}_2$-値確率変数とする．k,l,m を正整数とする．

$\mathcal{X}_1 = \mathbb{F}_q^{k\times l}$，$\mathcal{X}_2 = \mathbb{F}_q^{l\times m}$ である場合，2 個の確率変数行列 X_1 と X_2 の積行列 X_1X_2 もまた $\mathbb{F}_q^{k\times m}$ 上に値をとる確率変数行列である．

$\mathcal{X}_1 = \mathbb{F}_q^{k\times l}$，$\mathcal{X}_2 = \mathbb{F}_q^{k\times l}$ である場合，2 個の確率変数行列 X_1 と X_2 の和行列 $X_1 + X_2$ もまた $\mathbb{F}_q^{k\times l}$ 上に値をとる確率変数行列である．

しばしば，論じたい確率変数と独立で値域上一様分布に従う確率変数行列，確率変数ベクトルをそれぞれ**乱数行列**，**乱数ベクトル**と呼ぶ．

2.3　情報理論における基礎事項

本節では，情報理論で用いる（シャノン）エントロピーの基礎事項について説明する．これらは秘密分散で用いる．本節の情報理論に関する事項の説明は [30] を参照せよ．

定義 2.3.1（エントロピー [30]）　定義 2.2.1 における確率空間上の確率変数 X に対し，以下で定義される実数値を確率変数 X の**エントロピー**と呼ぶ．

$$\mathrm{H}(X) \coloneqq -\sum_{x\in\mathcal{X}} p_X(x)\log_2 p_X(x) \tag{2.19}$$

ただし，$0\log_2 0 \coloneqq 0$．また，確率変数 X_1, X_2 に対し，確率変数ベクトル (X_1, X_2) のエントロピーを**確率変数 X_1, X_2 の同時エントロピー**，また，以下で定義される実数値をそれぞれ**確率変数 X_2 が与えられた下での確率変数 X_1 の条件付きエントロピー**，**確率変数 X_1, X_2 の相互情報量**と呼ぶ．

$$\mathrm{H}(X_1|X_2) \coloneqq -\sum_{x_1\in\mathcal{X}_1, x_2\in\mathcal{X}_2} p_{X_1,X_2}(x_1,x_2)\log_2 p_{X_1|X_2}(x_1|x_2) \tag{2.20}$$

$$\mathrm{I}(X_1;X_2) \coloneqq \mathrm{H}(X_1) - \mathrm{H}(X_1|X_2). \tag{2.21}$$

命題 2.3.1（エントロピーの非負性 [30]）　定義 2.2.1 で定義したエントロピー，同時エントロピー，条件付きエントロピー，相互情報量は常に非負である．

命題 2.3.2（[30]）　確率変数 X_1, X_2 について，

- $\mathrm{H}(X_1|X_2) = 0$ の必要十分条件は，ある（可測）関数 $f\colon \mathcal{X}_2 \to \mathcal{X}_1$ に対して $\Pr(\{\,\omega \in \Omega \mid X_1 = f(X_2(\omega))\,\}) = 1$ と表されることである (慣例では，これを $X_1 = f(X_2)$ と表す).
- $\mathrm{I}(X_1; X_2) = 0$ の必要十分条件は，X_1 と X_2 が独立であることである．

命題 2.3.3（相互情報量のチェイン則 [30]） m と n を正整数とする．確率変数 $X_1, \ldots, X_m, Y_1, \ldots, Y_n$ に対して，次式が成立する．

$$\mathrm{I}(X_1, \ldots, X_m\,;\, Y_1, \ldots, Y_n) \leq \sum_{i\in[m]} \sum_{j\in[n]} \mathrm{I}(X_i\,;\, Y_j). \tag{2.22}$$

注意 2.3.1 ある問題設定で，有限個の確率変数 $X_1, \ldots, X_n (n \in \mathbb{N})$ およびそれらの関数で表される確率変数を利用して議論を展開するときは，「$\Omega \coloneqq \prod_{i\in[n]} \mathcal{X}_i$」「$X_i$ は Ω の元から第 i 成分への射影関数である」という設定を暗黙裡に仮定している．本節以降，断りがない限りは確率空間は明示しない．

$X \in \mathcal{X}$ で，「X は $\mathcal{X}$ 上に値をとる確率変数である」ことをしばしば意味する．

本節では対数の底を 2 として議論したが，今後，$\mathbb{F}_q$ 上のベクトルや行列を用いるときは，断りがない限りは底を q とおく．

2.4 誤り訂正符号に関する基礎事項

本節では，誤り訂正符号の基礎事項について説明する．これらのうち，特に Reed Solomon 符号は後述の秘密分散法にも応用される．2.4.1 節，2.4.3 節，2.4.2 節の説明の詳細は [19] を参照せよ．2.4.4 節の説明の詳細は [31] を参照せよ．

2.4.1 誤り訂正符号の定式化

誤り訂正符号は，デジタル情報の保存において確率的に発生した誤りや消失を訂正することにより，効率性と信頼性の高い保存を行う方式である．誤り訂正符号はデジタル情報通信にも使われるが，数学的な定式化は同様であり，また，分散方式を論じる点では通信よりも保存の方がより本論文の問題設定に近いため，本論文ではストレージに焦点を当てて説明する．本小節では，有限体 $\mathbb{F}_q$ 上 k 次元ベクトルの分散ストレージ方式を論じる．ただし，q を任意の素数冪，k を任意の正整数とする．保存したい情報ベクトル $\boldsymbol{m} = (m_1, \ldots, m_k) \in \mathbb{F}_q^k$ の異なる値が主要なストレージ（マスター）に何回も入力され，その都度ワーカーは $\boldsymbol{m}$ の保存を試みる．

ここで，マスターは $\boldsymbol{m}$ の値を長期的に保存できないため，ほかのストレージ (ワーカー) を k 個用意して各ワーカー i に m_i の値を保存することを考える．しかし，各

ワーカー $i \in [k]$ は m_i の値を長期的に保存が可能であるが，故障が起こる，読み出しに失敗するなどして m_i の値が確率的に変化してしまうとする．

そこで，マスターのほかに，保存のために用意された $n(\geq k)$ 個のワーカーを利用して次のように保存を行う．ただし，q を任意の素数冪で，k を任意の正整数，n を $k \leq n$ を満たす任意の正整数とする．

〈**符号化処理**〉 マスターは $\boldsymbol{m} \in \mathbb{F}_q^k$ を入力されるたびに，単射な関数 $\phi\colon \mathbb{F}_q^k \to \mathbb{F}_q^n$ によって冗長性を持たせる[*4]変換を $\boldsymbol{m}$ に施し，n 次元ベクトル $\boldsymbol{c} = (c_1, \ldots, c_n) = \phi(\boldsymbol{m})$ を得る．マスターは各ワーカー i に c_i を保存する．

〈**復号処理**〉 マスターは各ワーカーからシンボルを取り出そうとするが，ワーカーの読み出しの失敗により，誤った数値の読み出しや，数値の消失が発生する．ワーカー i から読み出した値を $y_i \in \mathbb{F}_q \cup \{?\}$ とおき，$\boldsymbol{y} = (y_1, \ldots, y_n) \in (\mathbb{F}_q \cup \{?\})^n$ とおく．ここで，$y_i \in \mathbb{F}_q \setminus \{c_i\}$ であることは，誤った数値の読み出しに対応する．また，$y_i = ?$ であることは，数値の消失に対応する[*5]．また，$\boldsymbol{c}$ から $\boldsymbol{y}$ への変換は確率的なものとする．マスターは，$\psi\colon (\mathbb{F}_q \cup \{?\})^n \to C \cup \{?\}$ によって $\boldsymbol{y}$ から保存したベクトルの推定値 $\hat{\boldsymbol{c}} = \psi(\boldsymbol{y})$ を得る．ただし，$?$ は推定値ベクトルが得られないことを表す．また，$C \coloneqq \phi\left(\mathbb{F}_q^k\right)$．ここで，$\hat{\boldsymbol{c}} \in \phi\left(\mathbb{F}_q^k\right)$ であるとき，$\hat{\boldsymbol{c}}$ から $\hat{\boldsymbol{m}} \coloneqq \phi^{-1}(\hat{\boldsymbol{c}})$ を得る．

定義 2.4.1 上記のプロトコルにおいて，$\boldsymbol{m}$ を**情報 (ベクトル)**，$\boldsymbol{c}$ を**符号語**，$\boldsymbol{y}$ を**受信語**，$\hat{\boldsymbol{c}}$ を**推定送信語** $\hat{\boldsymbol{m}}$ を**推定値ベクトル**，単射な関数 $\phi\colon \mathbb{F}_q^k \to \mathbb{F}_q^n$ を**符号化関数**，符号語全体の集合 $C = \phi(\mathbb{F}_q^k)$ を $\mathbb{F}_q$ **上** (n, k) **符号**，k を**情報ベクトル長**，n を**符号語長**と呼ぶ．情報ベクトル $\boldsymbol{m}$ から符号語 $\boldsymbol{c}$ に変換する操作を**符号化**，受信語 $\boldsymbol{y}$ から推定値ベクトル $\hat{\boldsymbol{m}}$ に変換する操作を**復号**と呼ぶ．各**符号語シンボル** c_i が**受信語シンボル** y_i へ変化するとき，$y_i \in \mathbb{F}_q \setminus \{c_i\}$ に変換されることを**符号語シンボルに誤りが発生した**といい，$y_i = ?$ に変換されることを**符号語シンボルに消失が発生した**という．また，$\psi(\boldsymbol{y}) = ?$ であるときに**復号不能**であるという．さらに，$\psi(\boldsymbol{y}) \in C \setminus \{\phi(\boldsymbol{m})\}$ であるとき，**復号誤り**であるという．さらに，$\psi(\boldsymbol{y}) = \phi(\boldsymbol{m})$ であるとき，**正復号**であるという．復号誤りや復号不能が発生した時とき，復号関数は**正しく復号しなかった**という．

このモデルにおいて，信頼性が高く効率的な符号化と復号関数の組 (ϕ, ψ) を求めることが誤り訂正符号の目標である．誤り訂正符号では，**符号化率** $\log_q(|C|)/n$ の値の

[*4] $k \leq n$ を満たすため，シンボルの個数が k 個から n 個に増えるという意味．

[*5] $y_i = ?$ は，誤り訂正符号では，第 i シンボルが消失したことに対応する．

大きさ*6や符号化，復号の計算量を効率性の基準として定めることが多い．本論文では信頼性の高さの定義は数値としては定義されないが，どのような $\boldsymbol{y} \in (\mathbb{F}_q^n \cup \{?\})^n$ の値に対して正しく復号できるかを信頼性*7と定める．これを以下で定式化する．

定義 2.4.2（復号領域） $\mathbb{F}_q$ 上 (n,k) 符号 C と復号関数 $\psi\colon (\mathbb{F}_q \cup \{?\})^n \to C \cup \{?\}$ が与えられているとする．各 $\boldsymbol{c} \in C$ に対し，逆像

$$\psi^{-1}(\boldsymbol{c}) = \left\{ \boldsymbol{y} \in (\mathbb{F}_q^n \cup \{?\})^n \,\middle|\, \psi(\boldsymbol{y}) = \boldsymbol{c} \right\} \tag{2.23}$$

を符号語 $\boldsymbol{c}$ に対応する**復号領域**と呼ぶ*8．

本論文では，各 $\boldsymbol{c} \in C$ に対し，

$$\psi^{-1}(\boldsymbol{c}) = \left\{ \boldsymbol{c} + \boldsymbol{e} \,\middle|\, \boldsymbol{e} \in \psi^{-1}(\boldsymbol{0}) \right\} \tag{2.24}$$

が成立する場合のみを扱う．ただし，$\boldsymbol{0} \in \mathbb{F}_q^n$ は零ベクトル．このとき，信頼性の基準である「どのような $\boldsymbol{y} \in (\mathbb{F}_q^n \cup \{?\})^n$ の値に対して正しく復号できるか」という問題は，「$\mathcal{E} \coloneqq \psi^{-1}(\boldsymbol{0})$ がどのような形か」という問題に帰着される．

定義 2.4.3 $\mathcal{E} \subset (\mathbb{F}_q \cup \{?\})^n$ を **(訂正可能な) 誤りベクトル集合**と呼び，その元 $\boldsymbol{e}$ を **(訂正可能な) 誤りベクトル**と呼ぶ．

2.4.2 線形符号，限界距離復号

本論文では，主に，$\mathbb{F}_q$ 上 (n,k) 線形符号 $C \subset \mathbb{F}_q^n$ を扱い，さらに復号関数も限界距離復号関数である場合を扱う．このとき，各 $\boldsymbol{c} \in C$ に対し，$\psi^{-1}(\boldsymbol{c})$ は $\left\{ \boldsymbol{y} \in (\mathbb{F}_q \cup \{?\})^n \,\middle|\, \mathrm{d}(\boldsymbol{y}, \boldsymbol{c}) \le t \right\}$ という形をしている．ただし，d は $(\mathbb{F}_q \cup \{?\})^n$ 上の距離関数とする．

本論文で詳細に扱う線形符号について説明する．

定義 2.4.4（線形符号，生成行列） 符号化関数 ϕ が $\mathbb{F}_q$ 上線形関数であり，かつ単射な関数である時，符号 C を $\mathbb{F}_q$ 上 (n,k) **線形符号**と呼ぶ．線形符号 C は $\mathbb{F}_q^n$ の $\mathbb{F}_q$ 上 k 次元線形部分空間をなす．線形符号 C に対しては，ランク $\mathrm{rank}(\boldsymbol{G})$ が k である行列 $\boldsymbol{G} \in \mathbb{F}_q^{n \times k}$ が存在し，任意の $\boldsymbol{m} \in \mathbb{F}_q^k$ に対し，$\phi(\boldsymbol{m}) = \boldsymbol{G}\boldsymbol{m}$ が成立する [19]．この生成行列 $\boldsymbol{G}$ の各列は C の基底をなす．この行列 $\boldsymbol{G}$ を $\mathbb{F}_q$ 上 (n,k) 線形符号 C の**生成行列**という．

---

*6 後述の $\mathbb{F}_q$ 上 (n,k) 線形符号の符号化率は，$\log_q(|C|)/n = k/n$ である．

*7 通常は正しく復号しない確率である「復号誤り率」が信頼性の基準として採用されるが，本論文では採用しない．

*8 形式的に，? と有限体 $\mathbb{F}_q$ の元の和は ? であると定義する．

生成行列 $\boldsymbol{G}$ の取り方は C に応じてただ一つに決まるわけではない．$\boldsymbol{G}$ の中でも，特に，本論文の 4 章では，組織符号化が重要である．

定義 2.4.5（組織符号化） $\mathbb{F}_q$ 上 (n,k) 線形符号 C の生成行列 $\boldsymbol{G}$ のある第 $i_1, i_2, \ldots, i_k \in [k]$ 行目を集めた $k \times k$ 部分行列が k 次単位行列であるとする．そのときの符号化関数 $\phi\colon \mathbb{F}_q^k \to C\ ;\ \boldsymbol{m} \mapsto \boldsymbol{G}\boldsymbol{m}$ を，$\mathbb{F}_q$ 上 (n,k) 線形符号 C の**組織符号化**と呼ぶ．また，$\boldsymbol{G}$ を**組織符号化の生成行列**と呼ぶ．

どのような $\mathbb{F}_q$ 上 (n,k) 線形符号 C に対しても，組織符号化の生成行列 $\boldsymbol{G}$ は存在する．本論文では，簡単のため，$(i_1, i_2, \ldots, i_k) = (1, 2, \ldots, k)$ である場合を扱う[*9]．このとき，各 $\boldsymbol{m} \in \mathbb{F}_q^k$ に対し，$\phi(\boldsymbol{m}) = \boldsymbol{G}\boldsymbol{m}$ の第 $1, \ldots, k$ シンボルを集めたベクトルは $\boldsymbol{m}$ そのものになる．このことから，後述の復号の計算時間評価においては，符号語 $\phi(\boldsymbol{m}) = \boldsymbol{G}\boldsymbol{m}$ から情報ベクトル $\boldsymbol{m}$ を取り出す時間を無視することにする．

線形符号に対しては常にパリティ検査行列が存在するが，一意に定まるとは限らない．

定義 2.4.6（パリティ検査行列） $\mathbb{F}_q$ 上 (n,k) 線形符号とその生成行列 $\boldsymbol{G} \in \mathbb{F}_q^{n\times k}$ が与えられているとする．行列 $\boldsymbol{H} \in \mathbb{F}_q^{n\times m}$，$m \geq n-k$ が $\boldsymbol{G}^\top \boldsymbol{H} = \boldsymbol{0}_{k\times m}$ および $\mathrm{rank}\boldsymbol{H} = n-k$ を満たすとき，$\boldsymbol{H}$ をこの符号の**パリティ検査行列**と呼ぶ．

次に，復号について説明する．復号の考え方には 2 種類存在する．符号語から受信語への確率的な変換（遷移確率）を考慮しない代数的な復号と，それを考慮する復号である．本論文では前者のみ扱う．これらは符号の代数的性質に依存する．

$\mathbb{F}_q^n$ 上の距離関数 d に関して，符号の最小距離，限界距離を定義する．ただし，n は任意の正整数，$\mathbb{F}_q$ は任意の有限体とする．

定義 2.4.7（距離関数） $(\mathbb{F}_q \cup \{?\})^n$ 上の実関数 $\mathrm{d}\colon (\mathbb{F}_q \cup \{?\})^n \times (\mathbb{F}_q \cup \{?\})^n \to \mathbb{R}$ が非負性，非退化性，対称性，三角不等式を満たすとき，d を $(\mathbb{F}_q \cup \{?\})^n$ 上の距離関数と呼ぶ．

定義 2.4.8（最小距離） $\mathbb{F}_q$ 上 (n,k) 符号 C の距離関数 d に関する**最小距離** d を以下の式で定義する．

$$d \coloneqq \min_{\boldsymbol{c},\boldsymbol{c}\in C\colon \boldsymbol{c}\neq\boldsymbol{c}'} \mathrm{d}(\boldsymbol{c},\boldsymbol{c}') \tag{2.25}$$

$(\mathbb{F}_q \cup \{?\})^n$ 上の距離関数の一つに，ハミング消失距離がある．

[*9] どのような線形符号に対しても，このように $i_1, i_2, \ldots, i_k$ の値をとることができるとは限らない．このような組織符号化を標準組織符号化と呼ぶ．本論文では，$\mathbb{F}_q$ 上 (n,k) Reed Solomon 符号と $\mathbb{F}_{q^m}$ 上 (n,k) Gabidulin 符号しか扱わないが，どちらの符号も，$(i_1, i_2, \ldots, i_k) = (1, 2, \ldots, k)$ であるような標準組織符号化の生成行列をとることができる．

定義 2.4.9（ハミング消失距離）　$\mathbb{F}_q \cup \{?\}$ 上 n 次元ベクトル $\boldsymbol{s}_1 = (s_{11}, \ldots, s_{1n})$, $\boldsymbol{s}_2 = (s_{21}, \ldots, s_{2n})$ の**ハミング消失距離** $\mathrm{d_H}(\boldsymbol{s}_1, \boldsymbol{s}_2)$ を次式で定義する．

$$\left| \left\{ i \in [n] \;\middle|\; s_{1i}, s_{2i} \in \mathbb{F}_q, s_{1i} \neq s_{2i} \right\} \right| + \frac{1}{2} \left| \left\{ i \in [n] \mid s_{1i} \text{ or } s_{2i} = ?, s_{1i} \neq s_{2i} \right\} \right| \tag{2.26}$$

$\mathbb{F}_q^n$ 上のハミング消失距離は**ハミング距離**と呼ぶ．

定義 2.4.10（最小ハミング距離）　本論文においては，$\mathbb{F}_q^n$ 上の距離関数がハミング距離であるとき，符号 C の最小距離を符号 C の最小ハミング距離と呼ぶ．

ここで，誤りや消失が発生する状況における誤り訂正能力を論じる．最初に，限界距離復号を定義する．これは，符号語に誤りが加わっても，そのベクトルと元の符号語の距離が限界距離以下であれば正しく復号する関数である．

定義 2.4.11（限界距離復号）　$\mathbb{F}_q$ 上 (n, k) 符号 C の距離関数 d に関する最小距離を d とおく．また，$2t < d$ を満たす非負実数を任意に選ぶ．**限界距離復号** $\psi \colon (\mathbb{F}_q \cup \{?\})^n \to C \cup \{?\}$ とは，入力 $\boldsymbol{y} \in (\mathbb{F}_q \cup \{?\})^n$ に対し，$\mathrm{d}(\boldsymbol{y}, \boldsymbol{c}) \leq t$ を満たす $\boldsymbol{c} \in C$ が存在すれば $\boldsymbol{c}$ を出力し，そうでなければ ? を出力する関数である．t を距離関数 d に関する**限界距離**と呼ぶ．

本論文の分散符号化計算方式における復号処理では，限界距離復号およびその組み合わせのみを扱う．

限界距離復号 ψ は well defined な関数である．すなわち，$\mathrm{d}(\boldsymbol{y}, \boldsymbol{c}) \leq t$ を満たす $\boldsymbol{c} \in C$ は，存在すれば一意に定まる [19].

定義 2.4.12（限界ハミング距離復号）　$\mathbb{F}_q$ 上 (n, k) 符号 C のハミング消失距離関数 d に関する最小距離（**最小ハミング距離**）を d とおく．**限界ハミング消失距離復号** $\psi \colon (\mathbb{F}_q \cup \{?\})^n \to C \cup \{?\}$ とは，入力 $\boldsymbol{y} \in (\mathbb{F}_q \cup \{?\})^n$ に対し，$\mathrm{d}(\boldsymbol{y}, \boldsymbol{c}) \leq t$ を満たす $\boldsymbol{c} \in C$ が存在すれば $\boldsymbol{c}$ を出力し，そうでなければ ? を出力する関数である．ただし，t は $\lfloor (d-1)/2 \rfloor$ と定義する．t を距離関数 d に関する**限界ハミング消失距離**と呼ぶ．

特に，誤りのみが発生し，消失が発生しない場合は，**限界ハミング距離復号**，**限界ハミング距離**と呼ぶ．すなわち，**限界ハミング距離復号** $\psi \colon (\mathbb{F}_q)^n \to C \cup \{?\}$ とは，入力 $\boldsymbol{y} \in \mathbb{F}_q^n$ に対し，$\mathrm{d}(\boldsymbol{y}, \boldsymbol{c}) \leq t$ を満たす $\boldsymbol{c} \in C$ が存在すれば $\boldsymbol{c}$ を出力し，そうでなければ ? を出力する関数である．t を距離関数 d に関する**限界ハミング距離**と呼ぶ．

以下の命題は，符号の訂正能力を論じている．

命題 2.4.1（[19]）　距離関数 d に関する最小距離が d である $\mathbb{F}_q$ 上 (n, k) 線形符号 C は，符号語に誤りや消失が発生しても，その結果と元の符号語の距離が限界距離 t 以

下ならば, 限界距離復号を用いて正しく訂正（正復号）可能．すなわち, $\mathrm{d}(\boldsymbol{y}, \boldsymbol{c}) \leq t$ を満たす任意の符号語 $\boldsymbol{c} \in C$ および任意のベクトル $\boldsymbol{y} \in (\mathbb{F}_q \cup \{?\})^n$ に対し，$\psi(\boldsymbol{y}) = \boldsymbol{c}$ が成立する．逆に，$\mathrm{d}(\boldsymbol{y}, \boldsymbol{c}) > t$ である場合，限界距離復号は復号不能（$\psi(\boldsymbol{y}) =?$）もしくは復号誤り（$\psi(\boldsymbol{y}) \neq \boldsymbol{c}$）を引き起こす．

距離関数がハミング (消失) 距離である場合，限界距離復号を限界ハミング (消失) 距離復号と呼ぶ．普通の符号では，その限界ハミング消失距離復号は正復号，復号誤り，復号不能のいずれかを引き起こす[*10]

上記の議論から，符号としては，n, k, q 固定のもと，最小距離 d が最大の符号を取ることが望ましい．ハミング (消失) 距離に関しては，以下の命題が成立する．

命題 2.4.2（ハミング距離の Singleton 限界） 任意の $\mathbb{F}_q$ 上 (n, k) 線形符号 $C \subset \mathbb{F}_q^n$ の最小ハミング距離は $n - k + 1$ 以下である．

上記の限界を達成する（最小ハミング距離が $n - k + 1$ である）符号を $\mathbb{F}_q$ 上 (n, k) 線形 **MDS 符号**（Maximum Distance Separated 符号）と呼ぶ．

特に，誤りが発生せず，消失のみが発生する状況での復号を論じる．

命題 2.4.3 $\mathbb{F}_q$ 上 (n, k) 符号 $C \subset \mathbb{F}_q^n$ およびそれに対する限界ハミング消失距離復号 ψ が与えられているとする．最小ハミング距離が d である $\mathbb{F}_q$ 上 (n, k) 符号の符号語 $\boldsymbol{c} = (c_1, \ldots, c_n)$ において，高々 $d - 1$ 個のシンボルの値が消失し，$n - d + 1$ 個のシンボルの値が正しく残ったとする．このとき，限界ハミング消失距離復号により $\boldsymbol{c}$ の値を正しく復元可能である．

例えば，この符号が MDS 符号である場合，$d - 1 = n - k, n - d + 1 = k$ である．

また，消失が発生せず，誤りのみが発生する状況での復号を論じる．

命題 2.4.4 $\mathbb{F}_q$ 上 (n, k) 符号 $C \subset \mathbb{F}_q^n$ およびそれに対する限界ハミング距離復号 ψ が与えられているとする．最小ハミング距離が d である $\mathbb{F}_q$ 上 (n, k) 符号の符号語 $\boldsymbol{c} = (c_1, \ldots, c_n)$ において，高々 $t = \lfloor (d-1)/2 \rfloor$ 個のシンボルの値が誤り，残り $n - t$ 個のシンボルの値が正しく残ったとする．このとき，限界ハミング距離復号により $\boldsymbol{c}$ の値を正しく復元可能である．

[*10] 限界ハミング距離復号が復号不能を引き起こさない（正復号か復号誤りのいずれか）場合，その符号は完全符号 [19] と一致する．

2.4.3 Reed Solomon 符号

正整数 k, n および素数冪 q が $k \leq n < q$ を満たすときの MDS 符号の構成法の一つとして，Reed Solomon 符号があげられる．

定義 2.4.13（Reed Solomon 符号 [32]） $q > n \geq k$ を満たす任意の正整数 n, k，任意の素数冪 q が与えられているとする．$\mathbb{F}_q$ の原始元 $\alpha \in \mathbb{F}_q \setminus \{0\}$ の値を一つ任意に固定する．式 (2.27) で定義される $\mathbb{F}_q$ 上 $n \times k$ 行列行列を生成行列に持つ有限体 $\mathbb{F}_q$ 上 (n, k) 線形符号 $C_{\mathrm{RS}} (\subset \mathbb{F}_q^n)$ を $\mathbb{F}_q$ 上 (n, k)**Reed Solomon 符号**と呼ぶ．

$$\mathbf{RS}_{n\times k} \coloneqq \begin{pmatrix} 1 & \alpha^1 & \alpha^2 & \dots & \alpha^{k-1} \\ 1 & \alpha^2 & \alpha^4 & \dots & \alpha^{2(k-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^n & \alpha^{2n} & \dots & \alpha^{n(k-1)} \end{pmatrix} \tag{2.27}$$

Reed Solomon 符号の符号語は，多項式によって表現できる．

命題 2.4.5（Reed Solomon 符号の多項式表現 [32]） 定義 2.4.13 で定義される $\mathbb{F}_q$ 上 (n, k) Reed Solomon 符号 $C_{\mathrm{RS}} \subset \mathbb{F}_q^n$ と生成行列 $\mathbf{RS}_{n\times k}$ が与えられているとする．各情報ベクトル $\boldsymbol{m} = (m_1, \dots, m_k) \in \mathbb{F}_q^k$ に対し，変数を x とする $\mathbb{F}_q$ 上多項式 $f(x) \coloneqq m_1 + m_2 x + \cdots + m_k x^{k-1}$（**情報多項式**）に対し，次式が成立する．

$$\mathbf{RS}_{n\times k} \boldsymbol{m} = (f(\alpha^1), \dots, f(\alpha^n)). \tag{2.28}$$

したがって，次式が成立する．

$$C_{\mathrm{RS}} = \left\{ (f(\alpha^1), \dots, f(\alpha^n)) \,\middle|\, f(x) \in \mathbb{F}_q[x], \deg f(x) \leq k-1 \right\}. \tag{2.29}$$

ただし，$\mathbb{F}_q[x]$ は $\mathbb{F}_q$ 上多項式全体の集合を表す．また，$\deg f(x)$ は多項式 $f(x)$ の次数を表す．

命題 2.4.6（[32]） $\mathbb{F}_q$ 上 (n, k)Reed Solomon 符号 C_{RS} は MDS 符号である．

符号 C_{RS} は $\mathbb{F}_q$ 上 (n, k) 線形 MDS 符号であるため，$\mathbb{F}_q$ 上 (n, k) 線形符号の中でその限界ハミング (消失) 距離復号の誤り訂正能力 t が最も高い．

特に，誤りが発生せず，消失のみが発生する状況での RS 符号の復号（消失復号）を論じる．命題 2.4.3 から，以下のことが示される．$\mathbb{F}_q$ 上 (n, k) Reed Solomon 符号においては，情報ベクトル $\boldsymbol{m}$ を符号化して得られた各符号語 $\boldsymbol{c} = (c_1, \dots, c_n)$ から，

高々 $n-k$ シンボルが消失してベクトル $\boldsymbol{y}=(y_1,\ldots,y_n)\in(\mathbb{F}_q\cup\{?\})^n$ が得られたとする．このとき，限界ハミング消失距離復号により，ベクトル $\boldsymbol{y}$ から，正しく符号語 $\boldsymbol{c}$（および $\boldsymbol{m}$）を復元できる．$\boldsymbol{y}$ から $\boldsymbol{m}$ を復元するその具体的な復号アルゴリズムとして，ラグランジュ補間多項式を利用する方法が存在する．

命題 2.4.7（ラグランジュ補間多項式） k を正整数とする．体 $\mathbb{F}$ を位数が k 以上である任意の体とする．x 座標が互いに相異なる k 個の任意の点 $(x_1,y_1),\ldots,(x_k,y_k)\in\mathbb{F}\times\mathbb{F}$ に対し，$y_1=f(x_1),\ldots,y_k=f(x_k)$ が成立する次数 $k-1$ 以下の $\mathbb{F}$ 上多項式 $f(x)$ は存在し，一意に定まる．それは次式で与えられる．

$$f(x)=\sum_{h=1}^{k} y_h \frac{\prod_{h'\in[k]\setminus\{h\}}(x-x_{h'})}{\prod_{h'\in[k]\setminus\{h\}}(x_h-x_{h'})} \tag{2.30}$$

ラグランジュの補間多項式については [18] を参照せよ．

ラグランジュ補間多項式を利用する復号法は以下の手順で行われる．α は $\mathbb{F}_q$ の原始元であり，$q-1\geq n$ が成立するため，$\alpha^1,\ldots,\alpha^n$ は互いにすべて異なっていることに注意する．$\boldsymbol{y}$ が与えられたとき，$\boldsymbol{y}$ に残っている k 個の符号語シンボルを $f(\alpha^{i_1}),\ldots,f(\alpha^{i_k})$ とおくと，$(\alpha^{i_1},f(\alpha^{i_1})),\ldots,(\alpha^{i_k},f(\alpha^{i_k}))$ から，ラグランジュ補間多項式により，情報多項式 $f(x)=m_1+m_2x+\cdots+\cdots+m_kx^{k-1}$ の各係数を求めることで，$\boldsymbol{m}$ を復元できる．

より具体的なアルゴリズムとその計算時間については，付録 A.1 を参照．Reed Solomon 符号の復号アルゴリズムには，離散フーリエ変換を一般化したアルゴリズムを用いるものも従来提案されている [33] が，本論文では計算時間の評価のしやすさの観点から [34] のアルゴリズムを用いる．ただし，このアルゴリズムにおいては，次の仮定を置いている．

仮定 2.4.1 q は素数であり，ある正整数 b に対して，$q=2^{2^b}+1$ が成立する．また，$n=q-1$.

