2014年度　　修士論文

論理演算を用いて高速化した
カオス暗号の特性評価に関する研究

指導教員　鎌田 弘之　教授

明治大学大学院 理工学研究科 電気工学専攻 複合情報処理研究室

学籍番号　451R131061

氏名　田中　大貴

提出年月　2015 年 2 月

目次

第 1 章

はじめに

1.1　研究背景・目的

2014 年は，ニュース等で情報セキュリティに関する事件や問題が数多く報道され、情報セキュリティへの脅威がより増した年であったといえる．OpenSSL の脆弱性（通称:HeartBleed）や Struts の脆弱性，Bash の脆弱性（通称:ShellShock）など多くの脆弱性が報告され企業が対応に追われる一方，ID やパスワードが盗まれるといった一般生活者にとっても身近でわかりやすい事件が多発した．また，大手 SNS のように利用者のアカウントが乗っ取られプリペイドカードを買わせるという詐欺も報告されている．執筆現在では，Linux GNU C ライブラリの脆弱性（通称:GHOST）が報告されており，今後もこの傾向は続くと考えられる．通信技術の発達により，我々が様々な恩恵を受け生活はより便利なものになっているが，このようなセキュリティインシデントは未然に防ぐ必要がある．つまり安全な通信を行えるということが重要であり，それを実現するための暗号技術が必要とされている．当研究室では，従来よりカオス暗号の研究を行っている．[1] カオス暗号は大容量のデータをリアルタイムで暗号化できるという利点があるが，カオスという現象がまだ完全に定義されていないことなど，統一的な見解には至っていないという点からまだ多くの発展の余地が残されている研究領域である．研究当初のカオス暗号は，カオティックニューロン非線形要素（Chaotic Neuron Type Nonlinearity）に線形フィルタを組み合わせるものであった．その後，ボルテラフィルタの導入 [2][3] や非線形写像の改良 [4]，固定小数点演算の改良 [5][6] などが行われてきた．そして最近まで研究されていた連立ボルテラフィルタを用いる手法 [7][8] によって，鍵の数の増加とカオス性の向上が得られ，結果として暗号強度は高まったといえる．しかし，その計算の複雑さゆえに処理が遅くなってしまい，カオス暗号の利点であるリアルタイムに暗号化できるという特徴が失われてしまっているという欠点がある．本研究では，その計算処理の遅延原因となっている式内部の乗算の一部を論理演算のひとつである排他的論理和演算に置き換えた，新たなカオス暗号システムを提案し，ランダム性を落とすことなく暗号化処理の高速化を図ることを目的とする．そして，従来のカオス暗号システムと比較し，提案したカオス暗号システムの有効性を検証する．

1.2　論文の構成

本論文の構成は，以下のとおりである．

第２章

暗号の基礎知識と分類について述べる．

第３章

カオスとは何かについて述べる．

第４章

従来法と提案法のカオス暗号システム，および用いている計算手法について述べる．

第５章

システムの特性評価を行い，その結果について述べる．

第６章

まとめ，今後の展望について述べる．

第 2 章

暗号

2.1　情報セキュリティの３大要素

暗号について述べる前に，まず情報セキュリティの３大要素について述べる．情報セキュリティは ISO/IEC 27002 によって以下の３つの要素からなると定義されている．この３つの要素を満たし，維持することが重要だと考えられており，それぞれの頭文字を取って “CIA（シーアイエー）” と呼ばれている．

機密性（Confidentiality）
: 許可された者だけが情報資産にアクセス出来ることを確実にすること．許可されていない者は情報資産へのアクセスが出来ない，もしくは閲覧が出来ても情報の改ざんが出来ないこと．暗号技術を使うことにより機密性を満たすことができる．

完全性（Integrity）
: 情報資産が正しいものであるか，不正に改ざんされていないか，また正しく扱えているかという完全さを保護すること．具体的には，データが処理される過程で，データの欠落や改ざん，破壊などの異常が起きないようにすることをいう．「