このとき，この復号アルゴリズムにより，次の命題が成立する．

命題 2.4.8 体の位数 q と符号長 n が仮定 2.4.1 を満たしているとする．このとき，[34] で述べられている $\mathbb{F}_q$ 上 (n,k) Reed Solomon 符号の復号アルゴリズムの $\mathbb{F}_q$ 上四則演算回数の上界 T_{RS} は以下で与えられる．

$$\begin{aligned}T_{\mathrm{RS}}=&(9(\lceil\log_2 k\rceil)^2+9\lceil\log_2 k\rceil+19)2^{\lceil\log_2 k\rceil}-\frac{27}{2}\lceil\log_2 k\rceil\\&+4k+21n\log_2 n+\frac{27}{2}n+26.\end{aligned} \tag{2.31}$$

基本的な導出方針は [34] と同じだが，本論文ではオーダー表記ではなく回数を直接表記する．演算回数の導出は付録 A.1 を参照せよ．

2.4.4 Gabidulin 符号

本小節では Reed Solomon 符号と類似した性質を有する Gabidulin 符号 [31] を説明する．Gabidulin 符号は後述の分散符号化計算方式の提案で必要となる．本節で論じる訂正能力は**誤り訂正能力のみであり，消失訂正能力については論じない**ことに注意する．k, m, n を $m \geq n \geq k$ を満たす任意の正整数，q を任意の素数冪とする．以降，$\boldsymbol{v}$ を定義 2.1.11 に従って定義されるベクトル，f^1 を定義 2.1.12 に従って定義される関数，$\boldsymbol{f}^n$ を定義 2.1.13 に従って定義される関数とする．本節ではこれらの値を任意に一つ固定する．

Gabidulin 符号は検査行列によって定義される．

定義 2.4.14（Gabidulin 符号 [31]） 任意の素数冪 q と，$m \geq n \geq k$ を満たす任意の正整数 m, n, k をとる．$h_1, \ldots, h_n \in \mathbb{F}_{q^m}$ を $\mathbb{F}_q$ 上線形独立な元とする．行列 $\boldsymbol{H} \in \mathbb{F}_{q^m}^{n \times (n-k)}$ を第 $(i, j) \in [n] \times [k]$ 成分が $h_i^{q^{j-1}}$ であるような行列とおく．パリティ検査行列が行列 $\boldsymbol{H}$ である $\mathbb{F}_{q^m}$ 上 (n, k) 線形符号 $\tilde{C}_{\mathrm{G}} \subset \mathbb{F}_{q^m}^n$ を $\mathbb{F}_{q^m}$ 上 (n, k) Gabidulin 符号と呼ぶ．

例えば，$\mathbb{F}_{q^m}$ の $\mathbb{F}_q$ 上正規基底 $\{h, h_{q^1}, \ldots, h_{q^{m-1}}\} \subset \mathbb{F}_{q^m}$ に対し，$h_1 = h^{q^{1-1}}$，$\ldots$，$h_1 = h^{q^{m-1}}$ は $\mathbb{F}_q$ 上線形独立である．今回は，各 h_i を上記のようにとる．

この符号の生成行列は以下で与えられる．

命題 2.4.9 定義 2.4.14 で定義される $\mathbb{F}_{q^m}$ 上 (n, k) Gabidulin 符号の生成行列の一つは，$\mathbb{F}_q$ 上線形独立なある元 $g_1, \ldots, g_n \in \mathbb{F}_{q^m}$ を用いて，第 $(i, j) \in [n] \times [k]$ 成分が $g_i^{q^{j-1}}$ であるような行列 $\boldsymbol{G} \in \mathbb{F}_{q^m}^{n \times k}$ という形で表される．

ここで，符号 $\tilde{C}_{\mathrm{G}}$ と $\mathbb{F}_q$ 上線形同型である $\mathbb{F}_q$ 上線形符号 $C_{\mathrm{G}} \subset \mathbb{F}_q^{n \times m}$ を，$C_{\mathrm{G}} \coloneqq \boldsymbol{f}^n(\tilde{C}_{\mathrm{G}})$ と定義する．この符号のランク距離上の限界距離復号について，以下で論じる．

定義 2.4.15（ランク距離 [31]） 2 個の行列 $\boldsymbol{S}_1, \boldsymbol{S}_2 \in \mathbb{F}_q^{n \times m}$ のランク距離を $\mathrm{rank}(\boldsymbol{S}_1 - \boldsymbol{S}_2)$ と定義する．

$\mathbb{F}_q$ 上の線形符号 $C \subset \mathbb{F}_q^{n \times m}$ の最小ランク距離，限界ランク距離，限界距離復号関数は，ハミング距離の時と同様に定められる．

命題 2.4.10（ランク距離の Singleton 限界 [31]） 任意の $\mathbb{F}_{q^m}$ 上 (n, k) 線形符号 $\tilde{C} \subset$

$\mathbb{F}_{q^m}^n$ に対する $\mathbb{F}_q$ 上線形符号 $C \coloneqq \boldsymbol{f}^n(\tilde{C})$ の最小ランク距離は，$n-k+1$ 以下である．

上記の限界を達成する符号を $\mathbb{F}_{q^m}$ 上 (n,k) 線形 **MRD 符号**と呼ぶ．符号 $\tilde{C}_{\mathrm{G}}$ はランク距離の Singleton 限界を達成する [31]．よって，符号 C_{G} は，$\tilde{C} \subset \mathbb{F}_{q^m}^n$ が $\mathbb{F}_{q^m}$ 上 (n,k) 線形符号である符号 $C = \boldsymbol{f}^n(\tilde{C})$ の中で，その限界ランク距離復号の誤り訂正能力が最も高い．後に使うので，符号 C_{G} の限界ランク距離復号関数を改めて説明する．

定義 2.4.16（Gabidulin 符号の限界ランク距離復号関数）　限界ランク距離復号関数 $\psi\colon \mathbb{F}_{q^m}^n \to C_{\mathrm{G}} \cup \{?\}$ は，入力 $\boldsymbol{Y} \in \mathbb{F}_q^{n\times m}$ に対し，$\mathrm{rank}(\boldsymbol{Y}-\boldsymbol{C}) \leq \coloneqq \lfloor (n-k)/2 \rfloor$ を満たす $\boldsymbol{C} \in C_{\mathrm{G}}$ が存在すれば $\boldsymbol{C}$ を出力し，そうでなければ復号不能を表す記号として記号 ? を出力する関数である．ただし，$t \coloneqq \lfloor (n-k)/2 \rfloor$ とする．

この関数は well-defined である．よって，符号 C_{G} の限界ランク距離復号関数 ψ によって，ランク t 以下の誤り行列 $\boldsymbol{E}$ が訂正可能である．

符号 C_{G} の限界距離復号の復号アルゴリズム [31] の計算時間を論じる．

命題 2.4.11（限界ランク距離復号の演算回数）　素数冪 q，正整数 $m \geq n \geq k$ が与えられているとする．[21] で述べられている $\mathbb{F}_{q^m}$ 上 (n,k) Gabidulin 符号の $\mathbb{F}_q$ 上四則演算回数の上界は，以下で与えられる．

$\mathbb{F}_q$ 上の加算，乗算，逆元を求める演算の合計回数が $2mnt+m^2+m-1+\frac{2}{3}(m(m-1)(2m-1)-t(t-1)(2t-1))-(t-2)(m-t)-(t-1)(m^2-m-t^2+t)$ 回，$\mathbb{F}_{q^m}$ 上の加算が $dn+d^2-t^2+2dt+mt-n-4d-t-1$ 回，$\mathbb{F}_{q^m}$ 上の乗算が $dn+3d^2+3t^2-4dt+mt-n-9d+9t+5$ 回，$\mathbb{F}_{q^m}$ 上の乗法の逆元を求める演算が $2t$ 回である．ただし，$t \coloneqq \lfloor (n-k)/2 \rfloor$，$d \coloneqq n-k+1$ と定義する．

これは [21] の考え方に基づく値である．上記の通り，本論文では [21] で算出されている演算回数を一部修正したものを使用する．演算回数の算出の仕方は付録 A.2 を参照せよ．

拡大体 $\mathbb{F}_{q^m}$ の四則演算が $\mathbb{F}_q$ 上四則演算を何回行うことで計算できるかに応じて，Gabidulin 符号の復号アルゴリズムの $\mathbb{F}_q$ 上四則演算回数の上界は変化する．ここでは，q,m の値を補題 2.1.4 の条件を満たす値にして，拡大体の演算方法を定めてから，Gabidulin 符号の $\mathbb{F}_q$ 上四則演算回数の上界を与える．

系 2.4.1　有限体の位数 q と，拡大次数 m が補題 2.1.4 の条件を満たしているとする．このとき，[21] で述べられている $\mathbb{F}_{q^m}$ 上 (n,k_{A})Gabidulin 符号の復号アルゴリズム

の $\mathbb{F}_q$ 上四則演算回数の上界 D は以下で与えられる．

$$\begin{aligned} D &= 2mnt + m^2 + m - 1 + \frac{2}{3}(m(m-1)(2m-1) - t(t-1)(2t-1)) \\ &\quad - (t-2)(m-t) - (t-1)(m^2 - m - t^2 + t) \\ &\quad + (dn + d^2 - t^2 + 2dt + mt - n - 4d - t - 1)m \\ &\quad + (dn + 3d^2 + 3t^2 - 4dt + mt - n - 9d + 9t + 5)(2m^2 - 1) \\ &\quad + 2t(2m^2 - 1)(2\lfloor \log_2(m \log_2 q - 1) \rfloor). \end{aligned} \tag{2.32}$$

これは，命題 2.1.15 と命題 2.4.11 から示される．

次に，Erasure 復号について説明する．

定義 2.4.17（Erasure 復号 [22]） 誤り行列 $\boldsymbol{E} \in \mathbb{F}_q^{n\times m}$ に対し，ある行列 $\boldsymbol{E}_{\mathrm{R}} \in \mathbb{F}_q^{n\times \mathrm{rank}\boldsymbol{E}}$ と $\boldsymbol{E}_{\mathrm{C}} \in \mathbb{F}_q^{\mathrm{rank}\boldsymbol{E}\times m}$ が存在し，$\boldsymbol{E} = \boldsymbol{E}_{\mathrm{R}}\boldsymbol{E}_{\mathrm{C}}$ が成立し，さらに $\mathrm{rank}\boldsymbol{E} = \mathrm{rank}\boldsymbol{E}_{\mathrm{R}} = \mathrm{rank}\boldsymbol{E}_{\mathrm{C}}$ が成立するとする．$\boldsymbol{E}_{\mathrm{C}}$ を **column error** と呼び，$\boldsymbol{E}_{\mathrm{R}}$ を **row error** と呼ぶ．$\mathrm{rank}\boldsymbol{E}$ と row error $\boldsymbol{E}_{\mathrm{R}}$ がマスターに既知の下で，column error $\boldsymbol{E}_{\mathrm{C}}$ を求める問題を **Erasure 復号**と呼ぶ．

column error が既知，row error が未知，としても同様に Erasure 復号と呼ぶ．

各 $\boldsymbol{E}$ に対し，上記を満たす $\boldsymbol{E}_{\mathrm{C}}, \boldsymbol{E}_{\mathrm{R}}$ の取り方は唯一とは限らない．row error のみがマスターに既知ということは，計算結果の各行に加わっている誤りが，どの行の誤りの線形結合から構成されるかがわかっているということを指す．row error の値が既知であることにより，後述の通り，復号の計算時間の短縮が期待される．

命題 2.4.12（Erasure 復号の演算回数 [22]） $\mathbb{F}_{q^m}$ 上 (n,k)Gabidulin 符号の Erasure 復号アルゴリズムの計算回数は，$\mathbb{F}_q$ 上の加算が $\mathbb{F}_q$ 上 $(n-1)(n-k) + 2mnt - mn - nt$ 回，$\mathbb{F}_q$ 上の乗算が $\mathbb{F}_q$ 上 $n(n-k) + 2mnt$ 回，$\mathbb{F}_{q^m}$ 上の加算が $\mathbb{F}_{q^m}$ 上 $\frac{3}{2}t(t-1)$ 回，$\mathbb{F}_{q^m}$ 上の乗算が $\mathbb{F}_{q^m}$ 上 $\frac{3}{2}t^2 + \frac{1}{2}t - 1$ 回，$\mathbb{F}_{q^m}$ 上の乗法の逆元を求める演算が $\mathbb{F}_{q^m}$ 上 t 回である．ただし，$t := \lfloor (n-k)/2 \rfloor$ である．

系 2.4.2 有限体の位数 q と，拡大次数 m が補題 2.1.4 の条件を満たしているとする．このとき，$\mathbb{F}_{q^m}$ 上 (n,k)Gabidulin 符号の Erasure 復号アルゴリズムは，$\mathbb{F}_q$ 上の四則演算を $(2n-1)(n-k) + 4mnt - mn - nt + \frac{3}{2}mt(t-1) + (2m^2-1)(2mt\log_2 q + \frac{3}{2}t^2 - \frac{5}{2}t - 1)$ 回行うことで計算できる．

2.5 秘密分散における基礎事項

秘密分散は，秘密情報 $A \in \mathbb{F}_q$（一様分布に従う確率変数）を $n(\geq 2)$ 台の記憶装置に A の情報を秘匿しつつ分散して保存するストレージ方式である．後述の (k,n) 閾値秘密分散法，一般的なアクセス構造を実現する秘密分散方式は，分散秘匿符号化計算方式の構成法の提案において用いるので，それらについて説明する．最初に，秘密分散方式の手順を説明し，その後，(k,n) 閾値秘密分散法，一般的なアクセス構造を実現する秘密分散方式の具体的な構成をそれぞれ説明する．

最初に，素数冪 q，2 以上の正整数 t, N, n，組 $\Gamma = (\mathfrak{P}_{\mathrm{fo}}, \mathfrak{P}_{\mathrm{qu}}) \in 2^{[n]} \times 2^{[n]}$ が与えられた下での**秘密分散方式** $((c_i)_{i\in[n]}, \boldsymbol{G}, t-1, \psi)$ を説明する．アクセス構造 Γ は，$\mathfrak{P}_{\mathrm{fo}} \subset [n]$ は，結託して秘密情報 $A \in \mathbb{F}_q$ を読み取ろうとするワーカー集合全体の集合であり，$\mathfrak{P}_{\mathrm{fo}} \subset [n]$ は，マスターが秘密情報 A を復元するためにアクセスするワーカー集合全体の集合である．マスターには行列 $\boldsymbol{G}$ と復号関数が保存され，各ワーカーには行列の積関数が保存されている．ここで，行列 $\boldsymbol{G} \in \mathbb{F}_q^{c\times t}, c = \sum_{i\in[n]} c_i$ を分割して，$\boldsymbol{G}_{i\cdot} \in \mathbb{F}_q^{c_1\times t}, \ldots, \boldsymbol{G}_{i\cdot} \in \mathbb{F}_q^{c_n\times t}$ とする．ここで，

$$\boldsymbol{G} = \begin{pmatrix} \boldsymbol{G}_{1\cdot}^{\top} \\ \vdots \\ \boldsymbol{G}_{n\cdot}^{\top} \end{pmatrix}. \tag{2.33}$$

また，集合 $C(\subset \mathbb{F}_q^n)$ を，$\boldsymbol{G}$ を生成行列とする $\mathbb{F}_q$ 上 (n,k) 符号とする．

マスターに秘密情報 A が入力されると，マスターは**符号化処理**を行い，各ワーカーに**シェア**と呼ばれるベクトルを保存する．各ワーカーへのシェアの保存後，マスターには秘密情報や各シェアの値，および後述の乱数の値が保存されていなくてもよいものとする．マスターが秘密情報を復元したいとき，マスターは**復号処理**を行い，秘密情報 A の推定結果 $\hat{A} \in \mathbb{F}_q \cup \{?\}$ を出力する．

〈 **符号化処理** 〉 マスターに秘密情報 A が入力されると，マスターは，秘密情報 A，および A と独立で一様分布に従う乱数ベクトル $\boldsymbol{R} \in \mathbb{F}_q^{t-1}$ の組を生成行列 $\boldsymbol{G}_{i\cdot}^{\top} \in \mathbb{F}_q^{c_i\times t}$ によって符号化し，$\tilde{A}_i := \boldsymbol{G}_{i\cdot}^{\top}(A, \boldsymbol{R}^{\top})^{\top}$(**シェア**) を得る．その後，各ワーカー $i \in [n]$ にシェア $\tilde{A}_i$ を送信する．

〈 **復号処理** 〉 A の値の復元時，マスターは，集合 $\mathcal{P} \in \mathfrak{P}_{\mathrm{qu}}$ を一つ定め，各ワーカー $i \in \mathcal{P}$ にアクセスしてシェア $\tilde{A}_i$ を得る．一方，マスターは各ワーカー $i \notin \mathcal{P}$ からはシェアの値を得ないとする．ここで，各ワーカー $i \in [n]$ に対し，$\boldsymbol{Y}_i \in \mathbb{F}_q \cup \{?\}$ を，$i \in \mathcal{P}$ のとき $\tilde{A}_i$ であり，$i \notin \mathcal{P}$ の時は消失を表す記号 ? とおく．マスターは，ベクトル $\boldsymbol{Y} := (\boldsymbol{Y}_i)_{i\in[n]}$ から，関数 $\psi\colon (\mathbb{F}_q \cup \{?\})^c \to C \cup \{?\}$

を利用して，$\psi(\boldsymbol{Y}) \in C \cup \{?\}$ を得る．$\psi(\boldsymbol{Y}) \in C$ であり，マスターは，行列 $\psi(\boldsymbol{Y})$ から，秘密情報 A を得る．すなわち，$\hat{A} \coloneqq A$ とする．

ここで，Γ に関わる諸定義について説明する．

定義 2.5.1（信頼性，秘匿性） 結託して秘密情報 A の値を得ようとするワーカー全体の集合が $\mathcal{P} \in \mathfrak{P}_{\mathrm{fo}}$ であれば，A を完全に秘匿できる，すなわち，任意の $\mathcal{P} \in \mathfrak{P}_{\mathrm{fo}}$ に対し，$\mathrm{I}(A; (\tilde{A}_i)_{i\in\mathcal{P}}) = 0$ が成立する．本論文ではこれを**秘密分散方式の秘匿性**と呼ぶ．

一方で，任意の $\mathcal{P} \in \mathfrak{P}_{\mathrm{qu}}$ に対し $\mathrm{H}(A|(\tilde{A})_{i\in\mathcal{P}}) = 0$ が成立するとする．この式は出力結果を送信するワーカー全体の集合 $\mathcal{P} \subset [n]$ が $\mathcal{P} \in \mathfrak{P}_{\mathrm{fo}}$ を満たすとき，秘密情報 A の値を常に正しく復号できるということを表すという仮定である．本論文では，これを**秘密分散方式の信頼性**と呼ぶ．

定義 2.5.2（禁止集合族，許可集合族） 互いに排反な 2 つの集合族 $\mathfrak{P}_{\mathrm{fo}}, \mathfrak{P}_{\mathrm{qu}} \subset 2^{[n]}$ をそれぞれ**禁止集合族**，**許可集合族**と呼び，組 Γ を**アクセス構造**と呼ぶ．この秘密分散方式を **Γ-秘密分散方式**と呼ぶ．

定義 2.5.3（(k,n) 閾値法）

$$\mathfrak{P}_{\mathrm{fo}} = \{\, \mathcal{P} \subset [n] \mid k-1 \geq |\mathcal{P}| \,\}, \mathfrak{P}_{\mathrm{fo}} = \{\, \mathcal{P} \subset [n] \mid k \leq |\mathcal{P}| \,\} \tag{2.34}$$

であるような秘密分散方式を (k,n) 閾値法と呼ぶ．

ここで，秘密分散法における平均保存レートを定義する．これが秘密分散法における評価基準である．

定義 2.5.4 上記の秘密分散方式において，$r_{\mathrm{av}} \coloneqq \frac{1}{n\mathrm{H}(A)} \sum_{i\in[n]} \mathrm{H}(\tilde{A}_i)$ を**平均保存レート**と呼ぶ．

上記の秘密分散方式が (k,n) 閾値法であるとき，その平均保存レートは 1 以上であることが知られている．[35].

2.5.1 (k,n) 閾値秘密分散方式の構成 [12]

(k,n) 閾値法の具体的な構成法を説明する．この構成法は Shamir [12] が提案した構成法であるが，その情報理論的な解析は [35] で行われている．n, k を，$n \geq k, n \geq 2$ を満たす任意の正整数として，q を $q \geq n+1$ を満たす任意の素数冪とする．マスターと n 個のストレージ（ワーカー）を用いる．各ワーカーはワーカー 1 からワーカー n まで番号付けされている．どのワーカーにどの番号が付与されているかはマ

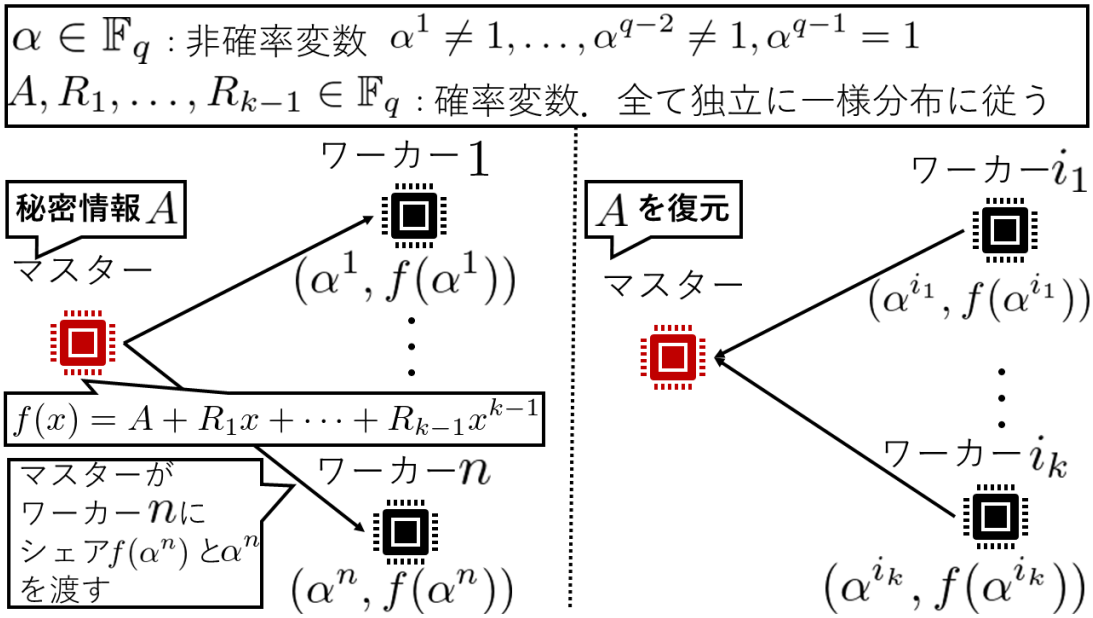

図 2.1　秘密情報の分散

スターと全ワーカーに公開されているものとする．マスターは秘密情報 $A \in \mathbb{F}_q$ を n 個のワーカーに分散して保存したい．ここで，マスターとしては，$k-1$ 個以下の任意のワーカーが結託することによって，秘密情報 A の値が漏洩することは防ぎたいとする．一方，k 個のワーカーに保存したシェアから秘密情報 A が正しく復元できるような方式にしたい．

Shamir による (k, n) 閾値法の構成法 [12] は，以下の手順で構成できる（図 2.1）．以下では，$\mathbb{F}_q$ の原始元 $\alpha \in \mathbb{F}_q$ の値を任意に一つ値とり，固定する．

〈**符号化処理**〉 マスターは秘密情報 $A \in \mathbb{F}_q$ が入力されると，乱数 $R_1, \ldots, R_{k-1}$ の実現値を選ぶ．マスターは $(\alpha^1, f(\alpha^1)), \ldots, (\alpha^n, f(\alpha^n))$ を得る．ただし，$f(x) \coloneqq A + R_1 x \cdots + R_{k-1} x^{k-1}$ は $\mathbb{F}_q$ 上多項式である．α の性質から，$\alpha^1, \ldots, \alpha^n$ はすべて値が異なることに注意する．マスターはワーカー i に $\tilde{A}_i \coloneqq f(\alpha^i)$ と α^i の組を配布する．各ワーカー i の有する値のうち，$\alpha^i \in \mathbb{F}_q$ の値は，マスターと全ワーカーに公開されている．$f(\alpha^i)$ の値はワーカー i 以外のワーカーには公開されていない．

〈**復号処理**〉 秘密情報 A の復元時において，マスターは一部のワーカーにアクセスして，それらが有するシェアの値を得るが，残りのワーカーからは受信できなかったとする[*11]．受信したワーカーのインデックス全体の集合 $\{i_1, \ldots, i_h\}$ とおく．h は受信したワーカーの個数を表す．$h \leq k-1$ であれば，マス

[*11] 例えば一定時間経過後に受信できたか否かを判断する

ターは復元できなかったことを出力する．$h \geq k$ であれば，マスターは $(\alpha^{i_1}, f(\alpha^{i_1})), \ldots, (\alpha^{i_k}, f(\alpha^{i_k}))$ から復号し，正しく A の値を得る．

$k-1$ 個以下の任意のワーカーの結託に対して秘密情報を秘匿可能 (**秘匿性**) であることと，k 個以上の任意のシェアから秘密情報を復元可能（**信頼性**）であることは，次のように定式化される．

命題 2.5.1　$A, R_1, \ldots, R_{k-1}$ はすべて独立に $\mathbb{F}_q$ 上一様分布に従う確率変数とする．$k-1$ 個以下の任意のワーカーからなる集合 $\{i_1, \ldots, i_h\} \subset [n]$，$|\{i_1, \ldots, i_h\}| \leq k-1$ に対して，$\mathrm{I}(A\,;\, f(\alpha^{i_1}), \ldots, f(\alpha^{i_h})) = 0$ が成立する．一方，$|\{i_1, \ldots, i_h\}| \geq k$ を満たす任意の $\{i_1, \ldots, i_h\} \subset [n]$ に対して，$\mathrm{H}(A|f(\alpha^{i_1}), \ldots, f(\alpha^{i_h})) = 0$ が[*12]成立する．

秘匿性の証明は [35] 参照．信頼性（消失訂正能力）は，多項式補間により秘密情報 $A = f(0)$ を復元できる [12] ことから，明らかに成立する．

注意 2.5.1　前小節のノーテーションに合わせると，次のようになる．禁止集合族は $\mathfrak{P}_{\mathrm{fo}}$ は $\{\, \mathcal{P} \subset [n] \mid k-1 \geq |\mathcal{P}| \,\}$ であり，許可集合族は $\mathfrak{P}_{\mathrm{qu}}$ は $\{\, \mathcal{P} \subset [n] \mid k \leq |\mathcal{P}| \,\}$ である．これはアクセス構造の単調性を満たす．(k, n) 閾値秘密分散方式の構成 [12] は $\mathbb{F}_q$ 上 (n, k) Reed Solomon 符号の生成行列 $\mathbf{RS}_{n\times k}$ を用いる．ただし，q は n より大きい素数．秘密分散方式 $(\, (c_i, \boldsymbol{G}_{i\cdot})_{i\in[n]}, t-1, \psi \,)$ について，$c_i = 1$ であり，$\boldsymbol{G}_{i\cdot}$ は $\mathbf{RS}_{n\times k}$ の第 i 行目であり，$t = k$ である．ψ は $\mathbb{F}_q$ 上 (n, k) Reed Solomon 符号の限界ハミング消失距離復号である．

また，この構成法で構成した (k, n) 閾値法の平均保存レートはちょうど 1 である [35].

[*12] 関数 $\psi\colon \mathbb{F}_q^k \to \mathbb{F}_q$ が存在し，k 個以上の任意のワーカーからなる集合 $\{i_1, \ldots, i_h\} \subset [n]$, $|\{i_1, \ldots, i_h\}| \geq k$ に対し，$\Pr(\psi(f(\alpha^{i_1}), \ldots, f(\alpha^{i_h})) = A) = 1$.

第 3 章

分散符号化計算方式の従来研究

本章では，本論文の一つ目の主要な結果である (a) 計算時間，(b) 信頼性 (誤り訂正能力) を考慮した新たな分散符号化計算方式 (4 章) と，本論文の二つ目の主要な結果である (a) 計算時間，(b) 信頼性 (消失訂正能力)，(c) 情報秘匿性を考慮した新たな分散符号化計算方式 (5 章) に関係する従来研究を説明する．

3.1 節では，本論文における計算したい関数の目標と，その計算に対する分散計算方式の概要を説明する．

3.2 節は，本論文の一つ目の主要な結果に関する節であり，従来から研究されてきた (a) 計算時間と (b) 信頼性 (誤り/消失訂正能力) を考慮した分散符号化計算方式の一般的な手順の説明を行う．これにより，3.2.2 節で説明する方式 [1] をはじめとした従来の方式では，列ごとの誤りのみを訂正可能な方式であることを確認する．これに対し，第 4 章で提案するグループ型分散符号化計算方式は，従来とは異なり，行列全体に対する誤り，例えばランクが一定以下の誤り行列を訂正可能である．

3.3 節は，本論文の二つ目の主要な結果に関する節であり，第 5 章で提案する方式の従来研究を示す．本論文における (c) 情報秘匿性とは，行列 $\boldsymbol{A}$ の値を知るために，一部のワーカーが結託して自身に渡された情報を集約しても，$\boldsymbol{A}$ の値に関する情報が漏洩しないという秘匿性を保持しつつ，マスターが積行列 $\boldsymbol{AB}$ を分散符号化計算できることを表す．ただし，行列 $\boldsymbol{B}$ の値は各ワーカーに公開してもよいとする．従来研究として，(b) 信頼性 (誤り/消失訂正能力) と (c) 情報秘匿性を有する分散計算方式は行われていたが，分散計算によりシステム全体の (a) 計算時間を短くできる優位性も含めては論じられていなかった．そこで，最初に，分散計算に限らず，分散ストレージや分散型情報検索システムなど，ワーカーを分散するシステムを利用した従来の分散方式にはどのようなものがあるかを説明する．特にその中で，(b) 信頼性 (誤り/消失訂正能力) と (c) 情報秘匿性を有する従来の分散計算方式 [5] はどのようなものかを説明する．これに対し，5 章では，(a) 計算時間，(b) 信頼性 (誤り/消失訂正能

力)，(c) 情報秘匿性を全て考慮した分散符号化計算方式であるグループ型分散秘匿符号化計算方式を提案する．

3.1 分散計算方式

本論文では，有限体 $\mathbb{F}_q$ 上 $k_A \times l$ 行列 $\boldsymbol{A}$ と $l \times k_B$ 行列 $\boldsymbol{B}$ の積行列 $\boldsymbol{AB}$ の計算方式を論じる．ただし，$\mathbb{F}_q$ は任意の (位数 q の) 有限体[*1]で，k_A, k_B, l は 2 以上の任意の正整数である．また，行列 $\boldsymbol{A}$ の値が 1 回のみ主要な計算機（マスター）に入力され，マスターに $\boldsymbol{A}$ の値が全部もしくは一部保存される．行列 $\boldsymbol{B}$ については，異なる値がマスターに何回も入力され，その都度，マスターは積行列 $\boldsymbol{AB}$ の値の計算を試みる．

定義 3.1.1　本論文においては，システムの総計算時間とは，$\boldsymbol{B}$ の値がマスターに入力され，マスターが最終的な計算結果 $\boldsymbol{A\hat{B}}$ を得るまでの $\mathbb{F}_q$ 上の四則演算の回数とする．[*2]また，行列 $\boldsymbol{A}$ の値は一回しか入力されないため，後述の分散計算や分散符号化計算の前処理の計算時間はシステムの総計算時間に考慮に入れないとする．また，$\mathbb{F}_q$ 上四則演算は，すべて $\mathbb{F}_q$ 上演算として 1 回として数える．

定義 3.1.2　マスターが行列 $\boldsymbol{A}, \boldsymbol{B}$ を入力された下で，単独で $\boldsymbol{AB}$ を計算する方式を**単独方式**と呼ぶ．

命題 3.1.1　単独方式の計算時間（$\mathbb{F}_q$ 上演算回数）は，$\mathbb{F}_q$ 上の四則演算の回数は合計 $k_A k_B(2l-1)$ 回である．

証明　$\mathbb{F}_q$ 上 l 次元ベクトル同士の内積計算においては，$\mathbb{F}_q$ 上乗算が l 回，$\mathbb{F}_q$ 上加算が $l-1$ 回かかる．よって，$\mathbb{F}_q$ 上乗算が $k_A k_B l$ 回，$\mathbb{F}_q$ 上加算が $k_A k_B(l-1)$ 回であることから，命題が直ちに示される．□

単独方式よりも計算時間を抑えたい場合，マスターのほかに，並列計算のために用意された $n(\geq 2)$ 個の計算機（ワーカー）を利用して，分散計算を行う．以下の方式に限っては，簡単のため，k_A は n の倍数とする．

〈 **前処理** 〉　マスターは行列 $\boldsymbol{A}$ を行で等分割して，n 個の $(k_A/n)\times l$ 行列 $\boldsymbol{A}_{1\cdot}, \ldots, \boldsymbol{A}_{n\cdot}$ を得る．マスターは行列 $\boldsymbol{A}_{i\cdot}$ を各ワーカー $i \in [n]$ に保存する．

〈 **計算処理** 〉　マスターは行列 $\boldsymbol{B}$ を全ワーカーに送信する．各ワーカー i は，行列 $\boldsymbol{A}_{i\cdot}$ と行列 $\boldsymbol{B}$ から，$\boldsymbol{A}_{i\cdot}\boldsymbol{B}$ を計算する．

[*1] 後の節で，q は素数冪の場合に限られ，かつ位数 q が与えられたときに有限体 $\mathbb{F}_q$ の構造は一意に決まることが説明される．

[*2] 後述の分散計算や分散符号化計算を行う場合は，$\boldsymbol{A\hat{B}} = \boldsymbol{AB}$ になるとは限らない．

〈**集約処理**〉 マスターは，ワーカーの出力結果 $(\boldsymbol{A}_i.\boldsymbol{B})_{i\in[n]}$ を受信し，$\boldsymbol{AB}$ を得る．

ワーカーとマスターが同じ計算速度であり，さらに，通信時間や集約処理時間を無視することにする．上記の方式では，各ワーカーが保存する部分行列のサイズが行列 $\boldsymbol{A}$ の $1/n$ 倍であり，それらを並列に計算することから，分散計算の計算時間は単独のマスターの計算時間の $1/n$ 倍程度に抑えられる．具体的には，$\mathbb{F}_q$ 上乗算が $k_{\mathrm{A}}k_{\mathrm{B}}l/n$ 回，$\mathbb{F}_q$ 上加算が $k_{\mathrm{A}}k_{\mathrm{B}}(l-1)/n$ 回であるため，$\mathbb{F}_q$ 上の四則演算の回数は合計 $k_{\mathrm{A}}k_{\mathrm{B}}(2l-1)/n$ 回である．ここで，上記では $\boldsymbol{A}$ を分割した分散計算を説明したが，$\boldsymbol{B}$ を分割する，あるいは両方を分割することにより，同様に分散計算方式ができる．

ここで，本論文におけるマスターとワーカーの仮定をここに記載する．

- マスターと各ワーカーの有限体 $\mathbb{F}_q$ 上の四則演算 1 回を行う計算時間はすべて等しい．
- マスターの出力において誤りや消失を発生することはない．一方で，一部のワーカーの出力において誤りや消失が発生することがある．
- 行列 $\boldsymbol{A}$ の値を全ワーカーに秘匿しながら積行列 $\boldsymbol{AB}$ の計算を試みる場合，マスターが行列 $\boldsymbol{A}$ の値に関する情報をすべて有する．一方，各ワーカーは計算方式の途中でマスターから渡される情報のみを行列 $\boldsymbol{A}$ の値に関する情報として有する．
- 行列 $\boldsymbol{A}$ の値を全ワーカーに秘匿しながら積行列 $\boldsymbol{AB}$ の計算を試みる場合，一部のワーカーは結託して自身の持つ情報を持ち寄り，$\boldsymbol{A}$ の値を得ようとする．

そのため，序論でも述べた通り，分散計算することによって利点 (a) が生じる一方で，欠点 (b)，(c) が生じる．

本論文では，利点 (a) を生かしつつ欠点 (b) を解消する方法として第 4 章でグループ型分散符号化計算方式を提案する．加えて，利点 (a) を生かしつつ欠点 (b)，(c) を解消する方法としてグループ型分散秘匿符号化計算方式を提案する．(a) と (b) を考慮した分散符号化計算方式は従来から論じられてきた方式 [1] であり，詳細は第 3 章で説明する．これは，デジタルデータの分散ストレージにおいて発生した誤り，消失を訂正する方式である誤り訂正符号 (例えば (n,k) Reed Solomon 符号 [32]) の原理を利用している．誤り訂正符号については 2.4.1 節で説明する．第 4 章で提案するグループ型分散符号化計算方式は，従来と異なる誤り行列を訂正可能でありながら，分散計算によって単独方式よりも計算時間が短縮された方式である．(a)，(b)，(c) を考慮した分散符号化計算方式であるグループ型分散秘匿符号化計算方式は，本論文で新たに提案する方式であり，第 5 章で具体的な構成を与える．この方式は，従来から存在する分散秘匿符号化計算方式 [5] に，ワーカーのグループの概念を導入した方式で

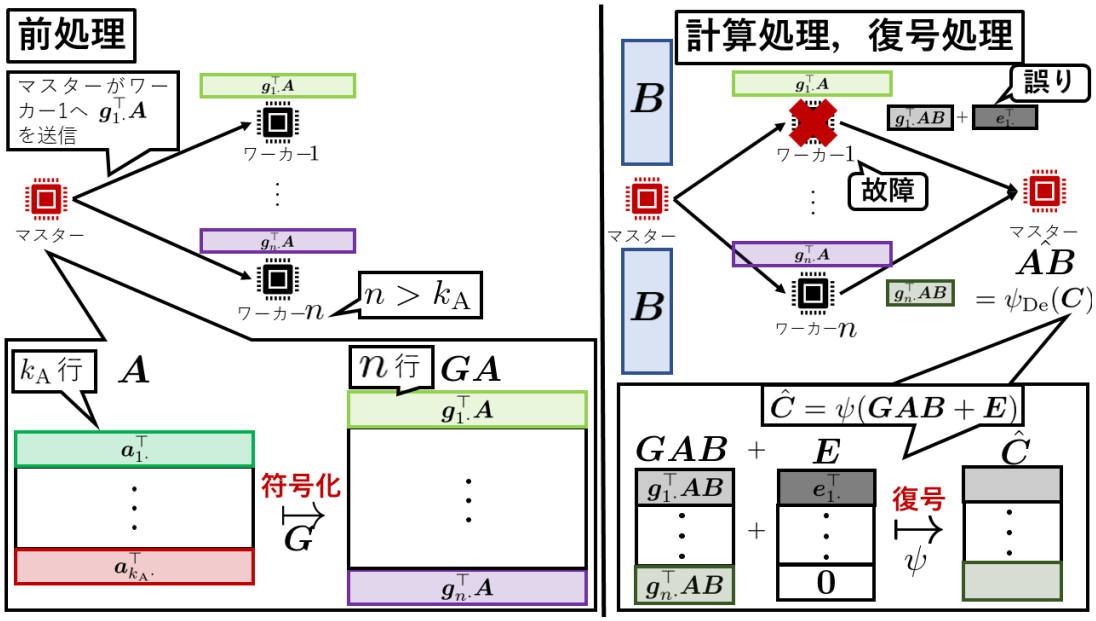

図 3.1　分散符号化計算方式

ある．ただし，分散秘匿符号化計算方式 [5] は (b),(c) に着目した研究であり，(a) に着目していない点で，評価基準がグループ型分散秘匿符号化計算方式と異なることに注意する．また，分散秘匿符号化計算方式 [5] 自体は，秘密情報を含む保管したいデジタルデータに保管者の秘密情報が含まれる場合，分散ストレージにおいて発生した誤り，消失を訂正しつつ，情報漏洩に耐性を持たせたストレージ方式である秘密分散法 ((k,n) 閾値秘密分散法 [12]）の原理を利用している．秘密分散法についても後述する．第 5 章で提案するグループ型分散符号化計算方式は，上記の方式を分散計算することで，(b) と (c) の点で優れた性質を持ちながら，単独方式よりも計算時間が短縮された方式である．

3.2　計算時間と信頼性を考慮した従来の分散符号化計算方式

(a) 計算時間と (b) 信頼性を考慮した従来の分散符号化計算方式の一般的手順を 3.2.1 節で説明する．さらに，従来の代表的な方式 [1] を 3.2.2 節で説明する．

3.2.1　従来の分散符号化計算方式の一般的手順

行列の積計算における従来の分散符号化計算方式の一般的な手順を説明する．q を任意の素数冪，$k_{\mathrm{A}}, k_{\mathrm{B}}, l, n$ を $2 \leq k_{\mathrm{A}} < n, 2 \leq k_{\mathrm{B}}, l$ を満たす任意の正整数とおく．本

論文では，行列 $\boldsymbol{A} \in \mathbb{F}_q^{k_\mathrm{A} \times l}, \boldsymbol{B} \in \mathbb{F}_q^{l \times k_\mathrm{B}}$ の積行列 $\boldsymbol{AB} \in \mathbb{F}_q^{k_\mathrm{A} \times k_\mathrm{B}}$ の計算方式を論じる．行列 $\boldsymbol{A}$ の値は固定され，行列 $\boldsymbol{B}$ の値が入力されるごとに積行列 $\boldsymbol{AB}$ の値を計算するが，ワーカーの計算中に誤りが発生する．

注意 3.2.1　パラメータ $k_\mathrm{A}, k_\mathrm{B}, l, n, q$ の値を定める順番について論じる．有限体の位数 q は一番最初に定めるので，残りのパラメータ $k_\mathrm{A}, k_\mathrm{B}, l, n$ の値を定める順番を論じる．

通常の分散計算方式では，計算機の個数 n の値を定めてから，行列のサイズに関わるパラメータ $k_\mathrm{A}, k_\mathrm{B}, l$ の値を定める．例えば，符号化したい巨大な $k'_\mathrm{A} \times l'$ 行列 $\boldsymbol{A}'$ が入力されると，それを行方向と列方向それぞれで n 分割することで，$k_\mathrm{A} \times l = (k'_\mathrm{A}/n) \times (l'/n)$ 行列 $\boldsymbol{A}$ を生成する [36]．なお，この節では，主に $\boldsymbol{A}$ を符号化した行列を行方向で分割して，各ワーカーに分散するのみを考える．すなわち，行列 $\boldsymbol{A}'$ が入力されると，それを行方向で n 分割することで，$k_\mathrm{A} \times l = (k'_\mathrm{A}/n) \times l'$ 行列 $\boldsymbol{A}$ を生成する．符号化したい行列を行方向で分割すること自体は [1] の方式と同様である．4.2 節では，また異なる行列の分割と，各ワーカーへの分散を考える．

一方，誤り訂正符号では，符号語長 n，情報ベクトル長 k の値を同時に定めてから符号化関数，復号関数を実装する．誤り訂正符号で主に用いられるブロック符号化の符号化関数は，長い入力系列を一定の長さ k の情報ベクトルに分割し，各情報ベクトルを独立に (n, k) 符号化することで，長さ n の符号語を複数個生成する [19][37]．このことに対応させて考えると，$k_\mathrm{A}, k_\mathrm{B}, l$ の値を定めてから n の値を定めることが普通である．すなわち，$\boldsymbol{A}$ に相当する巨大な行列がマスターに入力されたら，マスターはその行列を $k_\mathrm{A} \times l$ 部分行列ごとに分割する，ということである．

しかし，本論文においては全パラメータ $k_\mathrm{A}, k_\mathrm{B}, l, n$ の値が定まった下での計算時間，信頼性を論じるため，$k_\mathrm{A}, k_\mathrm{B}, l, n$ の値を定める順番は問わない．第 5 章の議論でも同様に，パラメータ $k_\mathrm{A}, k_\mathrm{B}, l$ と，ワーカーの台数に関わるパラメータ $n_\mathrm{A}, n_\mathrm{B}, h_\mathrm{A}$ を定める順番は問わない．

上記の計算を分散符号化計算方式で行う．この方式では，マスターと n 個のワーカーを利用する．ここで，$q, k_\mathrm{A}, k_\mathrm{B}, l, n$ が与えられた下での**分散符号化計算方式** $(\boldsymbol{G}, \pi, \psi)$ の手順を以下で説明する．

マスターにはフルランク行列 $\boldsymbol{G} \in \mathbb{F}_q^{n \times k_\mathrm{A}}$ と関数 ψ が，全ワーカーには関数 π が保存されている．ここで，$\boldsymbol{G}$ の第 $i \in [n]$ 行目を $\boldsymbol{g}_{i\cdot}^\top \in \mathbb{F}_q^{1 \times k_\mathrm{A}}$ と表す．また，集合 C $(\subset \mathbb{F}_q^{n \times l})$ は，$\boldsymbol{G}$ を生成行列とする (n, k_A) 符号 $C_\mathrm{H} \subset \mathbb{F}_q^n$ の l 個の直積集合を表す．符号 C_H の復号関数を $\psi_\mathrm{H} \colon \mathbb{F}_q^n \to C_\mathrm{H} \cup \{?\}$ とおく．記号 ? は復号不能を表す．C_H と $\mathbb{F}_q^{k_\mathrm{A}}$ の間には，全単射 $\psi_\mathrm{De} \colon C \to \mathbb{F}_q^{k_\mathrm{A} \times k_\mathrm{B}}$ が存在する．これは，任意の $\boldsymbol{m} \in \mathbb{F}_q^{k_\mathrm{A}}$ に対し，$\psi_\mathrm{De}(\boldsymbol{Gm}) = \boldsymbol{m}$ が成立する関数である．

マスターに行列 $\boldsymbol{A}$ が入力されると，ワーカーとマスターは**前処理**を行う．マスターに行列 $\boldsymbol{B}$ が入力されるごとに，ワーカーは**計算処理**を，マスターは**復号処理**を行い，積行列 $\boldsymbol{AB}$ の推定計算結果 $\hat{\boldsymbol{AB}} \in \mathbb{F}_q^{k_\mathrm{A} \times k_\mathrm{B}}$ を出力する．より詳細には，次の手順で計算する．

〈 **前処理** 〉 (図 3.1 の左参照)

マスターは行列 $\boldsymbol{A}$ を符号化した行列 $\boldsymbol{GA} \in \mathbb{F}_q^{n \times l}$ を得て，全ての $i \in [n]$ に対し，行列 $\boldsymbol{GA}$ の第 i 行目ベクトル $\boldsymbol{g}_{i\cdot}^\top \boldsymbol{A} \in \mathbb{F}_{q'}^{l}$ をワーカー i に保存する．

〈 **計算処理** 〉 (図 3.1 の中央参照)

マスターは行列 $\boldsymbol{B}$ を全ワーカーに送信する．各ワーカー i は，ベクトル $\boldsymbol{g}_{i\cdot}^\top \boldsymbol{A}$ と行列 $\boldsymbol{B}$ から，$\pi(\boldsymbol{g}_{i\cdot}^\top \boldsymbol{A}, \boldsymbol{B}) = \boldsymbol{g}_{i\cdot}^\top \boldsymbol{AB} \in \mathbb{F}_q^{1 \times k_\mathrm{B}}$ を計算する．本論文では，関数 $\pi \colon \mathbb{F}_{q'}^{1 \times l} \times \mathbb{F}_q^{l \times k_\mathrm{B}} \to \mathbb{F}_q^{l}$ を**積関数**と呼ぶ．ここで，各ワーカー i の計算中に誤り $\boldsymbol{e}_{i\cdot} \in \mathbb{F}_q^{b}$ が発生し，出力結果が $\boldsymbol{y}_{i\cdot} \coloneqq \boldsymbol{g}_{i\cdot}^\top \boldsymbol{AB} + \boldsymbol{e}_{i\cdot}$ になる．

〈 **復号処理** 〉 (図 3.1 の右参照)

マスターは，全ワーカーの出力結果 $\boldsymbol{y}_{1\cdot}$，$\ldots$，$\boldsymbol{y}_{n\cdot}$ を受信し，行列 $\boldsymbol{Y} \coloneqq (\boldsymbol{y}_{1\cdot}, \ldots, \boldsymbol{y}_{n\cdot})^\top \in \mathbb{F}_q^{n \times l}$ を得る．ここで，$\boldsymbol{E} \in \mathbb{F}_q^{n \times l}$ を，第 i 行目が $\boldsymbol{e}_{i\cdot}^\top$ である行列と定義すると，$\boldsymbol{Y} = \boldsymbol{GAB} + \boldsymbol{E}$ であることが示される．マスターは，関数

$$\psi \colon \mathbb{F}_q^{n \times l} \to (\mathbb{F}_q \cup \{?\})^{n \times l} \; ; \; \boldsymbol{Y} \mapsto \psi(\boldsymbol{Y}) = (\psi_\mathrm{H}(\boldsymbol{y}_{\cdot 1}), \ldots, \psi_\mathrm{H}(\boldsymbol{y}_{\cdot l})) \quad (3.1)$$

を利用して，$\boldsymbol{Y}$ から $\hat{\boldsymbol{C}} \coloneqq \psi(\boldsymbol{Y})$ を得る．また，$\boldsymbol{y}_{\cdot j}$ は $\boldsymbol{Y}$ の第 $j \in [l]$ 列目を表す．$\psi_\mathrm{H}(\boldsymbol{y}_{\cdot 1}), \ldots, \psi_\mathrm{H}(\boldsymbol{y}_{\cdot l}) \in C_\mathrm{H}$ ならば，マスターは，行列 $\hat{\boldsymbol{AB}}$ を $(\psi_\mathrm{De}(\psi(\boldsymbol{y}_{\cdot 1})), \ldots, \psi_\mathrm{De}(\psi(\boldsymbol{y}_{\cdot l})))$ とおいて，これを出力する．

この方式では，任意の $j \in [k_\mathrm{B}]$ に対し，$\psi(\boldsymbol{y}_{\cdot j}) = \boldsymbol{GAb}_{\cdot j}$ が成立するならば，正しく計算ができる．これが，この方式の (b) 信頼性 (誤り訂正能力) に関する評価である．ただし，$\boldsymbol{b}_{\cdot j}$ は $\boldsymbol{B}$ の第 j 列目である．

3.2.2 計算結果が積行列の各列を Reed Solomon 符号化した行列である従来の方式 [1]

$q \geq n + 1$ が成立するときの従来の分散符号化計算方式としては，積行列 $\boldsymbol{AB}$ の各列を Reed Solomon 符号化（**RS 符号化**），限界ハミング距離復号する方式がある [1]. 本論文では，これを **RS 方式**と呼ぶ．$\boldsymbol{G}$ を，$\mathbb{F}_q$ 上 (n, k) Reed Solomon 符号の生成行列とする．本論文では，ハミング重み関数を $\mathrm{w}(\cdot)$ と表す．復号関数 $\psi_\mathrm{H} \colon \mathbb{F}_q^n \to C_\mathrm{H} \cup \{?\}$ を，限界ハミング距離復号と定義する．$\lfloor (n - k_\mathrm{A})/2 \rfloor$ を t と表す．

命題 3.2.1（(b) に関する評価） 誤り行列 $\boldsymbol{E}$ が与えられているとする．RS 方式で

は，任意の $j \in [k_{\mathrm{B}}]$ に対し，$\mathrm{w}(\boldsymbol{e}_{\cdot j}) \leq t$ が成立するならば，正しく計算ができる．ただし，$\boldsymbol{e}_{\cdot j}$ は $\boldsymbol{E}$ の第 j 列目である．

証明　誤りが発生しなかった場合の分散計算の結果は，$\boldsymbol{GAB}$ である．これの各列ベクトル $\boldsymbol{GAb}_{\cdot j}$ はすべて符号 C_{H} の符号語である．ただし，$\boldsymbol{b}_{\cdot j}$ は $\boldsymbol{B}$ の第 j 列目である．よって，各 $j \in [l]$ に対して，$\mathrm{w}(\boldsymbol{e}_{\cdot j}) \leq t$ が成立するならば，$\psi_{\mathrm{H}}(\boldsymbol{GAb}_{\cdot j} + \boldsymbol{e}_{\cdot j}) = \boldsymbol{GAb}_{\cdot j}$ が成立する．よって命題が示された．□

上記の通り，RS 方式では，ワーカーの試みる分散計算の結果 $\Pi(\boldsymbol{AB}) = \mathbf{RS}_{n \times k_{\mathrm{A}}} \boldsymbol{AB}$ が**積行列 $\boldsymbol{AB}$ の各列**を符号化した行列である．これによって，列ごとにハミング重みが低い誤りのみを訂正可能な方式であることを確認できた．

本論文 4 章では，ワーカーの試みる分散計算の結果が**積行列 $\boldsymbol{AB}$ 自体**をまとめて符号化した行列となるように方式を変更することにより，ワーカー毎の誤りのみでなく，ワーカー全体に及ぶ誤りをある条件の下で訂正可能な方式を提案することを説明する．

3.3　秘匿性を考慮した分散方式の従来研究

最初に，分散計算に限らず，分散ストレージや分散型情報検索システムなど，ワーカーを分散するシステムを利用した従来の分散方式にはどのようなものがあるかを説明する．次に，その中で，(b) 信頼性 (消失訂正能力) と (c) 情報秘匿性を有する従来の分散計算方式 [5] はどのようなものかを説明する．最後に，第 5 章で提案する方式の新規性について触れる．

3.3.1　秘匿性を考慮した分散型情報検索システム方式の従来研究

秘匿性を考慮した分散方式の従来研究としては，すでに説明した秘密分散 (Secret Sharing)[12] のほかには，Private Information Retrieval(PIR)[13]，Robust Byzantine PIR[14][15]，Private Computation[16] がある．上記の Robust Byzantine PIR，Private Computation は PIR の派生なので，ここでは PIR の概要を説明する．

PIR の手順を [38] に則り説明する．$m, n, l_{\mathrm{w}}, l_{\mathrm{a}}, f$ を $m, n \geq 2$ を満たす任意の正整数，q を任意の素数冪とする．$\mathrm{PIR}\left\{ (\theta, \boldsymbol{Q}^{(\theta)}, \boldsymbol{A}^{(\theta)}) \,\middle|\, \theta \in [m] \right\}$ は，メッセージの検索者（ユーザ）が $n(\geq 2)$ 台のデータベース（以降，DB）に保存されている m 個のメッセージにアクセスして，ユーザが検索したいメッセージのインデックス $\theta \in [m]$ が何かを各 DB に秘匿しつつメッセージの値を得るシステムである．$\left(\mathbb{F}_q^{l_{\mathrm{w}}}\right)^m$ 上一様分布に従う確率変数ベクトル $\boldsymbol{W} = (W_1, \ldots, W_m)$ **メッセージベクト**

ルが n 個のデータベース (DBs) に保存されている．ただし，各**メッセージシンボル** $W_1 = (W_{1,1}, \ldots, W_{1,l_\mathrm{w}}), \ldots, W_m = (W_{m,1}, \ldots, W_{m,l_\mathrm{w}})$ は独立に $\mathbb{F}_q^{l_\mathrm{w}}$ 上一様分布に従う．$n(\geq 2)$ は DB の個数，$m(\geq 2)$ は**メッセージベクトル長**，l_w は**メッセージシンボル長**である．ユーザは W_θ（$\theta \in [m]$ は**メッセージシンボルのインデックス**）の実現値を次の 3 つのフェーズに従って検索する．

クエリ送信処理　ユーザは確率変数 $Q_n^{(\theta)}$ (**クエリ**) の実現値を各 DB $i \in [n]$ に送信．ここで，各 $\theta \in [m]$ に対する**クエリベクトル** $\boldsymbol{Q}^{(\theta)} := (Q_1^{(\theta)}, \ldots, Q_n^{(\theta)})$ は集合 $[u]^n$ に値をとる確率変数ベクトル．

応答処理　各 DB$i \in [n]$ はクエリ $Q_i^{(\theta)}$ の実現値とメッセージベクトル $\boldsymbol{W}$ の実現値から確率変数 $A_i^{(\theta)}$ (**応答値**) の実現値をユーザに送信する．ここで，**応答値ベクトル** $\boldsymbol{A}^{(\theta)} = (A_1^{(\theta)}, \ldots, A_n^{(\theta)})$ は $\mathbb{F}_q^{l_\mathrm{a}}$ 上に値をとる確率変数ベクトル．

復号処理　ユーザは $\boldsymbol{Q}^{(\theta)}$ と $\boldsymbol{A}^{(\theta)}$ から W_θ を復号する．

ここで，任意の $\theta \in [m], i \in [n]$ に対して，$\mathrm{H}(A_i^{(\theta)}|\boldsymbol{W}, Q_i^{(\theta)}) = 0$ および $\mathrm{H}(W_\theta|\boldsymbol{A}^{(\theta)}, \boldsymbol{Q}^{(\theta)}) = 0$ を仮定する．前半は，各 DBi において，応答値がクエリメッセージベクトルの関数であることを表す．後半は，応答値ベクトルとクエリベクトルから所望のメッセージを正しく復元できることを表す．また，任意の $\theta \in [m]$ に対して $\mathrm{I}(\boldsymbol{W}\,;\,\boldsymbol{Q}^{(\theta)}) = 0$ を仮定する．これは，所望のメッセージのインデックス θ に対して，クエリベクトル $\boldsymbol{Q}^{(\theta)}$ の実現値を作るときにメッセージ W_θ の実現値の情報が一切得られていないことを表す．さらに，各 $\theta, \theta' \in [m], i \in [n], q \in [u]$ に対し，$\Pr\left(Q_i^{(\theta)} = q\right) = \Pr\left(Q_i^{(\theta')} = q\right)$ が成立する．これは，各 DBi に送信するクエリ $Q_i^{(\theta)}$ の確率分布が θ によらず一定であり，したがって DBi は受信したクエリから θ の値が何かを読み取ることはできないことを表す．上記の方式をパラメータ $(m, n, l_\mathrm{w}, l_\mathrm{a}, u, q)$ の下での PIR と呼ぶ．PIR の評価基準は，次のダウンロードレートである．

定義 3.3.1（ダウンロードレート）　上記の PIR のダウンロードレートを次のように定義する．

$$\min_{\theta \in [m]} \frac{l_\mathrm{w}}{\mathrm{H}\left(\boldsymbol{A}^{(\theta)}|\boldsymbol{Q}^{(\theta)}\right)}. \tag{3.2}$$

以上のように，PIR はユーザの検索対象を秘密情報とした分散型情報検索システムである．その派生として，DB の応答値の誤り，消失訂正を考慮した Byzantine PIR, Robust PIR などの方式も研究されている．

3.3.2 しきい値秘密分散法を利用した分散秘匿符号化計算方式 [5]

本小節では，信頼性と情報秘匿性を有する従来の分散計算方式として，しきい値秘密分散法を利用した分散秘匿符号化計算方式 [5] を説明する．[*3] この方式の評価基準は信頼性（消失訂正能力）および秘匿性である．計算時間を評価基準としていない点で本論文とは異なるが，本論文の提案方式の元であるため，以下でその方式を説明する．

この方式もまた行列 $\boldsymbol{A} \in \mathbb{F}_q^{k_\mathrm{A} \times l}, \boldsymbol{B} \in \mathbb{F}_q^{l \times k_\mathrm{B}}$ の積行列 $\boldsymbol{AB} \in \mathbb{F}_q^{k_\mathrm{A} \times k_\mathrm{B}}$ の計算方式である．この方式は，h 個のワーカーの応答値から $\boldsymbol{AB}$ を復元可能で，$h-1$ 個以下のワーカーが結託しても $\boldsymbol{A}$ の情報がまったく漏洩しない方式である．ただし，$\boldsymbol{B}$ の値はワーカーに公開してもよいとする．行列 $\boldsymbol{A}$ の値は固定され，行列 $\boldsymbol{B}$ の値が入力されるごとに積行列 $\boldsymbol{AB}$ の値を計算する．この方式では，マスターと n 個のワーカーを利用する．マスターに行列 $\boldsymbol{A}$ が入力されると，ワーカーとマスターは**前処理**を行う．マスターに行列 $\boldsymbol{B}$ が入力されるごとに，ワーカーは**計算処理**を，マスターは**復号処理**を行い，積行列 $\boldsymbol{AB}$ の推定計算結果 $\hat{\boldsymbol{AB}} \in \mathbb{F}_q^{k_\mathrm{A} \times k_\mathrm{B}}$ を出力する．詳細は次の通りである．

〈 **前処理** 〉 行列 $\boldsymbol{A}$ を入力されると，マスターは行列 $\tilde{\boldsymbol{A}}_{i\cdot} := \boldsymbol{A} + \alpha^i \boldsymbol{R}_1 + \cdots + \alpha^{i(h-1)} \boldsymbol{R}_{h-1} \in \mathbb{F}_q^{k_\mathrm{A} \times l}$ を得て，行列 $\tilde{\boldsymbol{A}}_{i\cdot}$ を各ワーカー i に保存する．$\boldsymbol{R}_1, \ldots, \boldsymbol{R}_{h-1} \in \mathbb{F}_q^{k_\mathrm{A} \times l}$ は乱数行列である．$\alpha \in \mathbb{F}_q$ は $\mathbb{F}_q$ の原始元で，値を一つ任意に固定する．この値はワーカーに公開してもよい．

〈 **計算処理** 〉 行列 $\boldsymbol{B}$ をマスターから全ワーカーに送信されると，各ワーカー $i \in [n]$ は，$\tilde{\boldsymbol{A}}_{i\cdot}$ と $\boldsymbol{B}$ から

$$\tilde{\boldsymbol{A}}_{i\cdot}\boldsymbol{B} = \boldsymbol{AB} + \alpha^i \boldsymbol{R}_1 \boldsymbol{B} + \cdots + \alpha^{i(h-1)} \boldsymbol{R}_{h-1} \boldsymbol{B} \in \mathbb{F}_q^{k_\mathrm{A} \times k_\mathrm{B}} \tag{3.3}$$

を計算し，$\tilde{\boldsymbol{A}}_{i\cdot}\boldsymbol{B}$ をマスターに送信する．ここで，一部のワーカー i からは値を受信できなかったとする．マスターが正しく値を受信したワーカー全体の集合を $\mathcal{P} \subset [n]$ とおく．残りのワーカーからは値が受信できなかったとする．

〈 **復号処理** 〉 マスターが正しく値を受信したワーカー全体の集合 $\mathcal{P} = \{i_1, \ldots, i_h\}$, $h \leq n$ がある条件を満たすならば，マスターは，すべての計算結果

[*3] 本論文と部分的に問題設定が異なるが，本節では本論文に合わせた問題設定で説明する．例えば，[5] では，値を公開しても良い行列 $\boldsymbol{B}$ を各ワーカーに保存しており，値を公開したくない行列 $\boldsymbol{A}$ をマスターがワーカーに送信し，$\boldsymbol{AB}$ を計算する．ほかには，$\boldsymbol{A}, \boldsymbol{B}$ を両方秘匿する方式も論じている．また，秘匿性を弱める代わりにシェアのサイズを小さくするランプ型秘密分散方式を利用した方式も論じている．

$\tilde{\boldsymbol{A}}_{i_1\cdot}\boldsymbol{B}, \ldots, \tilde{\boldsymbol{A}}_{i_h\cdot}\boldsymbol{B}$ から，推定計算結果 $\hat{\boldsymbol{AB}} \in \mathbb{F}_q^{k_\mathrm{A} \times k_\mathrm{B}}$ を得る．満たさなかった場合は，マスターは，計算できなかったことを表す記号？を出力する．

(b)，(c) に関して，直ちに次のことが示される．

命題 3.3.1（(b) に関する評価） マスターがすべてのワーカー $i \in \mathcal{P}$ のシェアから $\boldsymbol{AB}$ を復元しようとする．このとき，$|\mathcal{P}| \geq h$ であれば，$\boldsymbol{AB}$ を復元可能である，すなわち，$\mathrm{H}(\boldsymbol{AB} | (\tilde{\boldsymbol{A}}_{i\cdot}\boldsymbol{B})_{i\in\mathcal{P}}) = 0$.

注意 3.3.1 本方式 [5] の記述では，分散計算における消失は想定しておらず，マスターは全ワーカーから計算結果を受信するものとしている．しかし，本方式は，上記で述べた (b) 信頼性 (消失訂正能力) に関する性質を内在している．よって，この方式は (b)，(c) の両方を考慮していると解釈することができる．

命題 3.3.2（(c) に関する評価） 集合 $\mathcal{P} \subset [n]$ が与えられたとする．すべてのワーカー $i \in \mathcal{P}$ が結託して，自身のシェアから $\boldsymbol{A}$ の値を復元を復元しようとする．このとき，$|\mathcal{P}| \leq h - 1$ であれば，$\boldsymbol{A}$ の情報がまったく漏洩しない．すなわち，$\mathrm{I}(\boldsymbol{A}\,;\,(\tilde{\boldsymbol{A}}_{i\cdot})_{i\in\mathcal{P}}) = 0$.

3.3.3 本論文の新規性について

上記のような秘匿を考慮した分散方式の中で，第 5 章で提案するグループ型分散秘匿符号化計算方式はまだ提案されていない．この方式では，$\boldsymbol{A} \in \mathbb{F}_q^{k_\mathrm{A} \times l}, \boldsymbol{B} \in \mathbb{F}_q^{l \times k_\mathrm{B}}$ の積行列 $\boldsymbol{AB}$ の計算において，誤り/消失訂正能力，情報秘匿性，(a) システム全体の計算時間，(b) 信頼性 (消失訂正能力)，(c)$\boldsymbol{A}$ の値の情報秘匿性を考慮している．この方式では，秘密分散法の性質を用いて (b) と (c) を同時に満たすもとで，計算の並列化を行うことで (a) 計算時間を短縮している．さらに，第 5 章においては，上記の方式の一般化および上記 (a)(b)(c) の基準による性能評価を行い，提案された方式が優れた性質を有することを論じる．

第 4 章

計算時間と信頼性を考慮した分散符号化計算方式の構成法の提案

本章では，(a) 計算時間と (b) 信頼性を考慮した分散符号化計算方式の新たな構成法を提案する．従来方式は積行列 $\boldsymbol{AB}$ を列ごとに符号化する方式であったことを論じたが，本章では，新たに，ワーカーが試みる分散計算の結果 $\Pi(\boldsymbol{AB})$ が積行列 $\boldsymbol{AB}$ をまとめて符号化した行列である方式（**グループ型分散符号化計算方式**）の構成法を提案する．特に，具体的な構成法の一例として，$\boldsymbol{AB}$ を Gabidulin 符号化，限界ランク距離復号する方式（**グループ型 G 方式**）の構成法を提案する．この 2 つの方式では，複数のワーカーを均等にグループ分けして，各グループ内で行列積の並列計算を行うことで，システム全体の計算時間の短縮を図る．

本章の構成を説明する．最初に，提案の考え方の基礎として，従来の分散符号化計算方式と誤り訂正符号の原理的な対応関係を明確化する．具体的には，誤り訂正符号の符号化と，$\boldsymbol{AB}$ を $\Pi(\boldsymbol{AB})$ へ符号化することを対応させる．本論文で新たに構成法を提案する方式であるグループ型分散符号化計算方式，特にグループ型 G 方式は，$\Pi(\boldsymbol{AB})$ が積行列 $\boldsymbol{AB}$ 自体をまとめて符号化した行列となるような方式である．次に，グループ型分散符号化計算方式およびグループ型 G 方式の構成法を説明する．最後に，グループ型分散符号化計算方式の (b) 信頼性を評価し，さらに，グループ型 G 方式の (a) 計算時間と (b) 信頼性を評価する．特に，グループ型 G 方式の (a) の評価に関しては，グループ型 G 方式の計算時間が単独方式の計算時間より少なくなるパラメータの条件を導出する．また，グループ型 G 方式の (b) の評価に関しては，従来では訂正不可能であった誤り行列 $\boldsymbol{E}$ を訂正可能なことを示す．

本章では，q を任意の素数冪とし，正整数 $n, k_{\mathrm{A}}, k_{\mathrm{B}}, l$ を $2 \leq k_{\mathrm{A}} < n, 2 \leq k_{\mathrm{B}}, 2 \leq l$ を満たす任意の正整数とおく．また，本章では，$\max\{k_{\mathrm{B}}, n\}$ を m と表す．さらに，ベクトル $\boldsymbol{v} = (v_1, \ldots, v_m) \in \mathbb{F}_{q^m}^m$ を定義 2.1.11 に従って定義されるベクトル，

$f^1\colon \mathbb{F}_{q^m} \to \mathbb{F}_q^{1\times m}$ を定義 2.1.12 に従って定義される関数，$\boldsymbol{f}^n\colon \mathbb{F}_{q^m}^n \to \mathbb{F}_q^{n\times m}$ を定義 2.1.13 に従って定義される関数とする．本章ではこれらの値を任意に一つ固定する．

4.1 準備：誤り訂正符号と従来の分散符号化計算方式の対応

従来の分散符号化計算方式は誤り訂正符号を用いる方式であるが，両者の原理的な対応は，従来は明確にされていなかった．そこで，本論文で提案する方式の構成法を説明する前に，新たに両者の対応を明確にすることで，この方式の誤り訂正の原理を説明する．これが提案の考え方の基礎となる．

この方式は，積行列 $\boldsymbol{AB}$ の関数 $\Pi\colon \mathbb{F}_q^{k_\mathrm{A}\times k_\mathrm{B}} \to C(\subset \mathbb{F}_q^{n\times b})$ による符号化と，行列 $\boldsymbol{Y} = \Pi(\boldsymbol{AB}) + \boldsymbol{E}$ の関数 ψ による復号を原理的に行うことで，誤りのなす行列 $\boldsymbol{E} \coloneqq (\boldsymbol{e}_{1\cdot}, \ldots, \boldsymbol{e}_{n\cdot})^\top$ を訂正する．ただし，行列 $\Pi(\boldsymbol{AB})$ は，各 $i \in [n]$ 行目ベクトルが $\boldsymbol{g}_{i\cdot}^\top \boldsymbol{AB}$ である行列である．ここで，行列 $\boldsymbol{AB}$，行列 $\Pi(\boldsymbol{AB})$，行列 $\boldsymbol{E}$，行列 $\boldsymbol{Y}$，行列 $\boldsymbol{G}$，関数 Π，関数 ψ，集合 C を，それぞれ，誤り訂正符号における情報ベクトル，符号語，誤りベクトル，受信語，生成行列，符号化関数，復号関数，符号に対応させた．これらを，それぞれ，積行列，**符号語行列**，**誤り行列**，**受信語行列**，**生成行列**，**符号化関数**，**復号関数**，**符号語行列集合**と呼ぶ．

本論文で提案する方式は，分散計算の正しい結果 $\Pi(\boldsymbol{AB})$ 自体がある符号 $C(\subset \mathbb{F}_q^{n\times m})$, $m \coloneqq \max\{n, k_\mathrm{B}\}$ の符号語 $\boldsymbol{f}^n(\boldsymbol{GA}(\boldsymbol{B}, \boldsymbol{0}_{l\times(m-k_\mathrm{B})})\boldsymbol{v})$ であるような方式である．詳細な記号の説明は後述する．特に，$\Pi(\boldsymbol{AB})$ が $\boldsymbol{AB}$ を Gabidulin 符号化する方式では，従来方式では訂正不可能であった誤り行列 $\boldsymbol{E}$ を訂正可能である．ただし，計算時間を短縮するため，さらにワーカーの台数を増やし，各グループに分けた上で，分散符号化計算を行う．

4.2 グループ型分散符号化計算方式の提案

本節では，グループ型分散符号化計算方式を提案する．本方式の一例として，計算結果が積行列 $\boldsymbol{AB}$ を Gabidulin 符号化した行列である方式 (グループ型 G 方式) の構成法を提案する．

最初に，いくつかの記号を定義する．標準組織符号化の生成行列 $\boldsymbol{G} \in \mathbb{F}_{q^m}^{n\times k_\mathrm{A}}$ が存在するような $\mathbb{F}_{q^m}$ 上 (n, k_A) 線形符号を $\tilde{C} \subset \mathbb{F}_{q^m}^n$ と表す．$\mathbb{F}_q$ 上符号 $C \coloneqq \boldsymbol{f}^n(\tilde{C})(\subset \mathbb{F}_q^{n\times m})$ の復号関数を $\psi\colon \mathbb{F}_q^{n\times m} \to C \cup \{?\}$ と表す．また，行番号集合 $A \subset [n]$ に対し，$\boldsymbol{G}$ の全 $i \in A$ 行目 $\boldsymbol{g}_{i\cdot}^\top \in \mathbb{F}_q^{1\times k_\mathrm{A}}$ を並べた行列を $\boldsymbol{G}_{A\cdot}^\top \in \mathbb{F}_q^{|A|\times k_\mathrm{A}}$ とおく．訂正可能な誤

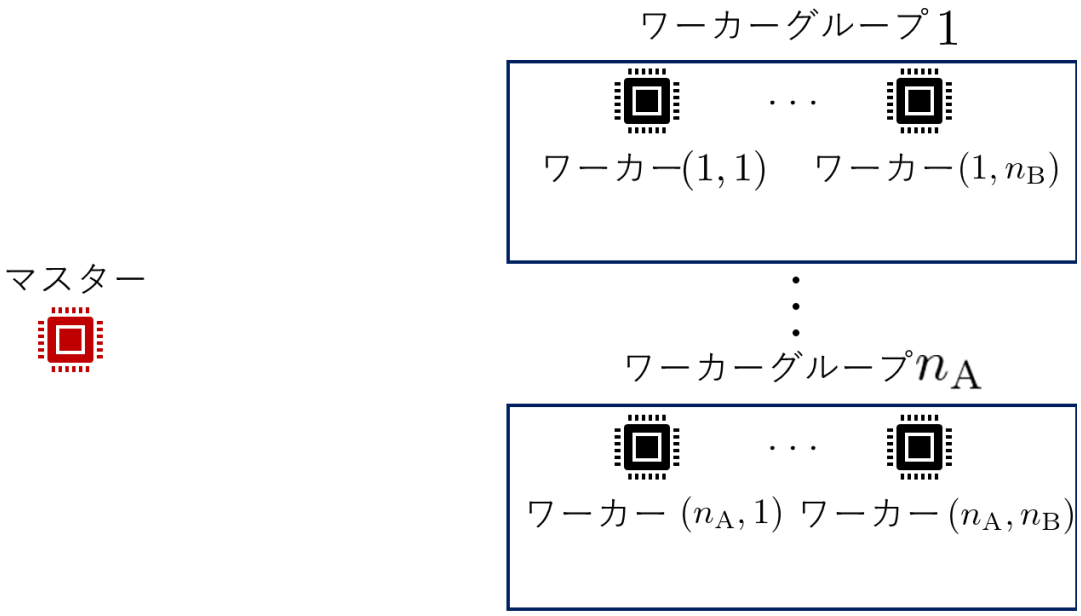

図 4.1　マスターとワーカーグループ

り行列全体の集合[*1]を $\mathcal{E} \subset \mathbb{F}_q^{n\times m}$ と表す．すなわち，任意の誤り行列 $\boldsymbol{E} \in \mathcal{E}$ と任意の符号語 $\boldsymbol{C} \in C$ に対し，$\psi(\boldsymbol{C}+\boldsymbol{E}) = \boldsymbol{C}$ が成立する．

補題 4.2.1　上記の符号 C の任意の符号語は，ある行列 $\boldsymbol{M} \in \mathbb{F}_q^{k_\mathrm{A}\times m}$ を用いて，$\boldsymbol{f}^n(\boldsymbol{GMv})$ と表される．

証明　$\mathbb{F}_{q^m}$ 上 (n, k_A) 線形符号 $\tilde{C} \subset \mathbb{F}_{q^m}^n$ の符号語は，ある $\boldsymbol{M} \in \mathbb{F}_q^{k_\mathrm{A}\times m}$ を用いて，$\boldsymbol{GMv}$ と表される．さらに，$C = \boldsymbol{f}^n(\tilde{C})$ である．□

$q, k_\mathrm{A}, k_\mathrm{B}, l, n, n_\mathrm{A}, n_\mathrm{B}$ が与えられた下での**グループ型分散符号化計算方式** $((\pi_{ij}|i \in [n_\mathrm{A}], j \in [n_\mathrm{B}]), \boldsymbol{G}, \psi)$ を説明する．本方式では，マスターとグループ毎に分かれた複数のワーカーを $n_\mathrm{A}n_\mathrm{B}$ 個利用する．ただし，n_A は n の正の約数，n_B は m の正の約数である．ワーカーは n_A 個のグループに均等に分かれており，各ワーカーグループにはワーカーが n_B 個ある．i 番目のグループの第 j 番目のワーカーは $(i, j) \in [n_\mathrm{A}] \times [n_\mathrm{B}]$ と番号付けられている（図 4.1）．本方式では，ワーカーの計算中における誤り発生のみを仮定している．マスターには，上記の生成行列 $\boldsymbol{G} \in \mathbb{F}_{q^m}^{n\times k_\mathrm{A}}$ と復号関数 ψ が保存されている．全ワーカー $(i, j) \in [n_\mathrm{A}] \times [n_\mathrm{B}]$ には後述の関数 π_{ij} が保存されている．

グループ型分散符号化計算方式では，マスターに行列 $\boldsymbol{A}$ が入力されると，ワーカーとマスターは**前処理**を行う．マスターに行列 $\boldsymbol{B}$ が入力されるごとに，ワーカーは**計算処理**を，マスターは**復号処理**を行い，積行列 $\boldsymbol{AB}$ の推定計算結果 $\hat{\boldsymbol{AB}} \in \mathbb{F}_q^{k_\mathrm{A}\times k_\mathrm{B}}$ を

[*1] 定義 2.4.3 で定義した誤りベクトル集合に相当する集合．

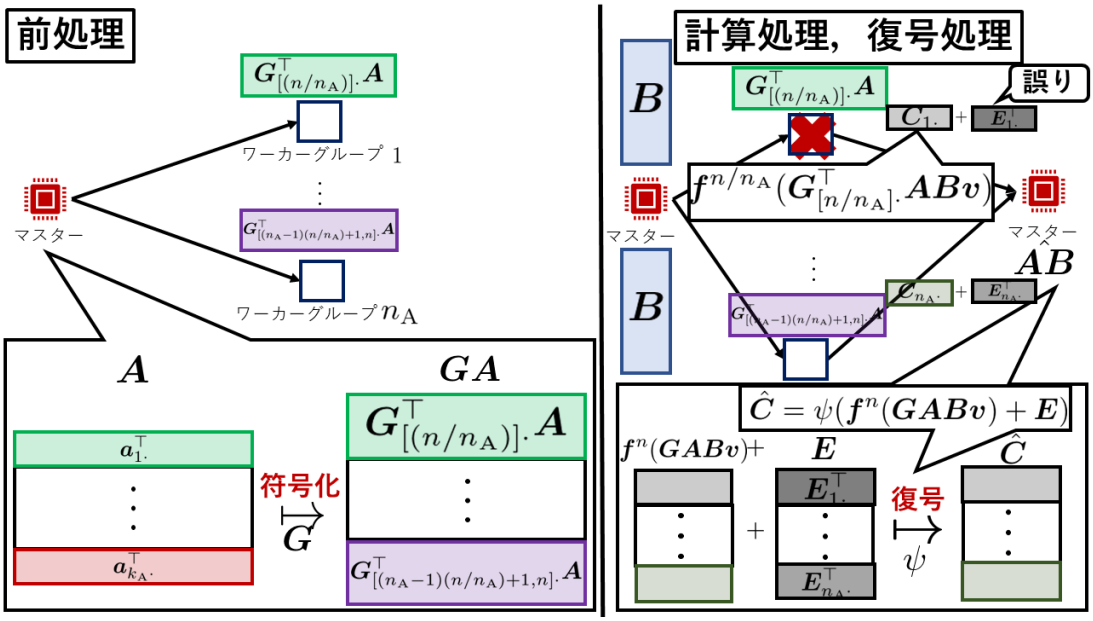

図 4.2　処理の流れ

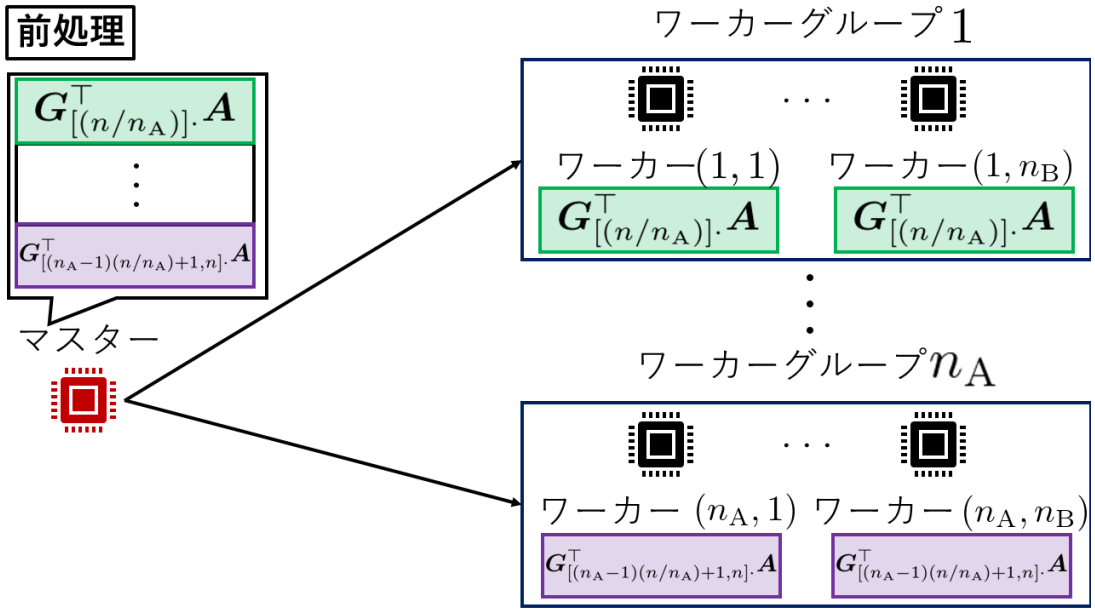

図 4.3　前処理

出力する．具体的には，次の手順を行う（図 4.2）．

〈 **前処理** 〉　(図 4.3) マスターは行列 $\boldsymbol{A}$ を符号化した行列

$$\boldsymbol{GA} = \begin{pmatrix} \boldsymbol{G}_{[1,n/n_{\mathrm{A}}]\cdot}^{\top}\boldsymbol{A} \\ \vdots \\ \boldsymbol{G}_{[(n_{\mathrm{A}}-1)(n/n_{\mathrm{A}})+1,n]\cdot}^{\top}\boldsymbol{A} \end{pmatrix} \in \mathbb{F}_{q^m}^{n\times l} \tag{4.1}$$

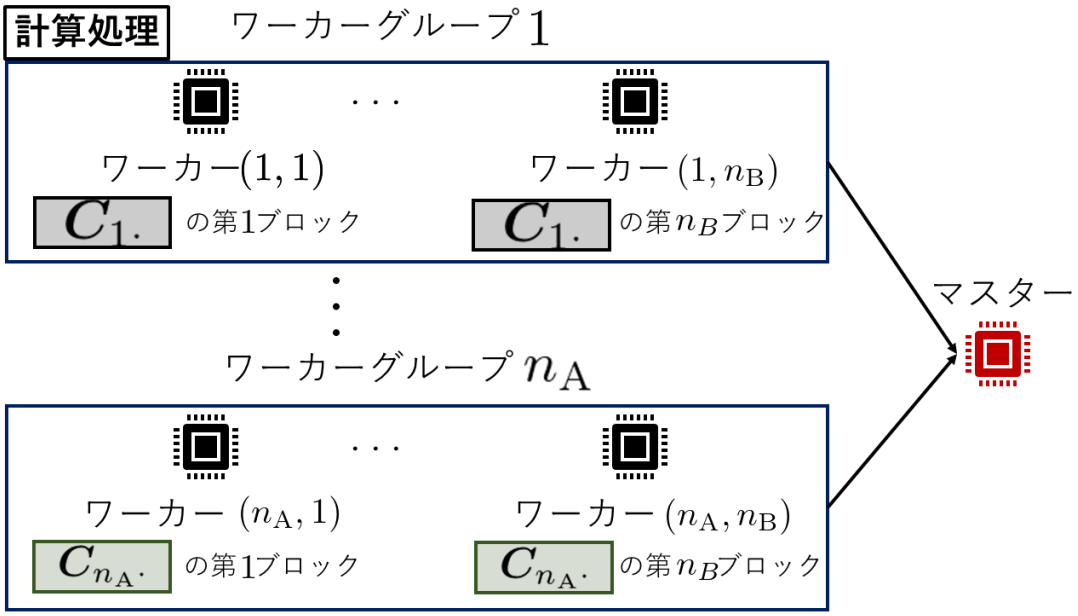

図 4.4　計算処理

の値を得て，各ワーカーグループ $i \in [n_A]$ の全ワーカーに

$$\boldsymbol{G}^{\top}_{[(i-1)(n/n_A)+1,i(n/n_A)]\cdot}\boldsymbol{A} \in \mathbb{F}_{q^m}^{(n/n_A)\times l} \tag{4.2}$$

を保存する．その後，各 $l' \in [l]$，$m' \in [m]$，$i' \in [(i-1)(n/n_A)+1, i(n/n_A)]$，$j' \in [(j-1)(m/n_B)+1, j(m/n_B)]$ に対し，$f^1(\boldsymbol{g}_{i'\cdot}^{\top}\boldsymbol{a}_{\cdot l'}v_{m'}) \in \mathbb{F}_q^{1\times m}$ の第 j' シンボルを計算する．また，これを $\tilde{a}_{i'l'm'j'} \in \mathbb{F}_q$ と表す．$\boldsymbol{a}_{\cdot l'} \in \mathbb{F}_q^{k_A}$ は，$\boldsymbol{A}$ の第 l' 列目である．$\tilde{a}_{i'l'm'j}$ は，$\boldsymbol{g}_{i'\cdot}^{\top}\boldsymbol{A} = (\boldsymbol{g}_{i'\cdot}^{\top}\boldsymbol{a}_{\cdot 1}, \ldots, \boldsymbol{g}_{i'\cdot}^{\top}\boldsymbol{a}_{\cdot l})$ から計算できる値である．

〈**計算処理**〉 (図 4.4) マスターは行列 $\boldsymbol{B}$ を全ワーカーに送信する．以下では，$n \le k_B$ のとき，$\boldsymbol{B}' = \boldsymbol{B}$ と定義し，$n > k_B$ のとき，$\boldsymbol{B}' = (\boldsymbol{B}, \boldsymbol{0})$ と定義する．$\boldsymbol{0}$ は $\mathbb{F}_q$ 上 $l \times (m - k_B)$ 零行列である．各ワーカー $(i, j) \in [n_A] \times [n_B]$ は，行列

$$\boldsymbol{C}_{i\cdot} := \boldsymbol{f}^{m/n_B}\left(\boldsymbol{G}_{[(i-1)(m/n_B)+1,i(m/n_B)]\cdot}\boldsymbol{A}\boldsymbol{B}'\boldsymbol{v}\right) \in \mathbb{F}_q^{(m/n_B)\times m} \tag{4.3}$$

の第 $(j-1)(m/n_B)+1, \ldots, j(m/n_B) \in [m]$ 列目（第 j ブロックと呼ぶ）を計算する．この計算は，各 $(i', j') \in [(i-1)(m/n_B)+1, i(m/n_B)] \times [(j-1)(m/n_B)+1, j(m/n_B)]$ において，$\boldsymbol{B}$ から $\sum_{l' \in [l]} \sum_{m' \in [m]} \tilde{a}_{i'l'm'j'} b_{l'm'}$ を計算することによって行われる．これについては命題 4.2.1 で詳述する．各ワーカー $(i, j) \in [n_A] \times [n_B]$ の正しい計算結果を $\pi_{ij}(\boldsymbol{g}_i^{\top}\boldsymbol{A}, \boldsymbol{B})$ とおく．本論文では，各 $(i, j) \in [n_A] \times [n_B]$ に対し，$\boldsymbol{G}^{\top}_{[(i-1)(n/n_A)+1,i(n/n_A)]\cdot}\boldsymbol{A} \in \mathbb{F}_{q^m}^{(n/n_A)\times l}$ と $\boldsymbol{B}$ から式 (4.3) の行列の第 $(j-1)(m/n_B)+1, \ldots, j(m/n_B) \in [m]$ 列目を計算する関

数 $\pi_{ij}\colon \mathbb{F}_{q^m}^{(n/n_{\mathrm{A}})\times l}\times\mathbb{F}_q^{l\times k_{\mathrm{B}}}\to\mathbb{F}_q$ を**積関数**と呼ぶ．ここで，$f^1\left(\boldsymbol{g}_{i'\cdot}^{\top}\boldsymbol{AB'v}\right)\in\mathbb{F}_q^{1\times m}$ の第 j' シンボルの計算中に誤り $e_{i'j'}\in\mathbb{F}_q$ を発生させたとする．マスターは，全ワーカーの出力結果を受信し，集約することで，

$$\boldsymbol{Y}\coloneqq\boldsymbol{f}^n(\boldsymbol{GAB'v})+\boldsymbol{E} \tag{4.4}$$

になるとする．ただし，$\boldsymbol{E}$ は任意の $(i',j')\in[n]\times[m]$ に対し，第 (i',j') 成分が $e_{i'j'}$ である行列である．

〈**復号処理**〉 マスターは，行列 $\boldsymbol{Y}$ から，関数 $\psi:\mathbb{F}_q^{n\times m}\to C\cup\{?\}$ を利用して，$\psi(\boldsymbol{Y})$ を得る．ただし，記号 $?\notin C$ は，推定計算結果 $\hat{\boldsymbol{AB}}$ を得られないことを表す．生成行列が標準組織符号化の生成行列であるため，$\psi(\boldsymbol{Y})\in C$ ならば，マスターは，行列 $\psi(\boldsymbol{Y})$ から $\hat{\boldsymbol{AB}}$ を直ちに得る．

命題 4.2.1　各 $(i',j')\in[(i-1)(m/n_{\mathrm{B}})+1,i(m/n_{\mathrm{B}})]\times[(j-1)(m/n_{\mathrm{B}})+1,j(m/n_{\mathrm{B}})]$ に対して，行列 $\boldsymbol{f}^n(\boldsymbol{GABv})\in\mathbb{F}_q^{n\times m}$ の第 (i',j') 成分は，$\sum_{l'\in[l]}\sum_{m'\in[k_{\mathrm{B}}]}\tilde{a}_{i'l'm'j'}b_{l'm'}$ に一致する．

証明　行列 $\boldsymbol{f}^n(\boldsymbol{GABv})\in\mathbb{F}_q^{n\times m}$ の第 (i',j') 成分は $f^1\left(\boldsymbol{g}_{i'\cdot}^{\top}\boldsymbol{AB'v}\right)$ の第 j' シンボルである．

$$f^1\left(\boldsymbol{g}_{i'\cdot}^{\top}\boldsymbol{AB'v}\right) \tag{4.5}$$

$$=f^1\left(\begin{pmatrix}\boldsymbol{g}_{i'\cdot}^{\top}\boldsymbol{a}_{\cdot 1} & \cdots & \boldsymbol{g}_{i'\cdot}^{\top}\boldsymbol{a}_{\cdot l}\end{pmatrix}\begin{pmatrix}\sum_{m'\in[m]}b'_{1m'}v_{m'}\\ \vdots\\ \sum_{m'\in[m]}b'_{lm'}v_{m'}\end{pmatrix}\right) \tag{4.6}$$

$$=f^1\left(\begin{pmatrix}\boldsymbol{g}_{i'\cdot}^{\top}\boldsymbol{a}_{\cdot 1} & \cdots & \boldsymbol{g}_{i'\cdot}^{\top}\boldsymbol{a}_{\cdot l}\end{pmatrix}\begin{pmatrix}\sum_{m'\in[k_{\mathrm{B}}]}b_{1m'}v_{m'}\\ \vdots\\ \sum_{m'\in[k_{\mathrm{B}}]}b_{lm'}v_{m'}\end{pmatrix}\right) \tag{4.7}$$

$$=f^1\left(\sum_{l'\in[l]}\boldsymbol{g}_{i'\cdot}^{\top}\boldsymbol{a}_{\cdot l'}\left(\sum_{m'\in[k_{\mathrm{B}}]}b_{l'm'}v_{m'}\right)\right) \tag{4.8}$$

$$=f^1\left(\sum_{l'\in[l]}\sum_{m'\in[k_{\mathrm{B}}]}b_{l'm'}\left(\boldsymbol{g}_{i'\cdot}^{\top}\boldsymbol{a}_{\cdot l'}\right)v_{m'}\right) \tag{4.9}$$

$$=f^1\left(\sum_{j\in[m]}\left(\sum_{l'\in[l]}\sum_{m'\in[k_{\mathrm{B}}]}\tilde{a}_{i'l'm'j}b_{l'm'}\right)v_j\right) \tag{4.10}$$

$$=\left(\sum_{l'\in[l]}\sum_{m'\in[k_{\mathrm{B}}]}\tilde{a}_{i'l'm'1}b_{l'm'},\ldots,\sum_{l'\in[l]}\sum_{m'\in[k_{\mathrm{B}}]}\tilde{a}_{i'l'm'm}b_{l'm'}\right). \tag{4.11}$$

これの第 j' シンボルは $\sum_{l'\in[l]}\sum_{m'\in[k_{\mathrm{B}}]}\tilde{a}_{i'l'm'j'}b_{l'm'}$ である．□

系 4.2.1　各ワーカー $(i,j) \in [n_{\mathrm{A}}] \times [n_{\mathrm{B}}]$ は，計算処理では，$\mathbb{F}_q$ 上加算を $(lk_{\mathrm{B}} - 1)(mn/n_{\mathrm{A}}n_{\mathrm{B}})$ 回，$\mathbb{F}_q$ 上乗算を $lk_{\mathrm{B}}(mn/n_{\mathrm{A}}n_{\mathrm{B}})$ 回行う．

証明　命題 4.2.1 から，ワーカー (i,j) の計算処理では，各 $(i',j') \in [(i-1)(m/n_{\mathrm{B}})+1 : i(m/n_{\mathrm{B}})] \times [(j-1)(m/n_{\mathrm{B}})+1, j(m/n_{\mathrm{B}})]$ に対し，$f^1(\boldsymbol{g}_{i'\cdot}^{\top}\boldsymbol{A}\boldsymbol{B}'\boldsymbol{v})$ の第 j' シンボル $\sum_{l' \in [l]} \sum_{m' \in [k_{\mathrm{B}}]} \tilde{a}_{i'l'm'j'} b_{l'm'}$ を計算していることが示された．また，各 $\tilde{a}_{i'l'm'j'}$ は，前処理の段階ですでに計算されている．よって，ワーカー (i,j) は，計算処理で，$\boldsymbol{B}$ を入力された下で $\sum_{l' \in [l]} \sum_{m' \in [k_{\mathrm{B}}]} \tilde{a}_{i'l'm'j'} b_{l'm'}$ を計算するには，$\mathbb{F}_q$ 上加算を $lk_{\mathrm{B}} - 1$ 回，$\mathbb{F}_q$ 上乗算を lk_{B} 回行えばよい．ワーカー (i,j) は，この計算をすべての $(i',j') \in [(i-1)(m/n_{\mathrm{B}})+1, i(m/n_{\mathrm{B}})] \times [(j-1)(m/n_{\mathrm{B}})+1, j(m/n_{\mathrm{B}})]$ に対して行う．□

上記のグループ型分散符号化計算方式の一例として，グループ型 G 方式を定義する．

定義 4.2.1（グループ型 G 方式）　$\boldsymbol{G} \in \mathbb{F}_{q^m}^{n \times k_{\mathrm{A}}}$ を $\mathbb{F}_{q^m}$ 上 (n, k_{A}) Gabidulin 符号の標準組織符号化の生成行列と定義する．ただし，この Gabidulin 符号の検査行列における集合 $\{h_1, \ldots, h_n\} \subset \mathbb{F}_{q^m}$(定義 2.4.14 参照) を $\mathbb{F}_q$ 上正規基底とする．復号関数 ψ を限界ランク距離復号関数（定義 2.4.16）と定義する．このとき，このグループ型分散符号化計算方式を，**グループ型 G 方式**と呼ぶ．

4.3　グループ型分散符号化計算方式の性能評価

本節では，グループ型分散符号化計算方式の (b) 信頼性（誤り訂正能力）と (a) 計算時間を評価する．(b) に関しては，グループ型分散符号化計算方式，さらにグループ型 G 方式を評価し，特に，グループ型 G 方式は従来の方式では訂正不可能だった誤り行列を訂正可能なことを述べる．グループ型分散符号化計算方式は，どの符号を選ぶかによって復号処理の時間が異なるため，一般的には評価が難しい．そこで，本論文では，(a) に関しては，グループ型 G 方式のみを評価する．さらに，単独方式の計算時間とグループ型 G 方式の計算時間の比較を行い，グループ型 G 方式が有利な条件を与える．

4.3.1　グループ型 G 方式の計算時間と信頼性の評価

(b) 信頼性（誤り訂正能力）の評価を行う．

定理 4.3.1（(b) に関する評価）　誤り行列 $\boldsymbol{E} \in \mathbb{F}_q^{n \times m}$ が与えられているとする．グ

ループ型分散符号化計算方式では，$\boldsymbol{E} \in \mathcal{E}$ ならば，正しく計算ができる．

証明　グループ型分散符号化計算方式において，行列 $\boldsymbol{A}$，$\boldsymbol{B}$ を入力すると，どのワーカーの計算でも誤りが発生しなかった場合に，行列 $\boldsymbol{f}^n(\boldsymbol{GAB'v}) \in \mathbb{F}_q^{n\times m}$ を出力する．補題 4.2.1 から，$\boldsymbol{f}^n(\boldsymbol{GAB'v}) \in \mathcal{C}$ であるので，$\boldsymbol{E} \in \mathcal{E}$ ならば，$\psi(\boldsymbol{f}^n(\boldsymbol{GAB'v}) + \boldsymbol{E}) = \boldsymbol{f}^n(\boldsymbol{GAB'v})$ である．よって，本方式では，$\mathcal{E}$ に属する誤り行列を訂正可能である．□

定理 4.3.2（(b) に関する評価）　誤り行列 $\boldsymbol{E}$ が与えられているとする．グループ型 G 方式では，$\mathrm{rank}\boldsymbol{E} \leq t(= \lfloor (n - k_\mathrm{A})/2 \rfloor)$ ならば，正しく計算ができる．

証明　定理 4.3.1 と，$\mathbb{F}_{q^m}$ 上 (n, k_A) Gabidulin 符号がランク t 以下の誤り行列 $\boldsymbol{E}$ を常に訂正できることから明らか．□

行列をまとめて符号化，復号するグループ型 G 方式により，従来方式では訂正できない誤り行列が訂正可能であることが示された．ただし，RS 方式で訂正不可能であり G 方式で訂正可能な誤り行列も，G 方式で訂正不可能であり RS 方式で訂正可能な誤り行列も両方存在する．$t = 1$ であれば，例えば，前者としては，全成分が同じ成分である行列や，特定の $t = 1$ 列の全成分の値が非零であるような行列が挙げられる．後者としては，左 n 列において対角線上に非零成分が並び，それ以外が零であるような行列があげられる．

次に，グループ型 G 方式の計算時間について評価する．本論文では，システムの総計算時間を，各ワーカーの並列計算処理の $\mathbb{F}_q$ 上四則演算回数と，マスターの復号処理の $\mathbb{F}_q$ 上四則演算回数の総和と定義する．なお，各方式では，$\boldsymbol{A}$ は一回だけ入力される一方，$\boldsymbol{B}$ の値は何回も入力されて都度 $\boldsymbol{AB}$ の計算を行うという仮定が置かれており，したがって，前処理の時間は計算時間として重要ではないため，前処理における $\mathbb{F}_q$ 上演算回数は総計算時間に含めていない．さらに，通信時間も同様に含めていない．ここで，拡大体 $\mathbb{F}_{q^m}$ 上の演算回数を数えるため，有限体として，次の条件を仮定する．

仮定 4.3.1（(a) 計算量評価のための仮定）　仮定として，q は 2 の冪乗としたもとで，$\log_2 q$ と m は互いに素，かつ $2m + 1$ は素数，かつ乗法群 $(\mathbb{Z}/(2m + 1)\mathbb{Z})^\star$ は 2 と -1 から生成されるとする[*2]．ただし，$(\mathbb{Z}/(2m + 1)\mathbb{Z})^\star$ を $\mathbb{Z}/(2m + 1)\mathbb{Z}$ から 0 の同値類を除いた集合と定義する．ベクトル $\boldsymbol{v} = (v_1, \ldots, v_m) \in \mathbb{F}_{q^m}^m$ を，$\mathbb{F}_q$ 上線形空間 $\mathbb{F}_{q^m}$ の最適な正規基底 $\{v_1, \ldots, v_m\}$ を並べたベクトルとする．

[*2] $q = 2$ の値のときは，[24] の Table 4.1 参照.

このとき，命題 2.1.15 で述べた通り，$\mathbb{F}_{q^m}$ 上の加算（または $\mathbb{F}_{q^m}$ 上の減算）は，m 回の $\mathbb{F}_q$ 上加算で行われる．また，$\mathbb{F}_{q^m}$ 上の乗算は，m^2-1 回の $\mathbb{F}_q$ 上加算と，m^2 回の $\mathbb{F}_q$ 上乗算によって行われる．さらに，$\mathbb{F}_{q^m}$ 上の逆元を求める演算は，$(m^2-1)(2\lfloor\log_2(m\log_2 q-1)\rfloor)$ 回の $\mathbb{F}_q$ 上加算と，$m^2(2\lfloor\log_2(m\log_2 q-1)\rfloor)$ 回の $\mathbb{F}_q$ 上乗算によって行われる．

$\mathrm{rank}\boldsymbol{E}\le t$，すなわち，正しい復号ができるとして，計算時間を評価する．ここで，系 2.4.1 における Gabidulin 符号の復号の一部に確率的アルゴリズムが使用しており，系 2.4.1 では $\mathbb{F}_q$ の期待値の上界を導出している．その上界式を用いて，本論文の方式の総計算時間を評価する．

定理 4.3.3（(a) に関する評価）　仮定 4.3.1 の下，各ワーカーの計算処理の $\mathbb{F}_q$ 上演算回数と，マスターの復号処理の $\mathbb{F}_q$ 上演算回数の総和は，高々

$$(2k_{\mathrm{B}}l-1)(mn/n_{\mathrm{A}}n_{\mathrm{B}})+D \tag{4.12}$$

回である．ただし，t を $\lfloor(n-k_{\mathrm{A}})/2\rfloor$，$d$ を $n-k_{\mathrm{A}}+1$，D を式 (2.32) で定義した値とする．

証明　ワーカー $(i,j)\in[n_{\mathrm{A}}]\times[n_{\mathrm{B}}]$ の計算処理の $\mathbb{F}_q$ 上の演算回数は，$(2k_{\mathrm{B}}l-1)(mn/n_{\mathrm{A}}n_{\mathrm{B}})$ 回である (系 4.2.1).

マスターの復号処理では，限界ランク距離復号により，受信した計算結果 $\boldsymbol{f}^n(\boldsymbol{GAB'v})+\boldsymbol{E}\in\mathbb{F}_q^{n\times m}$ から正しい計算結果 $\boldsymbol{f}^n(\boldsymbol{GAB'v})\in\mathbb{F}_q^{n\times m}$ を計算する．$\mathbb{F}_q$ 上の演算回数は，系 2.4.1 から，D 回であると示される．また，$\boldsymbol{G}$ が標準組織符号化の生成行列であることから，$\boldsymbol{f}^n(\boldsymbol{GAB'v})$ から $\boldsymbol{AB}$ を取り出す時間は計算時間の評価に入れなくてよい．□

注意 4.3.1　計算時間のオーダー評価から考えるに，[39] の復号アルゴリズムを用いると更なる高速化が期待される．

系 4.3.1（単独方式との (a) に関する比較）　仮定 4.3.1 の下，$n,n_{\mathrm{A}},n_{\mathrm{B}}$ が以下を満たすときに，提案方式の総計算時間の上界の値 (式 (4.12)) は，単独方式の $\mathbb{F}_q$ 上演算回数より小さい値になる．

$$l>\frac{1}{2k_{\mathrm{B}}(k_{\mathrm{A}}-(mn/n_{\mathrm{A}}n_{\mathrm{B}}))}(k_{\mathrm{A}}k_{\mathrm{B}}-(mn/n_{\mathrm{A}}n_{\mathrm{B}})+D) \tag{4.13}$$

例 4.3.1

$$(q,k_{\mathrm{A}},k_{\mathrm{B}},l)=(2,100,293,100000) \tag{4.14}$$

かつ $n=n_{\mathrm{A}},m=n_{\mathrm{B}}$ のとき，$n(>k_{\mathrm{A}}=100)$ が 101 以上 172 以下のときに，提案方

式が有利である．$(q, m) = (2, 293)$ が仮定 4.3.1 を満たしていることは，[24] の Table 4.1 から確かめられる．

4.3.2 Erasure 復号を用いた場合の性能評価

次に，Erasure 復号を考えることとする．このとき，(b) 信頼性は次のようになる．

命題 4.3.1（(b) に関する評価） 誤り行列 $\boldsymbol{E}$ と row error $\boldsymbol{E}_{\text{row}}$ が与えられているとする．グループ型 G 方式で Erasure 復号を行う場合，$\text{rank}(\boldsymbol{E}) \leq t$ ならば，正しく計算ができる．

Erasure 復号を用いた場合のシステムの総計算時間を評価する．$\text{rank}\boldsymbol{E} \leq t$，すなわち，正しい復号ができるとして，計算時間を評価する．

定理 4.3.4（(a) に関する評価） 仮定 4.3.1 の下，システムの総計算時間，すなわち $\mathbb{F}_q$ 上演算回数は，高々 $2k_{\text{B}}l - 1 + (2n-1)(n-k_{\text{A}}) + 4mnt - mn - nt + \frac{3}{2}mt(t-1) + (2m^2-1)(2mt + \frac{3}{2}t^2 - \frac{5}{2}t - 1)$ 回である．ただし，$t = \lfloor (n-k_{\text{A}})/2 \rfloor$．

証明 ワーカー $(i,k) \in [n] \times [m]$ の計算処理の $\mathbb{F}_q$ 上の演算回数は，$(2k_{\text{B}}l - 1)(mn/n_{\text{A}}n_{\text{B}})$ 回である．これは通常の復号を考える時と同様である．

マスターの復号処理では, Erasure 復号により, 受信した計算結果 $\boldsymbol{f}^n(\boldsymbol{GAB'v}) + \boldsymbol{E} \in \mathbb{F}_q^{n \times m}$ から $\boldsymbol{f}^n(\boldsymbol{GAB'v})$ を計算する．$\mathbb{F}_q$ 上の演算回数は $(2n-1)(n-k_{\text{A}}) + 4mnt - mn - nt + \frac{3}{2}mt(t-1) + (2m^2-1)(2mt\log_2 q + \frac{3}{2}t^2 - \frac{5}{2}t - 1)$ 回である（系 2.4.2）．また，$\boldsymbol{f}^n(\boldsymbol{GAB'v})$ から $\boldsymbol{AB}$ を取り出すのは，$\boldsymbol{G}$ が標準組織符号化の生成行列であることから，計算時間は評価に入れなくてよい．□

系 4.3.2（単独方式との (a) に関する比較） 仮定 4.3.1 の下，単独方式の $\mathbb{F}_q$ 上演算回数と比較すると，n が以下を満たすときに提案方式の計算時間が少ない．

$$
\begin{aligned}
l > {} & \frac{1}{2k_{\text{B}}(k_{\text{A}} - (mn/n_{\text{A}}n_{\text{B}}))}(k_{\text{A}}k_{\text{B}} - (mn/n_{\text{A}}n_{\text{B}}) \\
& + (2n-1)(n-k_{\text{A}}) + 4mnt - mn - nt + \frac{3}{2}mt(t-1) \\
& + (2m^2-1)(t(2\lfloor \log(m\log_2 q - 1) \rfloor + 2) + \frac{3}{2}t^2 + \frac{1}{2}t - 1))
\end{aligned} \tag{4.15}
$$

例 4.3.2

$$
(q, k_{\text{A}}, k_{\text{B}}, l) = (2, 100, 293, 100000) \tag{4.16}
$$

かつ $n = n_{\text{A}}, m = n_{\text{B}}$ のとき，のとき，$n (> k_{\text{A}} = 100)$ が 101 以上 342 以下のときに，提案方式が有利である．

4.3.3 そのほかの性能評価

復号誤り率による誤り訂正能力の特性の確認

(b) 信頼性の性能評価に関連して，定理 4.3.2, 命題 3.2.1 においてすでに理論的に示した両方式の誤り訂正能力の特性を，誤り行列 $\boldsymbol{E}$ の確率的発生モデルを仮定した下での復号誤り率の数値例により確認する．列ごとに誤り訂正するか，誤り行列自体を訂正するかという点で両方式は異なるが，補足として，各列のハミング重みが小さい誤り行列と低ランク誤り行列の両方を表現可能なモデルの一つを仮定し，前者が発生しやすいほど RS 方式が，後者が発生しやすいほどグループ型 G 方式が有効に働くというすでに示した特性を数値例で確認する．

誤り行列 $\boldsymbol{E}$ の発生モデルを説明する[*3]．最初に，行列 $\boldsymbol{E}$ のどの列が非零列になりうるかが独立に確率 $\delta(< 0.5)$ で選ばれる．選ばれなかった列は，確率 1 で全成分が 0 となる．次に，選ばれた各列では，どの位置の成分の値が $\mathbb{F}_q \setminus \{0\}$ の元になるかが独立に確率 $\epsilon(< 0.5)$ で選ばれる．選ばれなかった位置の成分の値は，確率 1 で 0 となる．最後に，選ばれた位置の成分の値は，$\mathbb{F}_q \setminus \{0\}$ 上一様分布に従って選ばれる．このモデルでは，δ の値が小さく ϵ の値が大きいほど，少ない列に非零成分が集中した誤り行列，したがって低ランク誤り行列が発生しやすくなる．

RS 方式とグループ型 G 方式の復号誤り率 $1 - \Pr\{\psi(\Pi(\boldsymbol{AB}) + \boldsymbol{E}) = \Pi(\boldsymbol{AB})\}$ を評価する．命題 3.2.1 と定理 4.3.2 から，RS 方式とグループ型 G 方式の復号誤り率はそれぞれ $1 - \Pr\left\{ \forall j \in [k_{\mathrm{B}}], \mathrm{w}(\boldsymbol{e}_{\cdot,j}) \leq t \right\}$ と $1 - \Pr\{ \mathrm{rank}(\boldsymbol{E}) \leq t \}$ であると示される．RS 方式の復号誤り率を導出する．

命題 4.3.2　RS 方式の復号誤り率は次式で与えられる．

$$1 - \left((1-\delta) + \delta \sum_{r=0}^{t} \mathrm{bin}(n, r, \epsilon) \right)^{k_{\mathrm{B}}} . \tag{4.17}$$

ただし，非負整数 n, w と実数 ϵ に対し，$\mathrm{bin}(n, w, \epsilon)$ を ${}_n\mathrm{C}_w \epsilon^w (1-\epsilon)^{n-w}$ と定義する．${}_n\mathrm{C}_w$ は二項係数である．

グループ型 G 方式の復号誤り率は正確に導出できないため，上界 $1 - \Pr\{ | \{ j \in [m] \mid \boldsymbol{e}_{\cdot j} \neq \boldsymbol{0} \} | \leq t \}$ を導出する．

[*3] 例えば，全プロセッサの出力に同時に誤りが発生しうる環境において，マルチプロセッサを搭載したコンピュータで計算する状況が考えられる．マスターとワーカーはプロセッサを表し，δ は同時に誤りが発生する確率を表し，ϵ はその中で各ワーカーの出力が誤る確率を表す．これは，例えばサージ電圧が発生する環境が考えられる．

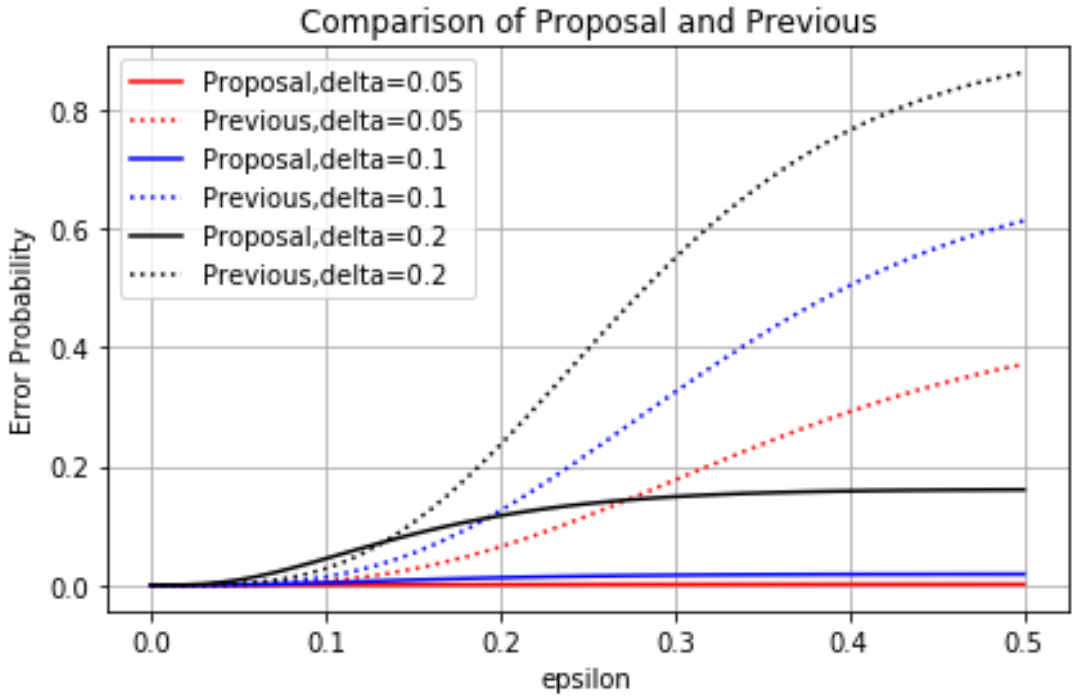

図 4.5　復号誤り率 ($q = 11, n = 10, k_{\mathrm{B}} = 11, k_{\mathrm{A}} = 3$)

命題 4.3.3　グループ型 G 方式の復号誤り率の上界は次式で与えられる．

$$1 - \sum_{c=0}^{m} \mathrm{bin}(m, c, \delta) \sum_{c'=0}^{\min\{t,c\}} \mathrm{bin}(c, c', 1 - (1-\epsilon)^n).$$

図 4.5，図 4.6 では，$q, n, k_{\mathrm{A}}, k_{\mathrm{B}}$ の値を固定し，δ を様々な値に動かすことで，横軸が ϵ，縦軸が復号誤り率であるグラフを複数描いている．グラフから，δ の値が小さく ϵ の値が大きいほど，誤り行列の特定の列に非零成分の個数が多く，非零成分を有する列の個数が少ないので，グループ型 G 方式が有効に働くことが確認される．

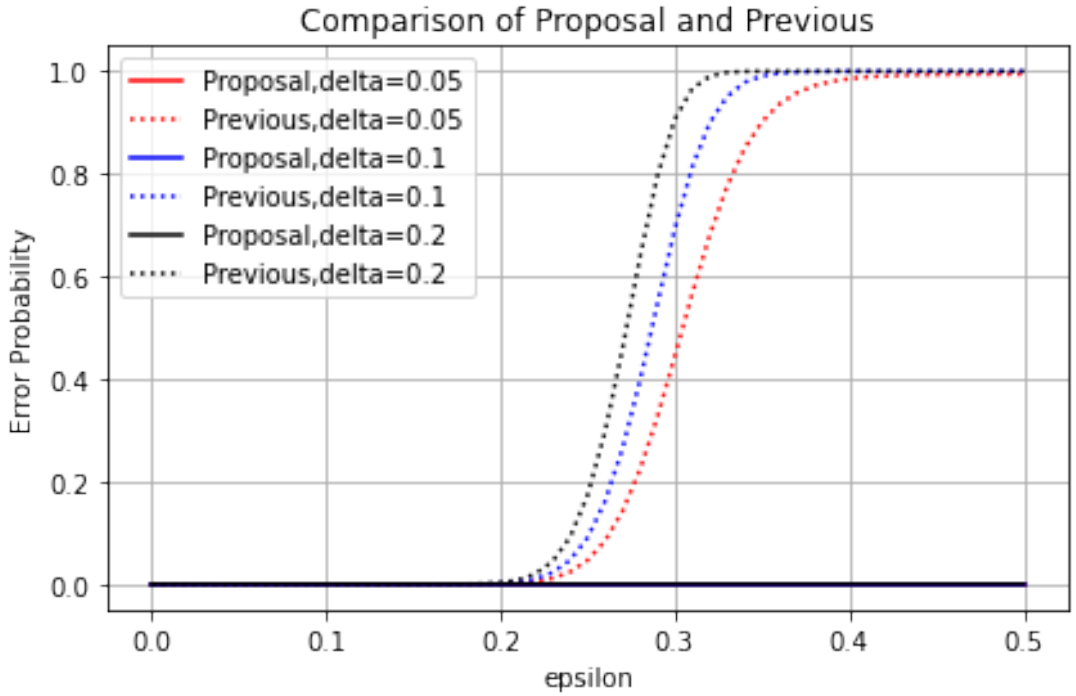

図 4.6　復号誤り率 ($q = 101, n = 100, k_{\mathrm{B}} = 101, k_{\mathrm{A}} = 30$)

第 5 章

計算時間と信頼性と情報秘匿性を考慮した新たな分散符号化計算方式

本章では，(a) 計算時間と (b) 信頼性に加え，(c) 情報秘匿性，すなわち，行列 $\boldsymbol{A}$ の値を各ワーカーに秘匿する機能も有する新たな分散符号化計算方式として，**グループ型分散秘匿符号化計算方式**を提案する．より詳細には，ワーカーが行列 $\boldsymbol{A}$ の値を知ろうとしても，ワーカーに渡された情報を一部のワーカーが結託して集約しただけでは $\boldsymbol{A}$ の値を復元不可能である，という秘匿性を保持しつつ，積行列 $\boldsymbol{AB}$ を分散計算する方式を論じる．ただし，本論文では行列 $\boldsymbol{B}$ の値は各ワーカーに公開してもよいとする．3 章で説明した通り，従来研究として，(b) 信頼性と (c) 情報秘匿性を有する分散計算方式は行われていたが，分散計算により (1) システム全体の計算時間を短くできる優位性も含めては論じられていなかった．本章では，(a)，(b)，(c) を全て考慮した**グループ型分散秘匿符号化計算方式**を提案し，その性能評価を行う．本章で提案する方式も，また，前章と同様に，$n_{\mathrm{A}}n_{\mathrm{B}}$ 台のワーカーを n_{A} 個のワーカーグループに分け，一つのワーカーグループに n_{B} 個のワーカーが属するようにする（図 4.1）．今回は，$\boldsymbol{B}$ をグループ内で分割して持たせることにより，計算時間の短縮を図る．さらに，$\boldsymbol{A}$ をランダム符号化した行列の部分行列を同グループ内に共通で持たせることで，グループ内での結託に対する $\boldsymbol{A}$ の値の秘匿も図る．

本章の構成は以下の通りである．最初に，秘密分散法を利用したグループ型分散秘匿符号化計算方式の構成法，特に基本方式を提案する．次に，(a) 計算時間，(b) 信頼性，(c) 情報秘匿性の評価を行う．(a) に関しては，前章と同様に，システム全体での総計算時間を算出し，単独方式のそれと比較して有利であるようなパラメータの値の条件を示す．(b) に関しては，マスターは一定個数以上のワーカーグループのシェア

から $\boldsymbol{AB}$ の値を復元できることを示す．(c) に関しては，$\boldsymbol{A}$ の値の秘匿に関して，一定個数以下のワーカーグループが結託しても $\boldsymbol{A}$ の値に関する情報が全く漏洩しないことを示す．最後に，これを一般化した方式である改良方式の構成法を提案し，同様に性能評価を行い，信頼性と情報秘匿性において優れた性質を有することを示す．

5.1 グループ型分散秘匿符号化計算方式の基本方式の提案

本節では，$(h_{\mathrm{A}}, n_{\mathrm{A}})$ しきい値秘密分散方式を用いて構成するグループ型分散秘匿符号化計算方式の**基本方式**を提案する．ただし，$q > n_{\mathrm{A}} (> h_{\mathrm{A}})$ を仮定する．

マスターには行列 $\boldsymbol{G}_{\mathrm{A},1}^{\top}, \ldots, \boldsymbol{G}_{\mathrm{A},n_{\mathrm{A}}}^{\top} \in \mathbb{F}_q^{k_{\mathrm{A}} \times h_{\mathrm{A}} k_{\mathrm{A}}}$ が保存されている．ただし，$\boldsymbol{G}_{\mathrm{A},i\cdot}$ を $(\boldsymbol{I}_{k_{\mathrm{A}}}, \alpha^i \boldsymbol{I}_{k_{\mathrm{B}}}, \ldots, \alpha^{i(h_{\mathrm{A}}-1)} \boldsymbol{I}_{k_{\mathrm{B}}})^{\top}$ と定義する．このとき，次式が成立する．

$$\begin{pmatrix} \boldsymbol{G}_{\mathrm{A},1\cdot}^{\top} \\ \vdots \\ \boldsymbol{G}_{\mathrm{A},n_{\mathrm{A}}\cdot}^{\top} \end{pmatrix} = \mathbf{RS}_{n_{\mathrm{A}} \times h_{\mathrm{A}}} \otimes \boldsymbol{I}_{k_{\mathrm{A}}} \tag{5.1}$$

$$= \begin{pmatrix} \boldsymbol{I}_{k_{\mathrm{A}}} & \alpha \boldsymbol{I}_{k_{\mathrm{A}}} & \ldots & \alpha^{h_{\mathrm{A}}-1} \boldsymbol{I}_{k_{\mathrm{A}}} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{I}_{k_{\mathrm{A}}} & \alpha^{n_{\mathrm{A}}} \boldsymbol{I}_{k_{\mathrm{A}}} & \ldots & \alpha^{n_{\mathrm{A}}(h_{\mathrm{A}}-1)} \boldsymbol{I}_{k_{\mathrm{A}}} \end{pmatrix}. \tag{5.2}$$

$\mathbf{RS}_{n_{\mathrm{A}} \times h_{\mathrm{A}}}$ は，式 (2.27) で定義される $\mathbb{F}_q$ 上 $(n_{\mathrm{A}}, h_{\mathrm{A}})$ Reed Solomon 符号の生成行列である．式 (5.1) の行列を $\boldsymbol{G}_{\mathrm{A}} \in \mathbb{F}_q^{n_{\mathrm{A}} k_{\mathrm{A}} \times h_{\mathrm{A}} k_{\mathrm{A}}}$ と定義する．

次の手順で $\boldsymbol{AB}$ を計算する[*1]．

〈 **前処理** 〉 (図 5.1)

マスターは行列 $\boldsymbol{A}$ が入力されると，マスターは乱数行列 $\boldsymbol{R}_1, \ldots, \boldsymbol{R}_{h-1}$ を取り，行列 $\tilde{\boldsymbol{A}}_{1\cdot}, \ldots, \tilde{\boldsymbol{A}}_{n_{\mathrm{A}}\cdot}$ を得る．ここで，各 $i \in [n_{\mathrm{A}}]$ に対して，次式が成立する．

$$\tilde{\boldsymbol{A}}_{i\cdot} = \boldsymbol{G}_{\mathrm{A},i\cdot}^{\top} \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{R} \end{pmatrix} \tag{5.3}$$

$$= \boldsymbol{A} + \alpha^i \boldsymbol{R}_1 + \cdots + \alpha^{i(h_{\mathrm{A}}-1)} \boldsymbol{R}_{h_{\mathrm{A}}-1}. \tag{5.4}$$

ただし，乱数行列 $\boldsymbol{R} \in \mathbb{F}_q^{(h_{\mathrm{A}}-1)k_{\mathrm{A}} \times l}$ を，$\boldsymbol{R}_1, \ldots, \boldsymbol{R}_{h_{\mathrm{A}}-1} \in \mathbb{F}_q^{k_{\mathrm{A}} \times l}$ を以下のように

[*1] 本方式は，3.3.2 節で説明した分散秘匿符号化計算方式 [5] と類似した方法である．例えば，前処理においては，符号化は同じである．ただし，各グループ i の全ワーカーに $\tilde{\boldsymbol{A}}_{i\cdot}$ を配布するか，各ワーカー i に $\tilde{\boldsymbol{A}}_{i\cdot}$ を配布するかという点に差異がある．

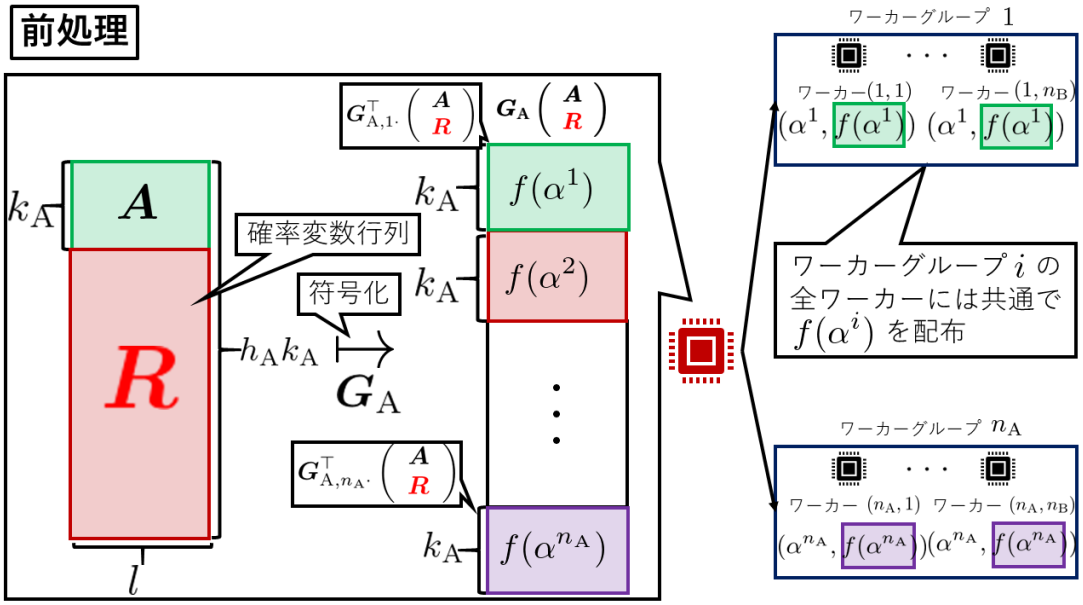

図 5.1　基本方式（前処理）

並べた行列と定義する．

$$\boldsymbol{R} = \begin{pmatrix} \boldsymbol{R}_1 \\ \vdots \\ \boldsymbol{R}_{h_A-1} \end{pmatrix}. \tag{5.5}$$

マスターは，式 (5.4) を，Algorithm 1 に基づいて計算する．ただし，各 $(u,v) \in [k_A] \times [l]$ に対し，$\boldsymbol{A}$ の第 (u,v) 成分を a_{uv} と表し，行列 $\boldsymbol{R}_h$，$h \in [h_A - 1]$ の第 (u,v) 成分を r_{huv} と表す．

マスターはワーカーグループ i に $(\alpha^i, \tilde{\boldsymbol{A}}_{i\cdot})$ を配布し，ワーカーグループ i 内の全ワーカーは共通でこれを受信する．

〈 **計算処理** 〉 (図 5.2)

マスターに行列 $\boldsymbol{B}$ が入力されるごとに，マスターは行列 $\boldsymbol{B}$ を n_B 個の $l \times (k_B/n_B)$ 行列に等分割する．$j \in [n_B]$ 番目の行列を $\boldsymbol{B}_{\cdot j}$ と表す．マスターは各ワーカーグループに $\boldsymbol{B} = (\boldsymbol{B}_{\cdot 1}, \ldots, \boldsymbol{B}_{\cdot n_B})$ を送信．各ワーカー $(i,j) \in [n_A] \times [n_B]$ に $\boldsymbol{B}_{\cdot j}$ が配布される．各ワーカー (i,j) は，行列

$$\boldsymbol{G}^\top_{A,i\cdot} \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{R} \end{pmatrix} \tag{5.6}$$

と $\boldsymbol{B}_{\cdot j}$ から，積行列

$$\boldsymbol{G}^\top_{A,i\cdot} \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{R} \end{pmatrix} \boldsymbol{B}_{\cdot j} \in \mathbb{F}_q^{c_{A,i} \times c_{B,j}} \tag{5.7}$$

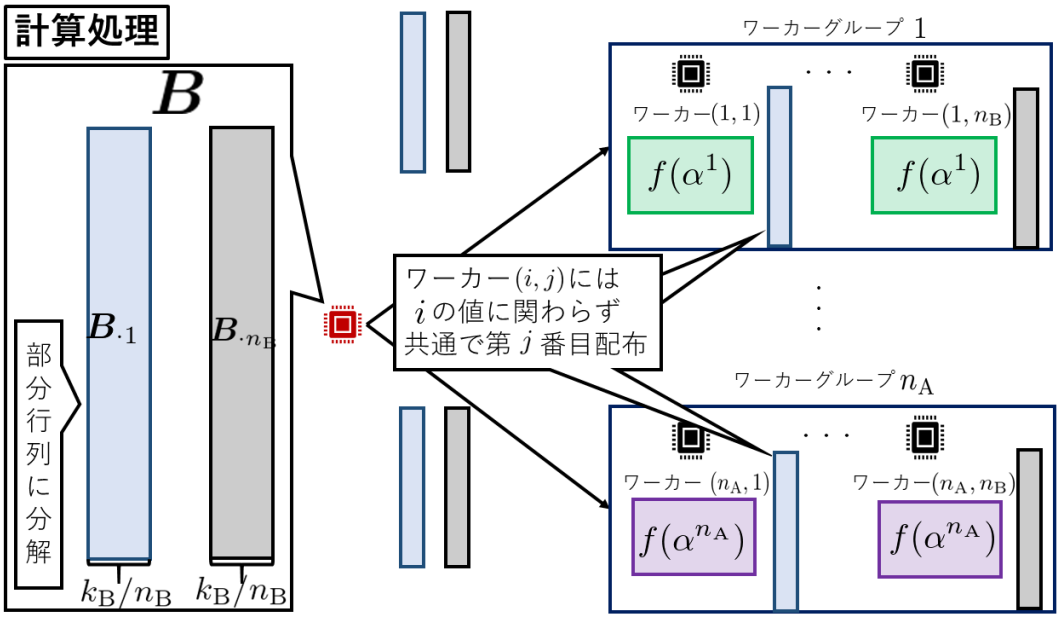

図 5.2　基本方式（計算処理）

を計算.

各ワーカーグループ i は，それを集約して

$$\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{R}\end{pmatrix}\boldsymbol{B}=\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix} \tag{5.8}$$

を得て，

$$(\alpha^{i},\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}) \tag{5.9}$$

をワーカーに送信.

ここで，マスターは h 個のワーカーグループから出力結果を受信するが，残りのワーカーグループからは受信できなかったとする[*2]．受信したワーカーグループのインデックス全体の集合 $\{i_1,\ldots,i_h\}$ とおく.

〈 **復号処理** 〉 (図 5.3[*3])

$h \le h_A - 1$ であれば，マスターは計算できなかったことを出力する.

$h \ge h_A$ であれば，マスターは $\boldsymbol{G}_{\mathrm{A},i_1\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix},\ldots,\boldsymbol{G}_{\mathrm{A},i_h\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}$ から Algorithm 2 により復号し，正しく $\boldsymbol{AB}$ の値を得る.

[*2] 例えば一定時間経過後に受信できたか否かを判断する

[*3] 図中のバツ印は，図中のワーカーグループの一部のワーカーの故障により，マスターがそのワーカーグループから結果を受信できなかったことを表す.

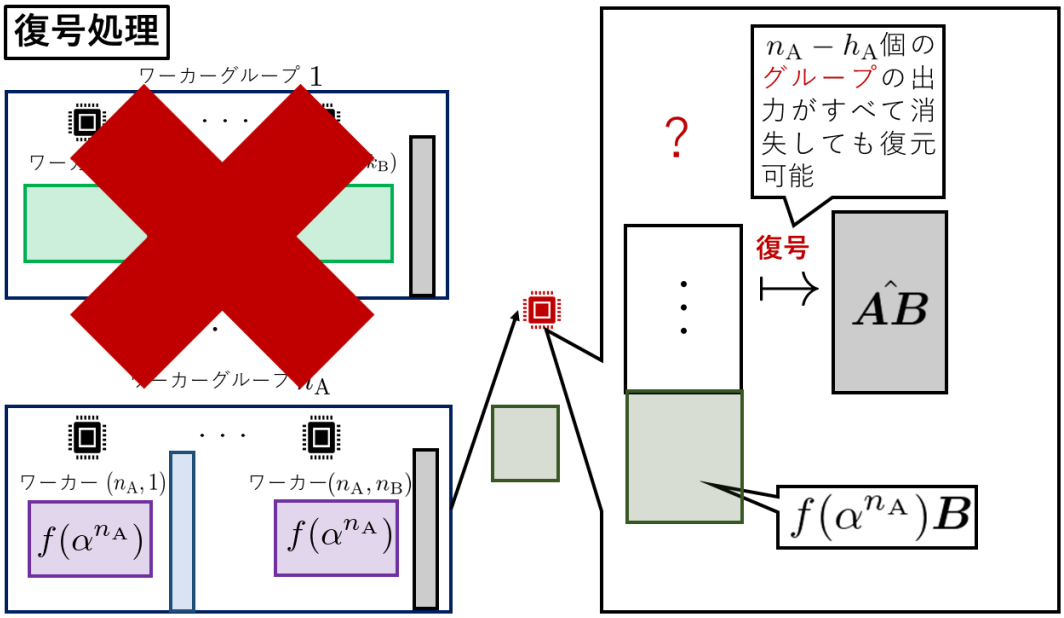

図 5.3　基本方式（復号処理）

Algorithm 1 前処理アルゴリズム

Require: $\boldsymbol{A} \in \mathbb{F}_q^{k_A \times l}$

Ensure: ワーカーグループ $i \in [n_A]$ に配布するシェア $\tilde{\boldsymbol{A}}_{i\cdot} = \boldsymbol{G}_{A,i\cdot}^{\top} \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{R} \end{pmatrix}$

1: **for** $(u, v) \in [k_A] \times [l]$ **do**
2: 　$\tilde{a}_{iuv} \coloneqq a_{uv} + \alpha^i r_{1uv} + \cdots + \alpha^{i(h_A-1)} r_{h_A-1,uv}$ を計算
3: **end for**
4: $\tilde{a}_{iuv}$ が第 (u, v) 成分の行列を $\tilde{\boldsymbol{A}}_{i\cdot}$ とおく．

5.2　基本方式の性能評価

基本方式の (a) 計算時間，(b) 信頼性，(c) 情報秘匿性の性能評価を行う．また，計算時間に関しては，単独方式の計算時間と比較し，基本方式が有利となるパラメータ n_B, h_A の値の条件を述べる．

この方式における $\boldsymbol{A}$ の値の (c) 情報秘匿性の定式化を定理 5.2.1 で述べる．ここで，$\boldsymbol{A}' \coloneqq \boldsymbol{AB}$，$\boldsymbol{R}_i' \coloneqq \boldsymbol{R}_i\boldsymbol{B}$ とおく．また，各 $(u, v) \in [k_A] \times [k_B]$ に対し，$\boldsymbol{A}'$ の第 (u, v) 成分を a'_{uv} とおき，$\boldsymbol{R}_i'$ の第 (u, v) 成分を r'_{iuv} とおく．変数 x の多項式 $f_{uv}(x) \coloneqq a'_{uv} + r'_{1uv}x + \cdots + r'_{h_A-1,uv}x^{h_A-1}$ を変数 x の多項式とする．

定理 5.2.1（基本方式の (c) に関する評価）　$\boldsymbol{A}, \boldsymbol{R}_1, \ldots, \boldsymbol{R}_{h-1}$ を，すべて独立に $\mathbb{F}_q^{k_A \times l}$

Algorithm 2 復号アルゴリズム

Require: $(\alpha^{i_1}, \boldsymbol{G}_{\mathrm{A},i_1\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}),\ldots,(\alpha^{i_{h_\mathrm{A}}}, \boldsymbol{G}_{\mathrm{A},i_{h_\mathrm{A}}\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix})$

Ensure: $\boldsymbol{A}' = \boldsymbol{AB}$

1: $\prod_{h'=1}^{h_\mathrm{A}} \alpha^{i_{h'}}$ を計算
2: **for** $h \in [h_\mathrm{A}]$ **do**
3: $\quad \alpha^{i_h} \prod_{h' \in [h_\mathrm{A}]\setminus\{h\}} (\alpha^{i_{h'}} - \alpha^{i_h})$ を計算
4: $\quad \frac{\prod_{h'=1}^{h_\mathrm{A}} \alpha^{i_{h'}}}{\alpha^{i_h} \prod_{h' \in [h_\mathrm{A}]\setminus\{h\}} (\alpha^{i_{h'}} - \alpha^{i_h})}$ を計算
5: **end for**
6: **for** $(u, v) \in [k_\mathrm{A}] \times [k_\mathrm{B}]$ **do**
7: $\quad a'_{uv} = 0$
8: $\quad$ **for** $h \in [h_\mathrm{A}]$ **do**
9: $\quad\quad a'_{uv} \leftarrow a'_{uv} + f_{uv}(\alpha^{i_h}) \frac{\prod_{h'=1}^{h_\mathrm{A}} \alpha^{i_{h'}}}{\alpha^{i_h} \prod_{h' \in [h_\mathrm{A}]\setminus\{h\}} (\alpha^{i_{h'}} - \alpha^{i_h})}$ を計算
10: $\quad$ **end for**
11: **end for**
12: 第 (u, v) 成分が a'_{uv} である行列を $\boldsymbol{A}'$ とおく．

上一様分布に従う確率変数行列とする．$h_\mathrm{A} - 1$ 個以下の任意のワーカーグループが結託してできた集合 $\mathcal{P} \subset [n_\mathrm{A}], |\mathcal{P}| \leq h_\mathrm{A} - 1$ に対し，$\mathcal{P}$ に属するワーカーグループのシェア $\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{R}\end{pmatrix}$, $i \in \mathcal{P}$ をすべて集めても，$\boldsymbol{A}$ の値に関する情報は全く漏洩しない，すなわち，

$$\mathrm{I}\left(\boldsymbol{A};\ (\tilde{\boldsymbol{A}}_{i\cdot})_{i\in\mathcal{P}}\right) = \mathrm{I}\left(\boldsymbol{A};\ \left(\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{R}\end{pmatrix}\right)_{i\in\mathcal{P}}\right) = 0. \tag{5.10}$$

証明　$\mathcal{P} \subset [n], |\mathcal{P}| \leq h_\mathrm{A} - 1$ を任意にとる．$\boldsymbol{G}_{\mathrm{A},i\cdot}$ の定義から，

$$\mathrm{I}\left(\boldsymbol{A};\ \left(\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{R}\end{pmatrix}\right)_{i\in\mathcal{P}}\right) \tag{5.11}$$

$$= \mathrm{I}\left(\boldsymbol{A};\ \left(\boldsymbol{A} + \alpha^{i-1}\boldsymbol{R}_1 + \cdots + \alpha^{(i-1)(h_\mathrm{A}-1)}\boldsymbol{R}_{h_\mathrm{A}-1}\right)_{i\in\mathcal{P}}\right) \tag{5.12}$$

ここで，相互情報量のチェイン則から，

$$\mathrm{I}\left(\boldsymbol{A};\ \left(\boldsymbol{A} + \alpha^{i-1}\boldsymbol{R}_1 + \cdots + \alpha^{(i-1)(h_\mathrm{A}-1)}\boldsymbol{R}_{h_\mathrm{A}-1}\right)_{i\in\mathcal{P}}\right) \tag{5.13}$$

$$\leq \sum_{(u,v)\in[k_\mathrm{A}]\times[l]} \mathrm{I}\left(a_{uv};\ \left(a_{uv} + \alpha^{i-1}r_{1uv} + \cdots + \alpha^{(i-1)(h_\mathrm{A}-1)}r_{h_\mathrm{A}-1,uv}\right)_{i\in\mathcal{P}}\right) \tag{5.14}$$

したがって，各 $(u,v) \in [k_{\mathrm{A}}] \times [l]$ に対して，

$$\mathrm{I}\left(a_{uv}\,;\ \left((a_{uv} + \alpha^{i-1} r_{1uv} + \cdots + \alpha^{(i-1)(h_{\mathrm{A}}-1)} r_{h_{\mathrm{A}}-1,uv})\right)_{i\in\mathcal{P}}\right) = 0. \tag{5.15}$$

が成立することを示せばよい．$a_{uv}, r_{1uv}, \ldots, r_{h_{\mathrm{A}}-1,uv}$ がすべて独立であり，すべて $\mathbb{F}_q$ 上一様分布に従う確率変数であることに着目すると，この式は，従来の $(h_{\mathrm{A}}, n_{\mathrm{A}})$ 閾値秘密分散法の結果から明らかである．□

この方式における $\boldsymbol{AB}$ の値の (b) 信頼性（消失訂正能力）の定式化を定理 5.2.2 で述べる．

定理 5.2.2（基本方式の (b) に関する評価） h_{A} 個以上のワーカーグループからなる任意の集合 $\mathcal{P} \subset [n_{\mathrm{A}}], |\mathcal{P}| \geq h_{\mathrm{A}}$ に対して，次式が成立する．

$$\mathrm{H}\left(\boldsymbol{AB} \mid \left(\tilde{\boldsymbol{A}}_{i\cdot}\boldsymbol{B}\right)_{i\in\mathcal{P}}\right) = \mathrm{H}\left(\boldsymbol{AB} \mid \left(\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}\right)_{i\in\mathcal{P}}\right) = 0. \tag{5.16}$$

式 (5.16) は，ある関数 $\psi\colon \left(\mathbb{F}_q^{k_{\mathrm{A}}\times k_{\mathrm{B}}}\right)^{h_{\mathrm{A}}} \to \mathbb{F}_q^{k_{\mathrm{A}}\times k_{\mathrm{B}}}$ が存在し，その出力

$$\left(\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}\right)_{i\in\mathcal{P}} \tag{5.17}$$

から $\boldsymbol{AB}$ を ψ を用いて確率 1 で復元することができる，すなわち

$$\Pr\left(\psi\left(\left(\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{AB}\\ \boldsymbol{RB}\end{pmatrix}\right)_{i\in\mathcal{P}}\right) = \boldsymbol{AB}\right) = 1 \tag{5.18}$$

であることを意味する．この関数 ψ は，$\mathcal{P}_{\mathrm{A}}$ に依存する関数である．

証明 マスターは互いに相異なるワーカーグループ $i_1, \ldots, i_{h_{\mathrm{A}}}$ から出力を受信する $(\mathcal{P} \supset \{i_1, \ldots, i_{h_{\mathrm{A}}}\})$ として，以下の復号アルゴリズムで復号が可能なことを示せば，定理が示される．ここで，次のようにノーテーションを定める．

- $\boldsymbol{A}' \coloneqq \boldsymbol{AB}$ とおき，その第 $(u,v) \in [k_{\mathrm{A}}] \times [k_{\mathrm{B}}]$ 成分を a'_{uv} とおく．
- $\boldsymbol{R}'_i \coloneqq \boldsymbol{R}_i\boldsymbol{B}$ とおき，その第 (u,v) 成分を r'_{iuv} とおく．
- 各 (u,v) に対し，$f_{uv}(x) \coloneqq a'_{uv} + r'_{1uv}x + \cdots + r'_{h_{\mathrm{A}}-1,uv}x^{h_{\mathrm{A}}-1}$ を変数 x の多項式とする．

このとき，第 (u,v) 成分が $f_{uv}(\alpha^i)$ である $k_{\mathrm{A}} \times k_{\mathrm{B}}$ 行列は，以下のようにして，ワーカーグループ i が出力した結果を集約した行列 $\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{R}\end{pmatrix}\boldsymbol{B}$ であることが示される．

$$(a'_{uv} + r'_{1uv}\alpha^{i-1} + \cdots + r'_{h_{\mathrm{A}}-1,uv}\alpha^{(i-1)(h_{\mathrm{A}}-1)})_{(u,v)\in[k_{\mathrm{A}}]\times[k_{\mathrm{B}}]} \tag{5.19}$$

$$= \boldsymbol{A}' + \boldsymbol{R}'_1 \alpha^{i-1} + \cdots + \boldsymbol{R}'_{h_\mathrm{A}-1} \alpha^{(i-1)(h_\mathrm{A}-1)} \tag{5.20}$$

$$= \boldsymbol{G}_{\mathrm{A},i\cdot}^\top \begin{pmatrix} \boldsymbol{A}' \\ \boldsymbol{R}' \end{pmatrix} \tag{5.21}$$

$$= \boldsymbol{G}_{\mathrm{A},i\cdot}^\top \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix} \tag{5.22}$$

$$= \boldsymbol{G}_{\mathrm{A},i\cdot}^\top \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{R} \end{pmatrix} \boldsymbol{B}. \tag{5.23}$$

また，各 $(u,v) \in [k_\mathrm{A}] \times [k_\mathrm{B}]$ に対して，$f_{uv}(\alpha^{i_1}), \ldots, f_{uv}(\alpha^{i_{h_\mathrm{A}-1}})$ の値が既知であれば，式 (5.24) によって，$\boldsymbol{A}' = \boldsymbol{AB}$ の第 (u,v) 成分 $a'_{uv} = f_{uv}(0)$ の値を計算できる．ラグランジュ多項式補間を利用することで，式 (5.24) の通り計算できる．この式は，ラグランジュ多項式補間から導出される式である．

$$f_{uv}(0) = \sum_{h \in [h_\mathrm{A}]} f_{uv}(\alpha^{i_h}) \frac{\prod_{h'=1}^{h_\mathrm{A}} \alpha^{i_{h'}}}{\alpha^{i_h} \prod_{h' \in [h_\mathrm{A}] \setminus \{h\}} (\alpha^{i_{h'}} - \alpha^{i_h})}. \tag{5.24}$$

以上より，第 (u,v) 成分が a'_{uv}，すなわち式 (5.24) である行列 $\boldsymbol{A}' = \boldsymbol{AB}$ を Algorithm 2 によって計算することができることが示された．□

基本方式の (a) 計算時間を評価する．

定理 5.2.3（基本方式の (a) に関する評価）　本方式の計算時間を，ワーカーの計算処理，マスターの復号処理における $\mathbb{F}_q$ 上四則演算の回数の総和と定めると，その上界は，以下で与えられる．

$$\frac{k_\mathrm{A} k_\mathrm{B} (2l-1)}{n_\mathrm{B}} + k_\mathrm{A} k_\mathrm{B} (2h_\mathrm{A} - 1) + 2h_\mathrm{A}^2 - 1. \tag{5.25}$$

証明　ワーカー $(i,j) \in [n_\mathrm{A}] \times [n_\mathrm{B}]$ の計算処理の $\mathbb{F}_q$ 上の演算回数のうち，加算が $\frac{k_\mathrm{A} k_\mathrm{B} (l-1)}{n_\mathrm{B}}$ 回，乗算が $\frac{k_\mathrm{A} k_\mathrm{B} l}{n_\mathrm{B}}$ 回である．実際，$\tilde{\boldsymbol{A}}_{i\cdot} \in \mathbb{F}_q^{k_\mathrm{A} \times l}$ と $\tilde{\boldsymbol{B}}_{\cdot j} \in \mathbb{F}_q^{l \times \frac{k_\mathrm{B}}{n_\mathrm{B}}}$ の積行列 $\tilde{\boldsymbol{A}}_{i\cdot} \tilde{\boldsymbol{B}}_{\cdot j} \in \mathbb{F}_q^{k_\mathrm{A} \times \frac{k_\mathrm{B}}{n_\mathrm{B}}}$ を計算するには，l 次元ベクトル同士の内積を $k_\mathrm{A}(k_\mathrm{B}/n_{,B})$ 回行えばよいことと，l 次元ベクトル同士の内積の加算が $l-1$ 回，乗算が l 回であることから，上述の回数が算出できる．

マスターの復号処理は，Algorithm 2 によって行われるので，この処理における $\mathbb{F}_q$ 上の演算回数は，加算が $k_\mathrm{A} k_\mathrm{B} h_\mathrm{A}$ 回，減算が $h_\mathrm{A}(h_\mathrm{A}-1)$ 回，乗算が $h_\mathrm{A}^2 + k_\mathrm{A} k_\mathrm{B} h_\mathrm{A}$ 回，逆元を求める演算が h_A 回である．

したがって，$\mathbb{F}_q$ 上の演算回数は，加算が $\frac{k_\mathrm{A} k_\mathrm{B} (l-1)}{n_\mathrm{B}} + k_\mathrm{A} k_\mathrm{B} h_\mathrm{A}$ 回，減算が $h_\mathrm{A}(h_\mathrm{A}-1)$ 回，乗算が $\frac{k_\mathrm{A} k_\mathrm{B} l}{n_\mathrm{B}} + h_\mathrm{A}^2 + k_\mathrm{A} k_\mathrm{B} h_\mathrm{A}$ 回，逆元を求める演算が h_A 回である．

加算と減算をそれぞれ同じ 1 回と数えるときは，加算，減算が $\frac{k_\mathrm{A} k_\mathrm{B} (l-1)}{n_\mathrm{B}} + k_\mathrm{A} k_\mathrm{B} h_\mathrm{A} + h_\mathrm{A}(h_\mathrm{A}-1)$ 回，乗算が $\frac{k_\mathrm{A} k_\mathrm{B} l}{n_\mathrm{B}} + h_\mathrm{A}^2 + k_\mathrm{A} k_\mathrm{B} h_\mathrm{A}$ 回，逆元を求める演算が h_A 回である．ま

た，$\mathbb{F}_q$ 上の演算回数がすべて等しいときは，$\frac{k_A k_B(2l-1)}{n_B} + 2k_A k_B h_A + 2h_A^2 - 1$ である．
□

系 5.2.1（基本方式の (a) における単独方式との比較） 単独方式と比較すると，n_B が十分大きく，h_A が十分小さいとき，基本方式の計算時間が少ない．具体的には，

$$(k_B \geq) n_B > \frac{2l-1}{2l - 2h_A - \frac{2h_A^2-1}{k_A k_B}} \tag{5.26}$$

かつ $1 \leq h_A < \frac{-k_A k_B + \sqrt{k_A^2 k_B^2 + 4k_A k_B l + 2}}{2}$ を満たすときに，基本方式が有利である．

$1 \leq h_A < \frac{-k_A k_B + \sqrt{k_A^2 k_B^2 + 4k_A k_B l + 2}}{2}$ は，式 (5.26) の分母が正である条件を表す．

例 5.2.1

$$(k_A, k_B, l) = (100, 293, 100000) \tag{5.27}$$

のとき，h_A が 41426 以下のときに分母が正になる．例えば h_A が 100 のとき，$n_B (\leq k_B = 293)$ が 2 以上 293 以下のときに，提案方式が有利である．

ここで，簡単のため，$n_A = n_B$ とする．$k_A = k_B$ とすると，$h_A \leq n_A = n_B \leq k_B = k_A$ なので，$\frac{h_A^2 + h_A - 1}{k_A k_B}$ や $\frac{2h_A - 1}{k_A k_B}$ は 2 以下の正実数である．よって l は $h_A + 2$ 以上であれば式 (5.26) の右辺がより大きくなる．ここで，$\frac{h_A^2 + h_A - 1}{k_A k_B}$ や $\frac{h_A^2 - h_A}{k_A k_B}$ は 2 以下の正実数となるので，$k_A = k_B$ である場合は，l が $h_A + 2$ 以上であり，かつ n_B は $\frac{l}{l - h_A}$ 以上であるときは，基本方式の方が優れている．他にも，$k_A = 1$ である場合は，l が $2h_A + 1$ 以上であり，かつ n_B は $\frac{l}{l - h_A}$ 以上であるときは，基本方式の方が優れている．

5.3 改良方式の提案

基本方式と同様の方式だが，計算処理において $\boldsymbol{B}$ をただ分割する代わりに，$\boldsymbol{B}$ を分割して (n_B, h_B) Reed Solomon 符号化に対応する符号化を行うグループ型分散秘匿符号化計算方式を構成する．ただし，正整数 h_B は n_B 以下の整数であり，k_B の約数とする．本論文ではこの方式を**改良方式**と呼ぶ．これにより，計算時間は増えるが，信頼性の向上が図れる．すなわち，マスターが結果を受信するワーカーグループ内で，$n_B - h_B$ 個のワーカーの計算結果の消失が発生しても，$\boldsymbol{AB}$ の値の復元が可能である．本方式では，$q > n_A (> h_A)$ および $q > n_B (> h_B)$ を仮定する．

マスターには，行列 $\boldsymbol{G}_{B,1\cdot}, \ldots, \boldsymbol{G}_{B,n_B\cdot} \in \mathbb{F}_q^{k_B \times (k_B/h_B)}$ が保存されている．ただし，$\boldsymbol{G}_{B,j\cdot}$ を $(\boldsymbol{I}_{k_B/h_B}, \alpha^j \boldsymbol{I}_{k_B/h_B}, \ldots, \alpha^{j(h_B-1)} \boldsymbol{I}_{k_B/h_B})^\top$ と定義する．ここで，次式が成立

する．

$$\begin{pmatrix} \boldsymbol{G}_{\mathrm{B},1\cdot}^{\top} \\ \vdots \\ \boldsymbol{G}_{\mathrm{B},n_{\mathrm{B}}\cdot}^{\top} \end{pmatrix} = \mathbf{RS}_{n_{\mathrm{B}}\times h_{\mathrm{B}}} \otimes \boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} \tag{5.28}$$

$$= \begin{pmatrix} \boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} & \alpha\boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} & \dots & \alpha^{h_{\mathrm{B}}-1}\boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} & \alpha^{n_{\mathrm{B}}}\boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} & \dots & \alpha^{n_{\mathrm{B}}(h_{\mathrm{B}}-1)}\boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} \end{pmatrix} \tag{5.29}$$

この行列を $\boldsymbol{G}_{\mathrm{B}}(\in \mathbb{F}_q^{n_{\mathrm{B}}(k_{\mathrm{B}}/h_{\mathrm{B}})\times k_{\mathrm{B}}})$ と表す．ここで，

$$\boldsymbol{B}\boldsymbol{G}_{\mathrm{B},j\cdot} = (\boldsymbol{B}_{\cdot 1}, \boldsymbol{B}_{\cdot 2}, \dots, \boldsymbol{B}_{\cdot h_{\mathrm{B}}}) \begin{pmatrix} \boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} \\ \alpha^{j}\boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} \\ \vdots \\ \vdots \\ \alpha^{(h_{\mathrm{B}}-1)j}\boldsymbol{I}_{k_{\mathrm{B}}/h_{\mathrm{B}}} \end{pmatrix} \tag{5.30}$$

$$= \boldsymbol{B}_{\cdot 1} + \alpha^{j}\boldsymbol{B}_{\cdot 2} + \cdots + \alpha^{j(n_{\mathrm{B}}-1)}\boldsymbol{B}_{\cdot h_{\mathrm{B}}} \tag{5.31}$$

が成立する．ただし，$\boldsymbol{B}_{\cdot j}$ は行列 $\boldsymbol{B}$ を列方向に h_{B} 等分割した時の $j \in [n_{\mathrm{B}}]$ 番目の部分行列である．

次の手順で $\boldsymbol{AB}$ を計算する．

〈 **前処理** 〉 マスターは行列 $\boldsymbol{A}$ が入力されると，基本方式の前処理と同じ処理を施す．

〈 **符号化処理** 〉 マスターは $\boldsymbol{B}$ を入力されると，$\boldsymbol{B}$ を全ワーカーに $\boldsymbol{B}$ を送信する．

各ワーカー (i, j) は $\boldsymbol{B}$ を入力されると，

$$\tilde{\boldsymbol{A}}_{i\cdot}\boldsymbol{B}\boldsymbol{G}_{\mathrm{B},j\cdot} = \boldsymbol{G}_{\mathrm{A},i\cdot}^{\top} \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix} \boldsymbol{G}_{\mathrm{B},j\cdot} \tag{5.32}$$

を計算する．

〈 **復号処理** 1-i 〉 各グループ $i \in [n_{\mathrm{A}}]$ 毎に次の処理を行う．正しく値を受信したワーカーの出力を集約することで，

$$\left(\alpha^{i}, \boldsymbol{G}_{\mathrm{A},i\cdot}^{\top} \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix} \boldsymbol{G}_{\mathrm{B},j\cdot}\right)_{j\in\mathcal{P}_{\mathrm{B},i}} \tag{5.33}$$

が得られる．$\mathcal{P}_{\mathrm{B},i} \subset [n_{\mathrm{B}}]$ はグループ i において正しく値を受信したワーカー全体の集合．アルゴリズムは定理 5.4.1 の証明を参照．これを正しく復号すれば，$\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top}\boldsymbol{AB}$ を得る．

〈**復号処理 2**〉 復号に成功したグループの出力を集約することで,

$$\left(\alpha^i, \boldsymbol{G}_{\mathrm{A},i\cdot}^{\top} \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix}\right)_{i \in \mathcal{P}_{\mathrm{A}}} \tag{5.34}$$

が得られる．$\mathcal{P}_{\mathrm{A}} \subset [n_{\mathrm{A}}]$ は正しく復号したグループ全体の集合である．復号アルゴリズムは Algorithm 2 と同様である．これを正しく復号すれば，$\boldsymbol{AB}$ を得る．

5.4 改良方式の性能評価

5.2 節と同様に，この方式の (a) 計算時間，(b) 信頼性（消失訂正能力），(c) 情報秘匿性の性能評価を行う．情報秘匿性については，定理 5.2.1 での定式化と同様であるため，消失訂正能力，計算時間を説明する．

この方式における $\boldsymbol{AB}$ の値の信頼性（消失訂正能力）の定式化を定理 5.4.1 で述べる．

定理 5.4.1（(b) に関する評価） h_{A} 個以上のワーカーグループからなる任意の集合 $\mathcal{P}_{\mathrm{A}} \subset [n_{\mathrm{A}}], |\mathcal{P}_{\mathrm{A}}| \geq h_{\mathrm{A}}$ および，各ワーカーグループ $i \in \mathcal{P}_{\mathrm{A}}$ における h_{B} 個以上のワーカーからなる任意の集合 $\mathcal{P}_{\mathrm{B},i} \subset [n_{\mathrm{B}}], |\mathcal{P}_{\mathrm{B},i}| \geq h_{\mathrm{B}}$ に対して，次式が成立する．

$$\mathrm{H}\left(\boldsymbol{AB} \,\middle|\, \left(\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top} \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix} \boldsymbol{G}_{\mathrm{B},j\cdot}\right)_{i \in \mathcal{P}_{\mathrm{A}}, j \in \mathcal{P}_{\mathrm{B},i}}\right) = 0. \tag{5.35}$$

証明 任意のワーカーグループ $i \in \mathcal{P}_{\mathrm{A}}$ に対して，復号処理 $1-i$ において，マスターが，ワーカー $j_1, \ldots, j_{h_{\mathrm{B}}}$ から計算結果を受信するとする．ただし，$j_1, \ldots, j_{h_{\mathrm{B}}} \in \mathcal{P}_{\mathrm{B},i}$ の値は i に依存する．マスターは，各 $i \in \mathcal{P}_{\mathrm{A}}$ に対して，ワーカー $j_1, \ldots, j_{h_{\mathrm{B}}}$ から受信した計算結果を集約した行列 $\left(\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top} \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix} \boldsymbol{G}_{\mathrm{B},j\cdot}\right)_{j \in \mathcal{P}_{\mathrm{B},i}}$ の値から，$\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top} \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix}$ の値をあるアルゴリズムによって計算することができる，ということを示せば，この定理は証明される．実際，$\left(\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top} \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix}\right)_{i \in \mathcal{P}_{\mathrm{A}}}$ から $\boldsymbol{AB}$ を復元可能なことは，定理 5.2.2 の証明と同様に証明できる．

以下の復号アルゴリズムで復号が可能なことを示せば，定理が示される．ここで，次のようにノーテーションを定める．$\tilde{\boldsymbol{A}}'_i \coloneqq \boldsymbol{G}_{\mathrm{A},i\cdot}^{\top} \begin{pmatrix} \boldsymbol{AB} \\ \boldsymbol{RB} \end{pmatrix} \in \mathbb{F}_q^{k_{\mathrm{A}} \times k_{\mathrm{B}}}$．その第 $(u, v) \in [k_{\mathrm{A}}] \times [k_{\mathrm{B}}]$ 成分を $\tilde{a}'_{iuv}$ と定義する．さらに，$\tilde{\boldsymbol{A}}'_i$ の第 $u \in [k_{\mathrm{A}}]$ 行目の転置ベクトル $(\tilde{a}'_{iu1}, \ldots, \tilde{a}'_{iuk_{\mathrm{B}}})^{\top} \in \mathbb{F}_q^{k_{\mathrm{B}}}$ を $\tilde{\boldsymbol{a}}'_{iu}$ と定義する．このとき，各 $u \in [k_{\mathrm{B}}]$ に対

して,

$$\left((\tilde{\boldsymbol{a}}'_{iu})^\top \begin{pmatrix} \boldsymbol{G}_{\mathrm{B},j_1\cdot} & \dots & \boldsymbol{G}_{\mathrm{B},j_{h_\mathrm{B}-1}\cdot} \end{pmatrix}\right)^\top \tag{5.36}$$

$$= \begin{pmatrix} \boldsymbol{G}^\top_{\mathrm{B},j_1\cdot} \\ \vdots \\ \boldsymbol{G}^\top_{\mathrm{B},j_{h_\mathrm{B}-1}\cdot} \end{pmatrix} \tilde{\boldsymbol{a}}'_{iu} \tag{5.37}$$

$$= \begin{pmatrix} \alpha^{0\cdot j_1}\begin{pmatrix} \tilde{a}'_{i,u,1} \\ \vdots \\ \tilde{a}'_{i,u,k_\mathrm{B}/h_\mathrm{B}} \end{pmatrix} + \dots + \alpha^{(h_\mathrm{B}-1)j_1}\begin{pmatrix} \tilde{a}'_{i,u,(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})+1} \\ \vdots \\ \tilde{a}'_{i,u,k_\mathrm{B}} \end{pmatrix} \\ \vdots \\ \alpha^{0\cdot j_{h_\mathrm{B}}}\begin{pmatrix} \tilde{a}'_{i,u,1} \\ \vdots \\ \tilde{a}'_{i,u,k_\mathrm{B}/h_\mathrm{B}} \end{pmatrix} + \dots + \alpha^{(h_\mathrm{B}-1)j_{h_\mathrm{B}}}\begin{pmatrix} \tilde{a}'_{i,u,(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})+1} \\ \vdots \\ \tilde{a}'_{i,u,k_\mathrm{B}} \end{pmatrix} \end{pmatrix} \tag{5.38}$$

$$\in \mathbb{F}_q^{k_\mathrm{B}}. \tag{5.39}$$

各 $w \in [k_\mathrm{B}/h_\mathrm{B}]$ に対して，式 (5.38) の第 $w, w+(k_\mathrm{B}/h_\mathrm{B}), \dots, w+(k_\mathrm{B}/h_\mathrm{B})(n_\mathrm{B}-1)$ 成分を取り出してきたベクトルは,

$$\begin{pmatrix} \tilde{a}'_{i,u,w} + \alpha^{j_1}\tilde{a}'_{i,u,(k_\mathrm{B}/h_\mathrm{B})+w} + \dots + \alpha^{j_1(h_\mathrm{B}-1)}\tilde{a}'_{i,u,(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})+w} \\ \vdots \\ \tilde{a}'_{i,u,w} + \alpha^{j_{h_\mathrm{B}}}\tilde{a}'_{i,u,(k_\mathrm{B}/h_\mathrm{B})+w} + \dots + \alpha^{j_{h_\mathrm{B}}(h_\mathrm{B}-1)}\tilde{a}'_{i,u,(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})+w} \end{pmatrix} \tag{5.40}$$

$$= \begin{pmatrix} \alpha^{j_1} & \dots & \alpha^{j_1(h_\mathrm{B}-1)} \\ \vdots & \ddots & \vdots \\ \alpha^{j_{h_\mathrm{B}}} & \dots & \alpha^{j_{h_\mathrm{B}}(h_\mathrm{B}-1)} \end{pmatrix} \begin{pmatrix} \tilde{a}'_{i,u,w} \\ \tilde{a}'_{i,u,w+(k_\mathrm{B}/h_\mathrm{B})} \\ \dots \\ \tilde{a}'_{i,u,w+(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})} \end{pmatrix}. \tag{5.41}$$

式 (5.41) のベクトルは，$\mathbb{F}_q$ 上 $(n_\mathrm{B}, h_\mathrm{B})$ Reed Solomon 符号語（式 (5.42)）の第 $j_1, \dots, j_{h_\mathrm{B}}$ 成分を取り出してきたベクトルである.

$$\mathbf{RS}_{n_\mathrm{B}\times k_\mathrm{B}} \begin{pmatrix} \tilde{a}'_{i,u,w} \\ \tilde{a}'_{i,u,w+(k_\mathrm{B}/h_\mathrm{B})} \\ \dots \\ \tilde{a}'_{i,u,w+(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})} \end{pmatrix}. \tag{5.42}$$

これは式 (5.43) のベクトルを $\mathbb{F}_q$ 上 $(n_\mathrm{B}, h_\mathrm{B})$ Reed Solomon 符号化したものである.

$$\begin{pmatrix} \tilde{a}'_{i,u,w} \\ \tilde{a}'_{i,u,w+(k_\mathrm{B}/h_\mathrm{B})} \\ \dots \\ \tilde{a}'_{i,u,w+(h_\mathrm{B}-1)(k_\mathrm{B}/h_\mathrm{B})} \end{pmatrix}. \tag{5.43}$$

また, 各 $w \in [k_{\mathrm{B}}/h_{\mathrm{B}}]$ に対して, 式 (5.38) の第 $w, w+(k_{\mathrm{B}}/h_{\mathrm{B}}), \ldots, w+(k_{\mathrm{B}}/h_{\mathrm{B}})(n_{\mathrm{B}}-1)$ 成分を取り出すと式 (5.42) のベクトルになるようなベクトルは，次式のベクトルである．

$$\left((\tilde{\boldsymbol{a}}'_{iu})^{\top} \left(\begin{array}{ccc} \boldsymbol{G}_{\mathrm{B},1\cdot} & \ldots & \boldsymbol{G}_{\mathrm{B},n_{\mathrm{B}}\cdot} \end{array} \right)\right)^{\top} = \left((\tilde{\boldsymbol{a}}'_{iu})^{\top}\boldsymbol{G}_{\mathrm{B}}^{\top}\right)^{\top} \in \mathbb{F}_q^{(k_{\mathrm{B}}/h_{\mathrm{B}})n_{\mathrm{B}}}. \tag{5.44}$$

これを転置した行ベクトルをすべての $u \in [k_{\mathrm{A}}]$ について並べた行列は,

$$\left(\begin{array}{ccc} \tilde{a}'_{i,1,1} & \cdots & \tilde{a}'_{i,1,k_{\mathrm{B}}} \\ \vdots & \ddots & \vdots \\ \tilde{a}'_{i,k_{\mathrm{A}},1} & \cdots & \tilde{a}'_{i,k_{\mathrm{A}},k_{\mathrm{B}}} \end{array} \right) \boldsymbol{G}_{\mathrm{B}}^{\top} = \boldsymbol{G}_{\mathrm{A},i\cdot}^{\top} \left(\begin{array}{c} \boldsymbol{AB} \\ \boldsymbol{RB} \end{array} \right) \boldsymbol{G}_{\mathrm{B}}^{\top} \in \mathbb{F}_q^{k_{\mathrm{A}}\times(k_{\mathrm{B}}/h_{\mathrm{B}})n_{\mathrm{B}}} \tag{5.45}$$

各 $u \in [k_{\mathrm{A}}]$，各 $w \in [k_{\mathrm{B}}/h_{\mathrm{B}}]$ に対して，$\mathbb{F}_q$ 上 $(n_{\mathrm{B}}, h_{\mathrm{B}})$ Reed Solomon 符号の限界ハミング消失距離復号を行うことで式 (5.41) から式 (5.43) のベクトルを得て，それらを並べなおすことで $\boldsymbol{G}_{\mathrm{A},i\cdot}^{\top} \left(\begin{array}{c} \boldsymbol{AB} \\ \boldsymbol{RB} \end{array} \right)$ が得られる．以上より，命題が示された．□

式 (5.41) から式 (5.43) のベクトルを求めるために $\mathbb{F}_q$ 上 $(n_{\mathrm{B}}, h_{\mathrm{B}})$ Reed Solomon 符号の限界ハミング消失距離復号を行うが，この計算方法としては，2 章で述べた通り，$\mathbb{F}_q$ 上 $h_{\mathrm{B}}-1$ 次多項式のラグランジュ補間多項式のすべての係数を計算することで求める方法がある．

補題 5.4.1　改良方式の計算時間を，ワーカーの計算処理，マスターの復号処理における $\mathbb{F}_q$ 上四則演算の回数の総和と定めると，その上界は，以下で与えられる．

$$\begin{aligned} &\frac{k_{\mathrm{B}}(k_{\mathrm{A}}(l-1)+l(k_{\mathrm{B}}-1))}{h_{\mathrm{B}}} + k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}} \\ &+ h_{\mathrm{A}}(h_{\mathrm{A}}-1) \\ &+ \frac{k_{\mathrm{B}}(k_{\mathrm{A}}+k_{\mathrm{B}})l}{h_{\mathrm{B}}} + h_{\mathrm{A}}^2 + k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}} \\ &+ h_{\mathrm{A}} + \frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS}}}{h_{\mathrm{B}}} \end{aligned} \tag{5.46}$$

である．ただし，T_{RS} を $\mathbb{F}_q$ 上 $(n_{\mathrm{B}}, h_{\mathrm{B}})$ Reed Solomon 符号の復号アルゴリズムにおける演算回数であると定義する．

証明　$\mathbb{F}_q$ 上 $(n_{\mathrm{B}}, h_{\mathrm{B}})$ Reed Solomon 符号の復号アルゴリズムにおいて，$\mathrm{T}_{\mathrm{RS},+}$ を加算の回数，$\mathrm{T}_{\mathrm{RS},-}$ を減算の回数，$\mathrm{T}_{\mathrm{RS},*}$ を乗算の回数，$\mathrm{T}_{\mathrm{RS},/}$ を逆元を求める演算の回数であると定義する．本論文においては，$\mathrm{T}_{\mathrm{RS}} = \mathrm{T}_{\mathrm{RS},+} + \mathrm{T}_{\mathrm{RS},-} + \mathrm{T}_{\mathrm{RS},*} + \mathrm{T}_{\mathrm{RS},/}$ が成立する．各ワーカーの計算処理においては，加算が $\frac{k_{\mathrm{B}}(k_{\mathrm{A}}(l-1)+l(k_{\mathrm{B}}-1))}{h_{\mathrm{B}}}$ 回，乗算が $\frac{k_{\mathrm{B}}(k_{\mathrm{A}}+k_{\mathrm{B}})l}{h_{\mathrm{B}}}$ 回，かかる．実際，$\boldsymbol{B} \in \mathbb{F}_q^{l\times k_{\mathrm{B}}}$ と $\boldsymbol{G}_{\mathrm{B},j\cdot} \in \mathrm{F}_q^{k_{\mathrm{B}}\times(k_{\mathrm{B}}/h_{\mathrm{B}})}$ の積 $\tilde{\boldsymbol{B}}_j \coloneqq \boldsymbol{B}\boldsymbol{G}_{\mathrm{B},j\cdot}$

の計算に加算が $\frac{k_{\mathrm{B}}l(k_{\mathrm{B}}-1)}{h_{\mathrm{B}}}$ 回，乗算が $\frac{k_{\mathrm{B}}^2 l}{h_{\mathrm{B}}}$ 回，また，$\tilde{\boldsymbol{A}}_{i\cdot} \in \mathbb{F}_q^{k_{\mathrm{A}}\times l}$ と $\tilde{\boldsymbol{B}}_{j} \in \mathbb{F}_q^{l\times(k_{\mathrm{B}}/h_{\mathrm{B}})}$ の積 $\tilde{\boldsymbol{A}}_{i\cdot}\tilde{\boldsymbol{B}}_{\cdot j} \in \mathbb{F}_q^{k_{\mathrm{A}}\times(k_{\mathrm{B}}/h_{\mathrm{B}})}$ の計算に加算が $\frac{k_{\mathrm{A}}k_{\mathrm{B}}(l-1)}{h_{\mathrm{B}}}$ 回，乗算が $\frac{k_{\mathrm{A}}k_{\mathrm{B}}l}{h_{\mathrm{B}}}$ 回かかる．したがって，合計で，加算が $\frac{k_{\mathrm{B}}(k_{\mathrm{A}}(l-1)+l(k_{\mathrm{B}}-1))}{h_{\mathrm{B}}}$ 回，乗算が $\frac{k_{\mathrm{B}}(k_{\mathrm{A}}+k_{\mathrm{B}})l}{h_{\mathrm{B}}}$ 回，かかる．

復号処理 1-i においては，マスターの $\mathbb{F}_q$ 上演算回数は，加算は $\frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS},+}}{h_{\mathrm{B}}}$ 回，減算は $\frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS},-}}{h_{\mathrm{B}}}$ 回，乗算は $\frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS},*}}{h_{\mathrm{B}}}$ 回，逆元を求める演算は $\frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS},/}}{h_{\mathrm{B}}}$ 回，である．実際，$\mathbb{F}_q$ 上 $(n_{\mathrm{B}}, h_{\mathrm{B}})$ Reed Solomon 符号の復号を $k_{\mathrm{A}}k_{\mathrm{B}}/h_{\mathrm{B}}$ 回行うからである．

復号処理 2 においては，マスターが Algorithm 2 と同様の処理を行うので，$\mathbb{F}_q$ 上の演算回数は，加算が $k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}}$ 回，減算が $h_{\mathrm{A}}(h_{\mathrm{A}}-1)$ 回，乗算が $h_{\mathrm{A}}^2 + k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}}$ 回，逆元を求める演算が h_{A} 回である．

したがって，$\mathbb{F}_q$ 上の演算回数は，システム全体の合計で，加算が $\frac{k_{\mathrm{B}}(k_{\mathrm{A}}(l-1)+l(k_{\mathrm{B}}-1))}{h_{\mathrm{B}}} + \frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS},+}}{h_{\mathrm{B}}} + k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}}$ 回，減算が $\frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS},-}}{h_{\mathrm{B}}} + h_{\mathrm{A}}(h_{\mathrm{A}}-1)$ 回，乗算が $\frac{k_{\mathrm{B}}(k_{\mathrm{A}}+k_{\mathrm{B}})l}{h_{\mathrm{B}}} + \frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS},*}}{h_{\mathrm{B}}} + h_{\mathrm{A}}^2 + k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}}$ 回，逆元を求める演算が $\frac{k_{\mathrm{A}}k_{\mathrm{B}}\mathrm{T}_{\mathrm{RS},/}}{h_{\mathrm{B}}} + h_{\mathrm{A}}$ 回である．この合計が，$\mathbb{F}_q$ 上の演算回数の合計である．□

定理 5.4.2（改良方式の (a) に関する評価） $q, n_{\mathrm{B}}, h_{\mathrm{B}}$ は $n = n_{\mathrm{B}}, k = h_{\mathrm{B}}$ と置いたときに仮定 2.4.1 を満たしているとする．改良方式の計算時間を，ワーカーの計算処理，マスターの復号処理における $\mathbb{F}_q$ 上四則演算の回数の総和と定めると，その上界は，以下で与えられる．

$$\frac{2k_{\mathrm{A}}k_{\mathrm{B}} + 2k_{\mathrm{B}} - 1}{h_{\mathrm{B}}} l + \left(2k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}} + 2h_{\mathrm{A}}^2 - \frac{k_{\mathrm{A}}k_{\mathrm{B}}}{h_{\mathrm{B}}} + \frac{k_{\mathrm{A}}k_{\mathrm{B}}}{h_{\mathrm{B}}} T_{\mathrm{RS}}\right). \tag{5.47}$$

である．T_{RS} は式 (2.31) で定義される値である．

これは定理 5.4.2 と命題 2.4.8 から明らかである．

系 5.4.1（改良方式の (a) における単独方式との比較） $q, n_{\mathrm{B}}, h_{\mathrm{B}}$ は $n = n_{\mathrm{B}}, k = h_{\mathrm{B}}$ と置いたときに仮定 2.4.1 を満たすとする．単独方式の計算時間 $k_{\mathrm{A}}k_{\mathrm{B}}(2l-1)$ と比較すると，

$$l \geq \frac{2k_{\mathrm{A}}k_{\mathrm{B}}h_{\mathrm{A}} + 2h_{\mathrm{A}}^2 + \frac{k_{\mathrm{A}}k_{\mathrm{B}}}{h_{\mathrm{B}}}(T_{\mathrm{RS}}-1) + 1}{2k_{\mathrm{A}}k_{\mathrm{B}} - \frac{1}{h_{\mathrm{B}}}(2k_{\mathrm{A}}k_{\mathrm{B}} + 2k_{\mathrm{B}} - 1)} \tag{5.48}$$

$$= \frac{h_{\mathrm{A}} + \frac{1}{k_{\mathrm{A}}k_{\mathrm{B}}}h_{\mathrm{A}}^2 + \frac{1}{2h_{\mathrm{B}}}(T_{\mathrm{RS}}-1) + \frac{1}{2k_{\mathrm{A}}k_{\mathrm{B}}}}{1 - \frac{1}{h_{\mathrm{B}}}\left(1 + \frac{1}{k_{\mathrm{A}}} - \frac{1}{2k_{\mathrm{A}}k_{\mathrm{B}}}\right)} \tag{5.49}$$

かつ $h_{\mathrm{B}} > 1 + \frac{1}{k_{\mathrm{A}}} - \frac{1}{2k_{\mathrm{A}}k_{\mathrm{B}}}$ を満たすときに，改良方式の計算時間が少なくなる．T_{RS} は $n = n_{\mathrm{B}}, k = h_{\mathrm{B}}$ と置いたときに式 (2.31) で定義される値である．

例 5.4.1

$$(h_{\mathrm{A}}, h_{\mathrm{B}}, k_{\mathrm{A}}, k_{\mathrm{B}}) = (10, 50, 100, 293) \tag{5.50}$$

のとき，h_{B} が 2 以上の時に分母が正になる．例えば，$n_{\mathrm{B}} = 2^{2^3}, q = n + 1$ のとき，提案方式が有利であるような l の最小値は 756 である．

第 6 章

結論と今後の展望

本論文は，(a) 計算時間 (b) 信頼性 (c) 情報秘匿性という評価基準の下で良い性質を有する分散符号化計算方式の構成法に関する理論研究である．内容は大別して 2 つに分かれている．

1 つ目では，(a)，(b) の基準で良い性質を有する新たなグループ型分散符号化計算方式の構成法を提案した．本方式は，拡大体上ベクトル $\boldsymbol{ABv} \in \mathbb{F}_{q^m}^{k_\mathrm{A}}$ から $\mathbb{F}_{q^m}$ 上 (n, k_A) 符号語 $\boldsymbol{GA}(\boldsymbol{B}, \boldsymbol{0}_{l\times(m-k_\mathrm{B})})\boldsymbol{v} \in \mathbb{F}_{q^m}^{n}$ への変換と同等の処理を，グループ内で $\mathbb{F}_q$ 上の計算に分解して分散計算することにより，計算時間の短縮および列同士が依存する誤り行列の訂正を同時に可能にする方式である．さらに，この一例として，$\boldsymbol{ABv}$ を $\mathbb{F}_{q^m}$ 上 (n, k_A)Gabidulin 符号化する方式であるグループ型 G 方式の構成法を提案した．(a) に関しては，定理 4.3.3 で述べた通り，グループ型 G 方式の計算時間の評価を行った．さらに，系 4.3.1 で述べた通り，単独方式と比較して有利になるパラメータ n_A の値の条件を説明した．(b) に関しては，定理 4.3.1, 定理 4.3.2 で述べた通り，グループ型分散符号化計算方式，グループ型 G 方式それぞれの方式の誤り訂正能力を評価した．特に，定理 4.3.2 から，従来方式で訂正不可能な誤り行列 $\boldsymbol{E}$ を，グループ型 G 方式で訂正可能であることが示されると説明した．

2 つ目では，(a)，(b)，(c) の基準で良い性質を有するグループ型分散秘匿符号化計算方式の構成法 (基本方式) を提案した．本方式は，行列に対する $(h_\mathrm{A}, n_\mathrm{A})$ 秘密分散法（分散秘匿符号化計算方式 [5] と同様）を各グループ内で分散計算することにより，計算時間の短縮，一部のグループの計算結果の消失に対する訂正，およびグループ間での結託に対する $\boldsymbol{A}$ の値の秘匿を同時に可能にする方式である．(a) に関しては，1 つ目の時と同様に，定理 5.2.3 で述べた通り，システム全体での総計算時間を算出した．さらに，系 5.2.1 で述べた通り，単独方式のそれと比較して有利であるようなパラメータ $n_\mathrm{B}, h_\mathrm{A}$ の値が存在することを示した．(b) に関しては，定理 5.2.2 で述べた通り，マスターは h_A 個以上のワーカーグループのシェアから $\boldsymbol{AB}$ の値を復元でき

ることを示した．(c) に関しては，定理 5.2.1 で述べた通り，$\boldsymbol{A}$ の値の秘匿に関して，$h_{\mathrm{A}} - 1$ 個以下のワーカーグループが結託しても $\boldsymbol{A}$ の値に関する情報が全く漏洩しないことを示した．最後に，基本方式を，(b) 信頼性，(c) 情報秘匿性の点で一般化した方式の構成法を提案した．

今後の展望としては，一つ目としては，現在よりも計算時間が低い方式を提案することである．そのためには，その値が零であるような成分が多くなるように生成行列を構成すればよい．そのような考え方に基づく従来の分散符号化計算方式には例えば [40] がある．今後の展望の二つ目としては，各ワーカーにおけるデータの保存量や通信量が小さい方式を提案することである．秘匿性の制約が緩いがシェアのサイズが小さくなる方式を**ランプ型秘密分散方式** (例えば [41], [42]) と呼ぶ．この考え方を本論文のグループ型分散秘匿符号化計算方式に適用することで，秘匿性は弱いが通信，計算時間だけでなく，データの保存量も小さい方式を構成可能であると考えられる．

参考文献

[1] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran. Speeding up distributed machine learning using codes. *IEEE Transactions on Information Theory*, Vol. 64, No. 3, pp. 1514–1529, 2018.

[2] K.H. Huang and J. A. Abraham. Algorithm-based fault tolerance for matrix operations. *IEEE Transactions on Computers*, Vol. C-33, No. 6, pp. 518–528, 1984.

[3] A. Reisizadeh, S. Prakash, R. Pedarsani, and S. Avestimehr. Coded computation over heterogeneous clusters. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pp. 2408–2412, 2017.

[4] S. Dutta, V. Cadambe, and P. Grover. "short-dot": Computing large linear transforms distributedly using coded short dot products. *IEEE Transactions on Information Theory*, Vol. 65, No. 10, pp. 6171–6193, 2019.

[5] W. Chang and R. Tandon. On the capacity of secure distributed matrix multiplication. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, 2018.

[6] P. M. Olmos and R. Urbanke. Scaling behavior of convolutional ldpc ensembles over the bec. In *2011 IEEE International Symposium on Information Theory(ISIT)*, pp. 1816–1820, 2011.

[7] 鈴木陽一. 3-2 衛星デジタル放送システムにおける誤り訂正符号. 映像情報メディア学会誌, Vol. 70, No. 9, pp. 747–753, 2016.

[8] J. S. Ng, W. Y. B. Lim, N. C. Luong, Z. Xiong, A. Asheralieva, D. Niyato, C. Leung, and C. Miao. A comprehensive survey on coded distributed computing: Fundamentals, challenges, and networking applications. *IEEE Communications Surveys Tutorials*, Vol. 23, No. 3, pp. 1800–1837, 2021.

[9] C. Ordonez, Y. Zhang, and W. Cabrera. The gamma matrix to summarize dense and sparse data sets for big data analytics. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 28, No. 7, pp. 1905–1918, 2016.

[10] M. H. Qasem, A. A. Sarhan, R. Qaddoura, and B. A. Mahafzah. Matrix multiplica-

tion of big data using mapreduce: A review. In *2017 2nd International Conference on the Applications of Information Technology in Developing Renewable Energy Processes Systems (IT-DREPS)*, pp. 1–6, 2017.

[11] 須子統太, 堀井俊佑, 小林学, 後藤正幸, 松嶋敏泰, 平澤茂一. プライバシー保護機能を持つ線形回帰モデルにおける最小二乗推定量の分散計算法について. 日本経営工学会論文誌, Vol. 65, No. 2, pp. 78–88, 2014.

[12] A. Shamir. How to share a secret. *Communications of the ACM*, Vol. 22, No. 11, pp. 612–613, 1979.

[13] H. Sun and S. A. Jafar. The capacity of private information retrieval. *IEEE Transactions on Information Theory*, Vol. 63, No. 7, pp. 4075–4088, 2017.

[14] H. Sun and S. A. Jafar. The capacity of robust private information retrieval with colluding databases. *IEEE Transactions on Information Theory*, Vol. 64, No. 4, pp. 2361–2370, 2018.

[15] K. Banawan and S. Ulukus. The capacity of private information retrieval from byzantine and colluding databases. *IEEE Transactions on Information Theory*, Vol. 65, No. 2, pp. 1206–1219, 2019.

[16] H. Sun and S. A. Jafar. The capacity of private computation. *IEEE Transactions on Information Theory*, Vol. 65, No. 6, pp. 3880–3897, 2019.

[17] 桂利行. 代数学 Ⅲ 体とガロア理論. 大学数学の入門. 東京大学出版会, 2005.

[18] H. Niederreiter. *Finite Fields*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1984.

[19] 今井秀樹. 符号理論. 一般社団法人電子情報通信学会, 1990.

[20] S. Gao, D. Panario, V. Shoup, et al. Algorithms for exponentiation in finite fields. *Journal of Symbolic Computation*, Vol. 29, No. 6, pp. 879–889, 2000.

[21] M. Gadouleau and Zhiyuan Yan. Complexity of decoding gabidulin codes. In *2008 42nd Annual Conference on Information Sciences and Systems*, pp. 1081–1085, 2008.

[22] R. B. Venturelli and D. Silva. An evaluation of erasure decoding algorithms for gabidulin codes. In *2014 International Telecommunications Symposium (ITS)*, pp. 1–5, 2014.

[23] D. Silva and F. R. Kschischang. Fast encoding and decoding of gabidulin codes. In *2009 IEEE International Symposium on Information Theory (ISIT)*, pp. 2858–2862, 2009.

[24] S. Gao. *Normal bases over finite fields.* University of Waterloo Waterloo, 1993.

[25] T. Itoh and S. Tsujii. A fast algorithm for computing multiplicative inverses in

gf(2m) using normal bases. *Information and Computation*, Vol. 78, No. 3, pp. 171–177, 1988.

[26] J. Guajardo and C. Paar. Itoh-tsujii inversion in standard basis and its application in cryptography and codes. *Des. Codes Cryptography*, Vol. 25, No. 2, pp. 207–216, 2002.

[27] Abdulah Abdulah Zadeh. Division and inversion over finite fields. In Jaydip Sen, editor, *Cryptography and Security in Computing*, chapter 6. IntechOpen, Rijeka, 2012.

[28] J. M. Couveignes and R. Lercier. Elliptic periods for finite fields. *Finite Fields and Their Applications*, Vol. 15, No. 1, pp. 1–22, 2009.

[29] 鈴木武, 山田作太郎. 数理統計学-基礎から学ぶデータ解析. 内田老鶴圃, 1996.

[30] T.M Cover and J.A Thomas. *Elements of Information Theory, 2nd ed.* John Wiley & Sons, 2006.

[31] E. M. Gabidulin. Theory of codes with maximum rank distance. *Problems of Information Transmission (English translation of Problemy Peredachi Informatsii)*, Vol. 21, No. 1, pp. 1–12, 1985.

[32] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the society for industrial and applied mathematics*, Vol. 8, No. 2, pp. 300–304, 1960.

[33] Joachim von zur Gathen. Modern computer algebra / joachim von zur gathen, j´urgen gerhard. [electronic resource], 2013.

[34] Alexandre Soro and Jerome Lacan. Fnt-based reed-solomon erasure codes. In *2010 7th IEEE Consumer Communications and Networking Conference*, pp. 1–5, 2010.

[35] E. Karnin, J. Greene, and M. Hellman. On secret sharing systems. *IEEE Transactions on Information Theory*, Vol. 29, No. 1, pp. 35–41, 1983.

[36] F. F. Magoulès, F. X. Roux, 拓也桑原. 並列計算の数理とアルゴリズム. 森北出版, 2015.

[37] S. Lin and D. J. Costello. *Error control coding*, Vol. 2. Prentice hall Scarborough, 2001.

[38] K. Kazama, A. Kamatsuka, T. Yoshida, and T. Matsushima. A note on a relationship between smooth locally decodable codes and private information retrieval. In *2020 International Symposium on Information Theory and Its Applications (ISITA)*, pp. 259–263, 2020.

[39] H. Bartz, T. Jerkovits, S. Puchinger, and J. Rosenkilde. Fast decoding of codes in the rank, subspace, and sum-rank metric. *IEEE Transactions on Information Theory*,

Vol. 67, No. 8, pp. 5026–5050, 2021.

[40] S. Dutta, V. Cadambe, and P. Grover. Short-dot: Computing large linear transforms distributedly using coded short dot products. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pp. 2100–2108. Curran Associates, Inc., 2016.

[41] M. Iwamoto, H. Yamamoto, and H. Ogawa. Optimal multiple assignments based on integer programming in secret sharing schemes with general access structures. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, Vol. 90, No. 1, pp. 101–112, 2007.

[42] 山本博資. (k,l,n) しきい値秘密分散システム. 電子通信学会論文誌 A, Vol. 68, No. 9, pp. 945–952, 1985.

付録 A

復号アルゴリズムにおける有限体上四則演算回数

A.1 Reed Solomon 符号の消失復号アルゴリズムにおける有限体上四則演算回数

Reed-Solomon 符号の消失復号アルゴリズム [34] における有限体 $\mathbb{F}_q$ 上四則演算回数を論じる．[34] においては演算回数のオーダーのみが論じられているが，本論文では各 Step における回数を論じる．

いくつかのノーテーションを定義する．q, n に関しては仮定 2.4.1 が成立するとする．$\mathbb{F}_q$ 上 (n, k) Reed Solomon 符号においては，$\mathbb{F}_q$ 上高々 $k-1$ 次多項式 $f(x)$（情報多項式）が符号語 $(f(\alpha^1), \ldots, f(\alpha^n))$ に符号化される．残った k 個のシンボルの位置を $i_1, \ldots, i_k \in [n]$ とおく．このとき，ラグランジュの多項式補間 (命題 2.4.7) から，

$$\frac{1}{A(x)} f(x) = \frac{1}{A(x)} \sum_{h=1}^{k} y_{i_h} \frac{\prod_{h' \in [k] \setminus \{h\}} (x - x_{i_{h'}})}{\prod_{h' \in [k] \setminus \{h\}} (x_{i_h} - x_{i_{h'}})} \tag{A.1}$$

$$= \sum_{h=1}^{k} \frac{\frac{y_{i_h}}{A'(x_{i_h})}}{x - x_{i_h}}. \tag{A.2}$$

ただし，$A(x) = \sum_{i \in [0,k]} a_i x^i$ を $\prod_{h' \in [k]} (x - x_{i_{h'}})$，$A(x)$ を x で微分した多項式 $A'(x)$ を $\sum_{i \in [0,k-1]} (i+1) a_{i+1} x^i$ と定義する．ただし，$i+1$ は $\mathbb{F}_q$ の情報単位元 1 を $i+1$ 回足したものである．式 (A.1) から式 (A.2) への変形は [34] を参照せよ．ここで，式 (A.2) 右辺に着目すると，右辺の k 個の分数を通分して足したとき，その分子多項式が $f(x)$ である．

以上のノーテーションの下，復号アルゴリズムの概要を説明する．入力は符号語か

ら高々 $n-k$ シンボルが消失することにより発生したベクトル $\boldsymbol{y}$ であり，出力は情報多項式 $f(x)$ である．

1. 多項式 $A(x)$ を計算する．
2. $A(x)$ を x で微分した多項式 $A'(x)$ を計算する．
3. $A'(\alpha^{i_1}),\ldots,A'(\alpha^{i_k}) \in \mathbb{F}_q$ を計算する．
4. $\frac{y_{i_1}}{A'(\alpha^{i_1})},\ldots,\frac{y_{i_k}}{A'(\alpha^{i_k})}$ を計算する．
5. 高々 $n-1$ 次多項式 $\sum_{h\in[k]} \frac{\frac{y_{i_h}}{A'(\alpha^{i_h})}}{x-\alpha^{i_h}} \mod x^n$ を計算する．ここで，$\sum_{h\in[k]} \frac{\frac{y_{i_h}}{A'(\alpha^{i_h})}}{x-\alpha^{i_h}} \mod x^n$ とは，$\sum_{h\in[k]} \frac{\frac{y_{i_h}}{A'(\alpha^{i_h})}}{x-\alpha^{i_h}}$ をテイラー展開して得た冪級数の高々 $n-1$ 次以下の部分である．
6. $f(x) = A(x)\sum_{h\in[k]} \frac{\frac{y_{i_h}}{A'(\alpha^{i_h})}}{x-\alpha^{i_h}} \mod x^n$ を計算する．

Step 1 の演算回数は高々 $(9(\lceil\log_2 k\rceil)^2+9\lceil\log_2 k\rceil+19)2^{\lceil\log_2 k\rceil}-\frac{27}{2}\lceil\log_2 k\rceil+35$ 回である．証明は後述する．

Step 2 においては，$A(x)$ が与えられたもとで，$k-1$ 回の加算により $2,\ldots,k$ を求めてから，k 回の乗算により各 $i\in[0,k-1]$ に対して $a_{i+1}(i+1)$ を求めることで，$A'(x)$ を求める．よって，$2k-1$ 回の演算回数により Step 2 は行われる．

Step 3 の演算回数は高々 $\frac{3}{2}n\log_2 n$ 回である．証明は後述する．

Step 4 においては，k 回の乗算と k 回の (乗法的) 逆元を求める演算により，各 $h\in[k]$ に対して $y_{i_h}(A'(\alpha^{i_h}))^{-1}$ を求める．よって，$2k$ 回の演算回数により Step 4 は行われる．

Step 5 の演算回数は高々 $\frac{3}{2}n\log_2 n$ 回である．証明は後述する．

Step 6 の演算回数は高々 $18n\log_2 n+\frac{27}{2}n-8$ 回である．証明は後述する．

以上より，演算回数は高々 $(9(\lceil\log_2 k\rceil)^2+9\lceil\log_2 k\rceil+19)2^{\lceil\log_2 k\rceil}-\frac{27}{2}\lceil\log_2 k\rceil+4k+21n\log_2 n+\frac{27}{2}n+26$ である．

A.1.1 Step 1,3,5,6 の演算回数

Step 1,3,5 の演算回数について説明する．本小節では，[33](Chapter 8) の定義，定理を引用する．[33] に記載されている定理の多くは可換環に対して適用されているが，本論文では仮定 2.4.1 が成立する有限体 $\mathbb{F}_q$ と正整数 n に限定して論じる．

定義 A.1.1（離散フーリエ変換 [33]） α を位数 n の元とする．$\mathbb{F}_q$ 上の高々 $n-1$ 次多項式 $f(x)$ に対し，

$$\mathrm{DFT}_\alpha(f) \coloneqq (f(\alpha),\ldots,f(\alpha^{n-1}),f(\alpha^n)) = (f(\alpha),\ldots,f(\alpha^{n-1}),f(1)) \quad (A.3)$$

と定義する．写像 DFT_α を**離散フーリエ変換**と呼ぶ.

これを高速に計算する手法を**高速フーリエ変換**と呼ぶ.

命題 A.1.1（高速フーリエ変換の計算時間）　$\mathbb{F}_q$ 上の高々 $n-1$ 次多項式 $f(x)$ の体 $\mathbb{F}_q$ 上フーリエ変換は，$\mathbb{F}_q$ 上の四則演算を $\frac{3}{2}n\log_2 n$ 回行うことによって求められる．

これは [33] の Theorem 8.15 のある特殊化である．

命題 A.1.2　2 個の $\mathbb{F}_q$ 上高々 $n-1$ 次多項式 $f(x), g(x)$ の積は高々 $18n\log_2 n+\frac{27}{2}n-8$ 回で計算可能である．

これは [33] の Corollary 8.19 を特殊化した命題であり，[33] の Theorem 8.18 の証明から直ちに導かれる．

以降，各 Step の演算回数を論じる．

命題 A.1.3（Step 1 の演算回数）　$\alpha^{i_1},\ldots,\alpha^{i_k}$ が与えられたもと, 多項式 $A(x)$ を高々

$$(9(\lceil\log_2 k\rceil)^2+9\lceil\log_2 k\rceil+19)2^{\lceil\log_2 k\rceil}-\frac{27}{2}\lceil\log_2 k\rceil+35 \tag{A.4}$$

回の演算で求めることができる．

証明には分割統治法を用いる．

証明　k が 2 の冪である場合にのみ示す．そうでない場合でも同様に示すことが可能である．

$A(x)$ を以下の通りに計算する．以下の Step を Step1-1,1-2,1-2,... とおく．

1. $(x-\alpha^{i_1})\,(x-\alpha^{i_2})$，$(x-\alpha^{i_3})\,(x-\alpha^{i_4})$，... を計算する．次に，$(x-\alpha^{i_1})\,(x-\alpha^{i_2})\,(x-\alpha^{i_3})\,(x-\alpha^{i_4})$，$(x-\alpha^{i_5})\,(x-\alpha^{i_6})\,(x-\alpha^{i_7})\,(x-\alpha^{i_8})$，... を計算する．
2. $(x-\alpha^{i_1})\ldots(x-\alpha^{i_8})$，$(x-\alpha^{i_9})\ldots(x-\alpha^{i_{16}})$，... を計算する．
3. 以下同様に繰り返すことにより，最終的に $A(x)$ を求める．

以下では，$\log_2 k$ を c とおく．各 $i\in[\log_2 k]$ に対し，上記の Step1-i においては，2^{i-1} 次多項式の積を 2^{-i+c} 回行う．ここで，2^{i-1} 次多項式は $2^{i-1}-1$ 次以下の多項式であるため，以下のように演算回数の上界が求められる．

$$\sum_{i=1}^{c}2^{-i+c}\left(18\cdot i\cdot 2^i+\frac{27}{2}i-8\right) \tag{A.5}$$

$$=18\frac{c(c+1)}{2}2^c+\frac{27}{2}\left(\sum_{i=0}^{c-1}2^i(c-i)\right)-8(2^c-1) \tag{A.6}$$

$$= (9c^2+9c)2^c - 8\cdot 2^c + 8 + \frac{27}{2}\left(c(2^c-1) - \sum_{i=0}^{c-1} 2^i i\right) \tag{A.7}$$

$$= (9c^2+9c)2^c - 8\cdot 2^c + 8 + \frac{27}{2}\left(c(2^c-1) - 2((c-1)2^c - c2^{c-1} + 1)\right) \tag{A.8}$$

$$= (9c^2+9c)2^c - 8\cdot 2^c + 8 + \frac{27}{2}\left(2^{c+1} - c - 2\right) \tag{A.9}$$

$$= (9c^2+9c+19)2^c - \frac{27}{2}c + 35. \tag{A.10}$$

ここで，一般に，$S \coloneqq \sum_{i=0}^{c-1} x^i i$ の値は，

$$(x-1)S = (c-1)x^c - (x^{c-1} + \cdots + x) = (c-1)x^c - \frac{x^c-1}{x-1} + 1 \tag{A.11}$$

なので

$$S = \frac{1}{(x-1)^2}((c-1)(x-1)x^c - (x^c-1) + (x-1)) \tag{A.12}$$

$$= \frac{1}{(x-1)^2}((c-1)x^{c+1} - (c-1)x^c - x^c + x) \tag{A.13}$$

$$= \frac{x}{(x-1)^2}((c-1)x^c - cx^{c-1} + 1). \tag{A.14}$$

□

Step 3 では，高速フーリエ変換を用いて $A'(\alpha^i)$, $i \in [n]$ を計算することにより行われる．よって，その演算回数は，命題 A.1.1 より，高々 $\frac{3}{2}n\log_2 n$ であると示される．

Step 5 では，$N(x) \coloneqq \sum_{h=1}^{k} n_h x^{i_h}$，$n_h \coloneqq -\frac{y_{i_h}}{A'(\alpha^{i_h})}$ としたときの値 $N(1),\ldots,N(\alpha^{n-1})$ の値を求めることになる．よって，その演算回数は，命題 A.1.1 より，高々 $\frac{3}{2}n\log_2 n$ であると示される．

Step 6 の演算回数は，命題 A.1.2 より，高々 $18n\log_2 n + \frac{27}{2}n - 8$ であると示される．

A.2 Gabidulin 符号の復号アルゴリズムにおける有限体上四則演算回数

Gabidulin 符号の復号アルゴリズム [31] の概要と，その有限体上四則演算回数を，[21] の考え方に基づいて説明する．受信語 $\boldsymbol{y} \in \mathbb{F}_{q^m}^n$ を入力として，$\hat{\boldsymbol{c}} \in \mathbb{F}_{q^m}^n$ を出力とするアルゴリズムは以下の Step1,...,7 から構成される．以下では t を限界ランク距離 $\lfloor (n-k_{\mathrm{A}})/2 \rfloor$，$d$ を最小ランク距離 $n-k_{\mathrm{A}}+1 (\geq 2)$ と定義する．[21] との差異は後述するが，特に Step 3 が大きく異なる．

1. シンドローム $(s_1,\ldots,s_{d-1})^\top \coloneqq \boldsymbol{H}^\top \boldsymbol{y} \in \mathbb{F}_{q^m}^{d-1}$ を計算する．

このとき，$\mathbb{F}_{q^m}$ 上加算は $(n-1)(d-1)$ 回，$\mathbb{F}_{q^m}$ 上乗算が $n(d-1)$ 回必要である．

2. 線形化多項式 $S(x) \coloneqq \sum_{i\in[n-k]} s_i x^{q^{i-1}}$ に対し，拡張ユークリッド互除法を用いて，$\deg_q F(x) < t$，$F(x) = \Lambda(x) * S(x) \mod z^{q^{d-1}}$ を満たす線形化多項式 $F(x), \Lambda(x)$ を計算する．
   計算時間の上界については，$\mathbb{F}_{q^m}$ 上加算が $(d-1)(d-2) - \frac{1}{2}t(t-1) + 2t(d-t-2)$ 回，$\mathbb{F}_{q^m}$ 上乗算が $(d-1)(d-2) - \frac{1}{2}t(t-5) + 2(d-t-1)(d-t-2)$ 回，$\mathbb{F}_{q^m}$ 上逆元を求める演算[*1]が t 回必要である．詳細は後述する．
3. $\Lambda(x)$ の t 個の根 $E_1, \ldots, E_t \in \mathbb{F}_{q^m}$ と値を計算する[*2]．
   計算時間については，$\mathbb{F}_q$ 上加算，減算，乗算，逆元を求める演算の合計が $\frac{2}{3}(m(m-1)(2m-1) - t(t-1)(2t-1)) - (t-2)(m-t) - (t-1)(m(m-1) - t(t-1)) + m^2 + m - 1$ 回，$\mathbb{F}_{q^m}$ 上加算が tm 回，$\mathbb{F}_{q^m}$ 上乗算が $(t+1)m$ 回である．詳細は後述する．
4. 各 $w \in [r]$ に対して $E_1 x_1^w + \cdots + E_t x_t^w = s_w$ を満たす $x_1, \ldots, x_t \in \mathbb{F}_{q^m}$ を計算する．
   計算時間については，$\mathbb{F}_{q^m}$ 上加算が $\frac{3}{2}t(t-1)$ 回，$\mathbb{F}_{q^m}$ 上乗算が $(t+1)\left(\frac{3}{2}t - 1\right)$ 回，$\mathbb{F}_{q^m}$ 上逆元を求める演算が t 回である．導出は [21] と同様なので省略する．
5. 各 $w \in [0, t-1]$ に対して $x_w = Y_{w1}h_1 + \cdots + Y_{wn}h_n$ を満たす $\boldsymbol{Y} \in \mathbb{F}_q^{t\times n}$ を計算する．
   計算時間は，$\{h_1, \ldots, h_n\}$ が正規基底 $\{v_1, \ldots, v_n\}$ に等しいとすると，演算回数 0 である．
6. $\boldsymbol{e} = \boldsymbol{Y}(E_1, \ldots, E_t)^\top \in \mathbb{F}_{q^m}^n$ を計算する．
   計算時間については，$\boldsymbol{Y}$ は $\mathbb{F}_q$ 上 $n \times t$ 行列であり，$(E_1, \ldots, E_t)^\top \in \mathbb{F}_{q^m}^t$ も $\mathbb{F}_q$ 上 $t \times m$ 行列に対応させられることを考えると，$\mathbb{F}_q$ 上加算，減算，乗算，逆元を求める演算の合計が $mn(2t-1)$ 回である．
7. $\boldsymbol{x} = \boldsymbol{y} - \boldsymbol{e} \in \mathbb{F}_{q^m}^n$ を計算する．
   計算時間は，$\mathbb{F}_q$ 上減算が mn 回である．

本論文と [21] では，以下のような $\mathbb{F}_q$ 上四則演算回数の差異が現れる．Step 2 において，線形化多項式の積の各係数を計算するときの演算回数に違いが現れることにより，本論文で算出した合計演算回数と，[21] で算出した回数が異なる．また，Step 6 においては，本論文では，mnr 回の $\mathbb{F}_q$ 上乗算と「$mn(t-1)$ 回」の $\mathbb{F}_q$ 上加算により

[*1] a から a^{-1} を求める演算を指す．除算 ab^{-1} は b^{-1} を求めてから乗算 ab^{-1} を求める演算である．
[*2] t 個未満しか解がない場合もあるが，ここでは簡単のため t 個にする

計算できると算出したが，[21] では mnt 回の $\mathbb{F}_q$ 上乗算と「mnt 回」の $\mathbb{F}_q$ 上加算により計算できると算出した．さらに，Step 3 の計算回数の導出においては，確率的アルゴリズムと確定的アルゴリズムを扱う点で計算量評価が全く異なる．

A.2.1 Step 2 の詳細

Step 2 は以下の 1,2 によって行われる．以下の 1,2 を以降では Step2-1,2-2 と称する．入力はシンドローム線形化多項式 $S(x) \in \mathbb{F}_{q^m}[x]$，出力は $F(x), \Lambda(x) \in \mathbb{F}_{q^m}[x]$ である．

1. $F_0(x) = x^{q^{d-1}}$，$F_1(x) = S(x)$ とおく．$\mathbb{F}_{q^m}$ 上線形化多項式 $G_1(x), G_2(x), \ldots$ および $F_0(x), F_1(x), \ldots$ を再帰的に以下のように定義する．各 $i = 0, 1, \ldots$ に対し，
$$\begin{cases} F_i(x) = G_{i+1}(x) * F_{i+1}(x) + F_{i+2}(x) \\ \deg_q F_{i+2}(x) < \deg_q F_{i+1}(x) < \deg_q F_i(x). \end{cases} \tag{A.15}$$
以降，$\deg_q F_{i+1}(x) < \frac{d-1}{2} \leq \deg_q F_i(x)$ を満たす正整数 i を r とおく．
2. $A_{-1}(x) = 0$，$F_0(x) = x$ とおく．$\mathbb{F}_{q^m}$ 上線形化多項式 $A_1(x), A_2(x), \ldots$ を再帰的に以下のように定義する．各 $i = 1, 2, \ldots, r$ に対し，
$$A_i(x) = G_i(x) * A_{i-1}(x) + A_{i-2}(x) \tag{A.16}$$
と定義し，さらに，$\Lambda(x)$ を $A_r(x)$ をその最高次係数で割ったモニック線形化多項式とおく．

この Step2-1,2-2 の計算時間について説明する．

命題 A.2.1 変数が x の $\mathbb{F}_{q^m}$ 上線形化多項式 $F(x) = \sum_{i\in[0,u]} f_i x^{q^i}$，$G(x) = \sum_{i\in[0,v]} g_i x^{q^i}$，$f_u g_v \neq 0$ の積は，$\mathbb{F}_{q^m}$ 上加算 uv 回，$\mathbb{F}_{q^m}$ 上乗算 $(u+1)(v+1)$ 回を行うことで計算できる．

[21] と本論文ではこの回数の算出が異なっている．[21] では，加算 $uv - u - v - 1$ 回，乗算 uv 回として算出している．

証明 積の回数を算出する．$f_j g_k^{q^k}, i \in [0, u], k \in [0, v]$ の計算には，積が $(u+1)(v+1)$ 回必要である．ただし，仮定 2.1.2 から，g_k が与えられたもとでの $g_k^{q^k}$ の計算時間は無視するものとした．和の回数は，積の回数の総和から積線形多項式の q-次数を引けばよい．□

命題 A.2.2 $\mathbb{F}_{q^m}$ 上の線形化多項式 $F(x)$，$G(x)$ の q-次数をそれぞれ u, v とおく．ここで，$u \leq v$ とする．$F(x)$ を $G(x)$ で割った時の商線形化多項式と剰余線形化多項式

の組，すなわち $F(x) = Q(x) * G(x) + R(x)$ を満たす組 $(Q(x), R(x))$ を求めるには，最大で $\mathbb{F}_{q^m}$ 上の加算が $v(u-v+1)$ 回，乗算が $(v+1)(u-v+1)$ 回，逆元を求める演算が 1 回必要である．

今回は証明は省略．この結果は [21] と同じである．

命題 A.2.3　上記のアルゴリズムにおいて，$r \leq t$ および $\deg_q \Lambda(x) \leq t$ が成立する．

証明　簡単のため，$\deg_q F_i(x)$ を d_i と表す．各 $i = 0, 1, \ldots, r$ に対して，$d_{i+1} < d_i$ および $d_i < d - 1 - i$ が成立する．よって，

$$(d_{r+1} <) \frac{d-1}{2} \leq d_r \leq d - 1 - r. \tag{A.17}$$

このことから，$r \leq t$ が示される．

式 (A.15) から，各 $i \in [r]$ に対して $\deg_q G_i(x) = d_{i-1} - d_i$ が成立することが示される．一方，式 (A.16) から，各 $i \in [r]$ に対して $\deg_q A_i(x) = d_{i-1} - d_i + \deg_q A_{i-1}(x)$ が成立することが示される．よって，$\deg_q A_i(x)$ が $d_0 - d_i$，すなわち $d - 1 - d_i$ であることがわかる．よって，

$$\deg_q \Lambda(x) = \deg_q A_r(x) = d - 1 - d_r \leq d - 1 - \frac{d-1}{2} = \frac{d-1}{2}. \tag{A.18}$$

このことから，$\deg_q \Lambda(x) \leq t$ が示される．□

Step 2-1 は，F_{q^m} 上の逆元を求める演算が t 回，乗算が高々 $2t(d-1) - \frac{1}{2}t(t-1)$ 回，加算が高々 $t(d-1)$ 回必要あれば求められる．以下で理由を説明する．命題 A.2.2 から，各 $i = 0, 1, \ldots, r-1$ において，$F_i(x), F_{i+1}(x)$ から $G_{i+1}(x), F_{i+2}(x)$ を求めるには，F_{q^m} 上の逆元を求める演算が 1 回，乗算が $(d_{i+1}+1)(d_i - d_{i+1} + 1)$ 回，加算が $d_{i+1}(d_i - d_{i+1} + 1)$ 回必要である．これを $i = 0, \ldots, r-1$ まで行うことにより，$F_0(x)$ から $F_{r+1}(x)$ まで，および $G_1(x)$ から $G_r(x)$ までを求めるには，F_{q^m} 上の逆元を求める演算が $r(\leq t)$ 回，乗算が高々 $(d-2)(d-1) - \frac{1}{2}t(t-5)$ 回，加算が高々 $(d-2)(d-1) - \frac{1}{2}t(t-1)$ 回あれば良いことがわかる．乗算の回数の上界については，以下の通り求められる．

$$\sum_{i=0}^{r-1} (d_{i+1}+1)(d_i - d_{i+1} + 1) \tag{A.19}$$

$$= r + \sum_{i=0}^{r-1} d_{i+1}(d_i - d_{i+1}) + \sum_{i=0}^{r-1} d_i \tag{A.20}$$

$$\leq r + d_1 \sum_{i=0}^{r-1} (d_i - d_{i+1}) + \sum_{i=0}^{r-1} (d - 1 - i) \tag{A.21}$$

$$\leq r + (d-2)(d-1-d_r) + rd - \frac{1}{2}r(r+1) \tag{A.22}$$

$$= (d-2)(d-1) - \frac{1}{2}r(r-5) + (d-2)(r-d_r) \tag{A.23}$$

$$\leq (d-2)(d-1) - \frac{1}{2}r(r-5) + (d-2)(r - \frac{d-1}{2}) \tag{A.24}$$

$$\leq (d-2)(d-1) - \frac{1}{2}t(t-5) + (d-2)(t - \frac{d-1}{2}) \tag{A.25}$$

$$\leq (d-2)(d-1) - \frac{1}{2}t(t-5). \tag{A.26}$$

加算については次のように算出できる．

$$\sum_{i=0}^{r-1} d_{i+1}(d_i - d_{i+1} + 1) \tag{A.27}$$

$$= -(d_0 - d_r + r) + \sum_{i=0}^{r-1}(d_{i+1}+1)(d_i - d_{i+1} + 1) \tag{A.28}$$

$$\leq -2r + (d-2)(d-1) - \frac{1}{2}t(t-5). \tag{A.29}$$

Step 2-2 は，F_{q^m} 上乗算が高々 $\frac{(d-1)(d+2t-5)}{4}$ 回，加算が高々 $\frac{(d-3)(d+2t-5)}{4}$ 回あれば求められる．以下で理由を説明する．命題 A.2.1 から，各 $i = 1, \ldots, r$ において，$A_{i-2}(x), A_{i-1}(x), G_i(x)$ から $A_i(x)$ を求めるには，F_{q^m} 上乗算が

$$(1 + \deg_q G_i(x))(1 + \deg_q A_{i-1}(x)) = (d_{i-1} - d_i + 1)(d - d_{i-1}) \tag{A.30}$$

回，加算が

$$\deg_q G_i(x) \deg_q A_{i-1}(x) + (1 + \deg_q A_{i-2}(x)) \tag{A.31}$$

$$= (d_{i-1} - d_i)(d - 1 - d_{i-1}) + (d - d_{i-2}) \tag{A.32}$$

回必要である．ところで，$i = 1$ においては，$A_i(x) = G_1(x)$ なので，計算は不要である．そこで，この計算を $i = 2, \ldots, r$ で行うことを考える．$A_r(x)$ までを求めるには，F_{q^m} 上乗算が高々 $2(d-t-1)(d-t-2)$ 回，加算が高々 $2t(d-t-2)$ 回あれば良いことがわかる．$(t \geq) r \geq 2$ の場合，乗算の回数の上界については，以下の通り求められる．

$$\sum_{i=2}^{r}(d_{i-1} - d_i + 1)(d - d_{i-1}) \leq \sum_{i=2}^{r}(d_{i-1} - d_i + 1)(d - d_{r-1}) \tag{A.33}$$

$$= (d_1 - d_r + r - 1)(d - d_{r-1}) \tag{A.34}$$

$$\leq (d - 2 - \frac{d-1}{2} + r - 1)(d - d_r - 1) \tag{A.35}$$

$$\leq \frac{d+2t-5}{2}\frac{d-1}{2} \tag{A.36}$$

$$\leq (2d-2t-4)\frac{d-1}{2} \tag{A.37}$$
$$\leq 2(d-t-2)(d-t-1). \tag{A.38}$$

式 (A.38) の値は，$r=1$ のときの演算回数 (=0) の上界でもある．加算の回数の上界も同様にして求められる．$(t \geq) r \geq 2$ とする．このとき，$d \geq 5$ である．

$$\sum_{i=2}^{r}(d_{i-1}-d_i)(d-1-d_{i-1})+(d-d_{i-2}) \tag{A.39}$$
$$\leq \sum_{i=2}^{r}(d_{i-1}-d_i)(d-1-d_{i-1})+(d-1-d_{i-1}) \tag{A.40}$$
$$\leq (d-1-d_{r-1})\sum_{i=2}^{r}(d_{i-1}-d_i+1) \tag{A.41}$$
$$\leq (d-2-d_r)(d_1-d_r+r-1) \tag{A.42}$$
$$\leq (d-2-\frac{d-1}{2})(d-2-\frac{d-1}{2}+t-1) \tag{A.43}$$
$$\leq \frac{d-3}{2}\frac{d+2t-5}{2} \tag{A.44}$$
$$\leq \frac{d-3}{2}(2d-2t-4) \tag{A.45}$$
$$\leq 2t(d-t-2) \tag{A.46}$$

式 (A.46) の値は，$r=1$ のときの演算回数 (=0) の上界でもある．よって，Step 2 においては，上記で示した演算回数の上界が求められる．

A.2.2 Step 3 の詳細

Step 3 は以下の 1,2 によって行われる．以下の 1,2 を以降では Step 3-1,3-2 と称する．入力は線形化多項式 $\Lambda(x)$ であり，出力は t 以下の自然数 r および $\mathbb{F}_q$ 上線形独立な r 個の根 $E_1,\ldots,E_r \in \mathbb{F}_{q^m}$ である．ただし，$\Lambda(x)$ の根全体の集合がなす $\mathbb{F}_q$ 上線形空間の次元を r とおく．

1. $\Lambda(v_1),\ldots,\Lambda(v_m)$ を求める．
   $\mathbb{F}_{q^m}$ 上乗算が $(t+1)m$ 回，$\mathbb{F}_{q^m}$ 上乗算が tm 回である．
2. $\Lambda \in \mathbb{F}_q^{m\times m}$ を，
   $$\boldsymbol{f}^m\left((\Lambda(v_1),\ldots,\Lambda(v_m))^\top\right) \in \mathbb{F}_{q^m}^m \tag{A.47}$$
   と定義する．$\boldsymbol{U}\Lambda=\boldsymbol{0}$ が成立するフルランク行列 $\boldsymbol{U} \in \mathbb{F}_q^{r\times m}$ を $\mathbb{F}_q$ 上ガウス消去法で求め，各 $i \in [r]$ に対し $E_i = u_{i1}v_1+\cdots+u_{im}v_m$ とおく．
   $\mathbb{F}_q$ 上四則演算回数の合計の上界は，合計 $m^2+m-1+\frac{2}{3}(m(m-1)(2m-1)-t(t-1)(2t-1))-(t-2)(m-t)-(t-1)(m^2-m-t^2+t)$ 回である．

このとき，$r = m - \mathrm{rank}\Lambda$ が成立する．

以下で $\mathbb{F}_q$ 上四則演算回数の上界を導出する．Step 3 は，線形化多項式 $\Lambda(x) \in \mathbb{F}_{q^m}[x]$ を入力したら，$\mathbb{F}_q$ 上線形独立な $r(\geq t)$ 個の根 $E_1, \ldots, E_r \in \mathbb{F}_{q^m}$ を出力するアルゴリズム．

命題 A.2.4　正整数 $r \in [t]$ を取る．各 $i \in [r]$ に対し，E_i を（$v_i = (v_1)^{q^{i-1}}$ を満たす）$\mathbb{F}_{q^m}$ の $\mathbb{F}_q$ 上正規基底 $\{v_1, \ldots, v_m\} \subset \mathbb{F}_{q^m}$ の $\mathbb{F}_q$ 上線形結合で表す，すなわち $E_i = u_{i1}v_1 + \cdots + u_{im}v_m$，$u_{i1}, \ldots, u_{im} \in \mathbb{F}_q$．$E_1, \ldots, E_r \in \mathbb{F}_{q^m}$ が線形化多項式 $\Lambda(x) \in \mathbb{F}_{q^m}[x]$ の根であり，$\mathbb{F}_q$ 上一次独立である必要十分条件は，$\boldsymbol{U}\Lambda = \boldsymbol{0}$ かつ $\mathrm{rank}(\boldsymbol{U}) = r$ が成立することである．

証明　$E_1, \ldots, E_r \in \mathbb{F}_{q^m}$ が $\mathbb{F}_q$ 上線形独立である必要十分条件は，u_{ij} を並べてできる $\mathbb{F}_q$ 上 $r \times m$ 行列 $\boldsymbol{U}$ がランク $\min\{m, r\} = r$ の行列であることである．実際，以下の式で，$\boldsymbol{a} = (a_1, \ldots, a_r)$ とおくと，

$$\forall \boldsymbol{a} \in \mathbb{F}_q^r, \sum_{i\in[t]} a_i E_i = 0 \Rightarrow \boldsymbol{a} = \boldsymbol{0} \tag{A.48}$$

$$\Leftrightarrow \forall \boldsymbol{a} \in \mathbb{F}_q^r, \sum_{i\in[t]} a_i \sum_{j\in[m]} u_{ij}v_j = 0 \Rightarrow \boldsymbol{a} = \boldsymbol{0} \tag{A.49}$$

$$\Leftrightarrow \forall \boldsymbol{a} \in \mathbb{F}_q^r, \sum_{j\in[m]} \left(\sum_{i\in[t]} a_i u_{ij} \right) v_j = 0 \Rightarrow \boldsymbol{a} = \boldsymbol{0} \tag{A.50}$$

$$\Leftrightarrow \forall \boldsymbol{a} \in \mathbb{F}_q^r, \forall j \in [m], \sum_{i\in[t]} a_i u_{ij} = 0 \Rightarrow \boldsymbol{a} = \boldsymbol{0} \tag{A.51}$$

$$\Leftrightarrow \forall \boldsymbol{a} \in \mathbb{F}_q^r, \forall j \in [m], \sum_{i\in[t]} a_i u_{ij} = 0 \Rightarrow \boldsymbol{a} = \boldsymbol{0} \tag{A.52}$$

$$\Leftrightarrow \forall \boldsymbol{a} \in \mathbb{F}_q^r, \boldsymbol{U}\boldsymbol{a} = \boldsymbol{0} \Rightarrow \boldsymbol{a} = \boldsymbol{0} \tag{A.53}$$

$$\Leftrightarrow \mathrm{rank}(\boldsymbol{U}) = \min\{m, r\} = r. \tag{A.54}$$

$E_1, \ldots, E_r$ が線形化多項式 $\Lambda(x)$ の根である必要十分条件は，$\boldsymbol{U}\Lambda = \boldsymbol{0}$ が成立することである．

$$\forall i \in [r], \Lambda(E_i) = 0 \Leftrightarrow \forall i \in [r], \Lambda(u_{i1}v_1 + \cdots + u_{im}v_m) = 0 \tag{A.55}$$

$$\Leftrightarrow \forall i \in [r], u_{i1}\Lambda(v_1) + \cdots + u_{im}\Lambda(v_m) = 0 \tag{A.56}$$

$$\Leftrightarrow \begin{pmatrix} u_{11} & \ldots & u_{1m} \\ \vdots & \ddots & \vdots \\ u_{r1} & \ldots & u_{rm} \end{pmatrix} \begin{pmatrix} \Lambda(v_1) \\ \vdots \\ \Lambda(v_m) \end{pmatrix} = \boldsymbol{0} (\in \mathbb{F}_{q^m}^r) \tag{A.57}$$

$$\Leftrightarrow \boldsymbol{U}\Lambda\boldsymbol{v} = \boldsymbol{0} (\in \mathbb{F}_{q^m}^r) \tag{A.58}$$

$$\Leftrightarrow \boldsymbol{U}\Lambda = \boldsymbol{0} (\in \mathbb{F}_q^{r\times m}). \tag{A.59}$$

以上のことから，$E_1, \ldots, E_r$ が線形化多項式 $\Lambda(x)$ の根であり $\mathbb{F}_q$ 上一次独立である必要十分条件は，$\boldsymbol{U}\Lambda = \boldsymbol{0}$ かつ $\mathrm{rank}(\boldsymbol{U}) = r$ が成立することである．□

このことから，線形化多項式 $\Lambda(x)$ が入力されたら，$(\Lambda(v_1), \ldots, \Lambda(v_m))^\top \in \mathbb{F}_{q^m}^m$ を求めた後に，上記の条件を満たす $\boldsymbol{U} \in \mathbb{F}_q^{r\times m}$ を $\mathbb{F}_q$ 上ガウス消去法で求め，各 $i \in [r]$ に対し，$E_i = u_{i1}v_1 + \cdots + u_{im}v_m$ を出力すればよい．

行列 $\boldsymbol{U}$ は **Algorithm 3** によって求められる．ただし簡単のため次の仮定を置いた．行列 Λ の第 (i, j) 成分を λ_{ij} とおく．記述の簡便さのため，Λ に対し **Algorithm 3**，すなわち列基本変形による簡約化を行うと以下のような行列の形になるとする[*3]．

$$\Lambda' = \begin{pmatrix} \boldsymbol{I}_{(m-r)\times(m-r)} & \boldsymbol{0}_{(m-r)\times r} \\ \Lambda'_{[m-r+1,m],[m-r]} & \boldsymbol{0}_{r\times r} \end{pmatrix}. \tag{A.60}$$

ただし，$\boldsymbol{I}_{(m-r)\times(m-r)}$ は $m-r$ 次単位行列である．また，$\Lambda'_{[m-r+1,m],[m-r]} \in \mathbb{F}_q^{r\times(m-r)}$ は何らかの行列である．このとき，行列 $\boldsymbol{U}$ を $\left(\Lambda'_{[m-r+1,m],[m-r]}, \boldsymbol{I}_{r\times r}\right)$ とすればよい．

Algorithm 3 Step 3

Require: Λ

Ensure: $\boldsymbol{U}$

1: **for** $k \in [m-r]$ **do**
2: $\quad\lambda_{kk}^{-1}$ を計算
3: $\quad$**for** $j \in [k+1, m]$ **do**
4: $\quad\quad\lambda_{kj}\lambda_{kk}^{-1}$ を計算
5: $\quad\quad$**for** $i \in [k+1, m]$ **do**
6: $\quad\quad\quad\lambda_{ij} - \left(\lambda_{kj}\lambda_{kk}^{-1}\right)\lambda_{ik}$ を計算
7: $\quad\quad$**end for**
8: $\quad$**end for**
9: **end for**
10: $\boldsymbol{U} \coloneqq \left(\Lambda_{[m-r+1,m],[m-r]}, \boldsymbol{I}_{r\times r}\right)$

最初に，Step 3-1 の演算回数を説明する．各 $i' \in [m]$ に対し，$\Lambda(x) = \sum_{j\in[0,t]} \lambda_j x^{q^j} \in \mathbb{F}_{q^m}[x]$ が入力された下で $\Lambda(v_{i'})$ を出力するために，$\mathbb{F}_{q^m}$ 上乗算を $t+1$ 回，加算を t 回行えば求められることを示す．

$$\Lambda(v_{i'}) = \sum_{j\in[0,t]} \lambda_j v_{i'}^{q^j} = \sum_{j\in[0,t]} \lambda_j v_1^{q^{i'+j-1}} = \sum_{j\in[0,t]} \lambda_j v_{i'+j}. \tag{A.61}$$

[*3] 必要であれば，行番号の置換によって式 (A.60) の形に変形したあと，それ以降の操作を行い $\boldsymbol{U}$ に対応する行列を求め，最後に行番号の逆置換を行った行列を改めて $\boldsymbol{U}$ と置けばよい．

正規基底の元 $v_{i'}, \ldots, v_{i'+t}$ の値はすでに求められているため，$\sum_{j\in[0,t]} \lambda_j v_{i'+j}$ の値は $\mathbb{F}_{q^m}$ 上乗算を $t+1$ 回，加算を t 回行えば求められる．

次に，Step 3-2 の演算回数を説明する．**Algorithm 3** の 1 行目から 9 行目にかけての $\mathbb{F}_q$ 上四則演算回数は，以下の通りである．

$$\sum_{k=1}^{m-r}\left(1+\sum_{j=k+1}^{m}\left(1+\sum_{i=k+1}^{m}2\right)\right) \tag{A.62}$$

$$= \frac{1}{3}(m(m-1)(2m-1)-r(r-1)(2r-1)) + (m-r)+\frac{1}{2}(m(m-1)-r(r-1)). \tag{A.63}$$

よって，四則演算回数の合計の上界は，以下の通りである．

$$\frac{1}{3}(m(m-1)(2m-1)-r(r-1)(2r-1)) + (m-r)+\frac{1}{2}(m(m-1)-r(r-1)) \tag{A.64}$$

$$\leq \frac{2}{3}(m(m-1)(2m-1)-r(r-1)(2r-1)) + (2-r)(m-r)+(1-r)(m(m-1)-r(r-1)) \tag{A.65}$$

$$\leq \frac{2}{3}(m(m-1)(2m-1)-t(t-1)(2t-1))+(2-t)(m-t) + (1-t)(m(m-1)-t(t-1)). \tag{A.66}$$

早稲田大学　博士（工学）　学位申請　研究業績書

氏名：風間　皐希　印

（2022年　1月　現在）

種類別	題名、　発表・発行掲載誌名、　発表・発行年月、　連名者（申請者含む）
論文○	Gabidulin符号に基づく符号化分散計算方式とその誤り訂正能力の評価 電子情報通信学会論文誌A, Vol.J104-A, No.6, pp.156-159, 2021年6月 風間皐希, 鎌塚明, 吉田隆弘, 松嶋敏泰
論文	拡張直交配列を用いた混合水準の実験計画法に関する一考察 電子情報通信学会論文誌A, Vol.J103-A, No.1, pp.17-24, 2020年1月 山口純輝, 風間皐希, 鎌塚明, 齋藤翔太, 松嶋敏泰
講演	一般的なアクセス構造を実現する秘密分散方式を用いた行列の積計算のための秘匿符号化分散計算方式に関する一考察 第44回情報理論とその応用シンポジウム(SITA2021), 兵庫県, 2021年12月 風間皐希, 松嶋敏泰
講演	高効率なプライバシー保護情報検索システムの構成アルゴリズムの提案 日本経営工学会 2021年 春季大会, オンライン開催, 2021年5月 今津潮, 風間皐希, 松嶋敏泰
講演	クラスタごとに状態遷移確率が異なる複数の対象を同時制御するためのマルコフ決定過程 電子情報通信学会情報論的学習理論と機械学習研究会（IBISML）, オンライン開催, 2021年3月 本村勇人, 鎌塚明, 風間皐希, 松嶋敏泰
講演○	A Note on a Relationship between Smooth Locally Decodable Codes and Private Information Retrieval 2020 International Symposium on Information Theory and Its Application (ISITA), online, Oct. 2020 Koki KAZAMA, Akira KAMATSUKA, Takahiro YOSHIDA, Toshiyasu MATSUSHIMA
講演	Private Information Retrieval と Smooth Locally Decodable Codes の対応関係に関する一考察 電子情報通信学会情報理論研究会（IT）, 発表中止, 2020年3月 風間皐希, 鎌塚明, 吉田隆弘, 松嶋敏泰
講演	セキュアな再生成符号に基づく分散ストレージシステムにおける秘匿情報検索 第42回情報理論とその応用シンポジウム(SITA2019), 鹿児島県, 2019年11月 鎌塚明, 風間皐希, 吉田隆弘, 松嶋敏泰
講演	拡張直交配列を用いた混合水準の実験計画法に関する一考察 みずほ銀行・早稲田大学 学術交流協定締結1周年記念シンポジウム, 東京都, 2019年7月 山口純輝, 風間皐希, 鎌塚明, 齋藤翔太, 松嶋敏泰
講演	$(n,k,d,r,t,x,y)_{q}$ LRC符号の最小距離および次元の限界式に関する一考察 第41回情報理論とその応用シンポジウム予稿集（SITA2018）, 福島県, 2018年12月 風間皐希, 鎌塚明, 松嶋敏泰
講演	拡張直交配列を利用した多水準の実験計画法に関する一考察 第41回情報理論とその応用シンポジウム予稿集（SITA2018）, 福島県, 2018年12月 山口純輝, 風間皐希, 鎌塚明, 齋藤翔太, 松嶋敏泰
講演○	A Note on a Bound on the Rate of a Locally Recoverable Code with Multiple Recovering Sets 2018 International Symposium on Information Theory and Its Application (ISITA), Singapore, Oct. 2018 Koki KAZAMA, Akira KAMATSUKA, Takahiro YOSHIDA, Toshiyasu MATSUSHIMA
講演	On Distance Properties of (r, t, x)-LRC Codes（最新論文紹介セッション） 第7回 誤り訂正符号のワークショップ, 岩手県, 2018年9月 風間皐希, 松嶋敏泰
講演	ランク誤りを考慮したcoded computationに関する一考察 データ科学総合研究教育センター第3回シンポジウム, 東京都, 2018年7月 風間皐希, 鎌塚明, 松嶋敏泰

早稲田大学　博士（工学）　学位申請　研究業績書

氏名：風間　皐希　　印

（2022年　　1月　現在）

種類別	題名、　発表・発行掲載誌名、　発表・発行年月、　連名者（申請者含む）
講演	ランク誤りを考慮したcoded computationに関する一考察 第40回情報理論とその応用シンポジウム(SITA2017), 新潟県, 2017年11月−12月 風間皐希, 鎌塚明, 松嶋敏泰
講演	シンボルペア通信路における符号のリスト復号に関する一考察 第39回情報理論とその応用シンポジウム(SITA2016), 岐阜県, 2016年12月 風間皐希, 鎌塚明, 松嶋敏泰
講演○	A Maximum Likelihood Decoding Algorithm of Gabidulin Codes in Deterministic Network Coding 2016 International Symposium on Information Theory and Its Applications (ISITA), Monterey, California, USA, Oct.-Nov. 2016 Koki KAZAMA, Akira KAMATSUKA, Toshiyasu MATSUSHIMA
講演	A Note on Unequal Error Protection in Random Network Coding 2016 International Symposium on Information Theory and Its Applications (ISITA2016), Monterey, California, USA, Oct.-Nov. 2016 Tomohiko SAITO, Koki KAZAMA, Toshihiro NIINOMI, Toshiyasu MATSUSHIMA
講演	Subspace Unequal Error Protection Codes for Random Linear Network Coding 2016 International Symposium on Multimedia and Communication Technology, Tokyo, Japan, Aug.-Sep. 2016 Tomohiko SAITO, Koki KAZAMA, Toshihiro NIINOMI, Toshiyasu MATSUSHIMA
講演	Array-Errorモデルにおける軟判定復号に関する一考察 電子情報通信学会情報理論研究会（IT）, 大阪府, 2016年1月 風間皐希，鎌塚明, 吉田隆弘, 松嶋敏泰

謝辞

本論文の執筆に際し，皆様から非常に多くのご支援とご指導を賜りました．心より感謝の意を表します．

本論文の主査としてご指導いただいた早稲田大学基幹理工学部応用数理学科松嶋敏泰教授に心より感謝いたします．松嶋教授には，本論文を含み，9 年間に渡る研究生活の全てにおいて常に多くのご指導とご助言をいただきました．研究や教育の姿勢，そのほか公私にわたる様々なことに関しまして，数えきれないほどの多くのことをご教示賜りました．ここに，深く感謝し，心よりお礼申し上げます．

本論文の副査として大変貴重なご意見を頂きました，早稲田大学基幹理工学部応用数理学科大石進一教授，同柏木雅英教授，早稲田大学理工学術院平澤茂一名誉教授に深く感謝し，厚く御礼申し上げます．

また，研究室のゼミを通じて議論していただいた，皆様に心より御礼申し上げます．特に，横浜商科大学浮田善文教授，日本大学吉田隆弘准教授，電気通信大学八木秀樹准教授，湘南工科大学齋藤友彦准教授，早稲田大学堀井俊佑准教授には，代数的誤り訂正符号に関するゼミなどを通じて本論文や研究姿勢に関して多くのご指導，ご鞭撻をいただきました．また，東京都市大学新家稔央准教授には，分散符号化計算の提案方式に関する議論をしていただきました．特に，吉田隆弘准教授には，非常に多くの時間を割き，ゼミを通じて様々な議論をさせていただくことで，研究の説明の仕方など基本的な部分をお教えいただいたり，また，本論文の多くの部分の進展に大きくご協力いただきました．皆様に心より深くお礼申し上げます．

そして，研究室の先輩，同期として非常に親身になってくださった松嶋研の皆様には感謝申し上げます．松嶋研の学生時代およびその後においても研究全般に関することや仕事に対する姿勢など様々なことで適切かつご助言，ご協力をいただきお世話になりました青山学院大学宮希望助教，著者と同分野の研究者かつ先輩として様々なゼミを通じて議論をしてくださりさらに仕事に対する姿勢や研究生活等で公私問わず長い間大変お世話になりました湘南工科大学鎌塚明講師，分野は違えど先輩として誤り訂正符号の研究のみならず研究の姿勢や考え方などに関するご指導をいただきさらに博士論文執筆の際にもお世話になりました群馬大学齋藤翔太准教授，誤り訂正符号の

研究者かつ研究室の同期でありさらに博士論文執筆の過程を含む本論文の過程や研究生活において親身になって非常に数多くのご協力をいただきました早稲田大学中原悠太講師には心より感謝申し上げます．また，研究室の後輩であり博士後期課程への進学が決定している一條尚希氏，島田航志氏には研究生活において大変お世話になりました．また，ゼミを通じて情報セキュリティや誤り訂正符号の研究に関して議論をしてくださった先輩，同期，後輩の皆様に関しましてもお礼申し上げます．それらの議論が自身の研究へのフィードバック，時には研究成果へとつながりました．

最後に，著者の研究生活を見守り，多大な支援をしてくれた家族に心より感謝いたします．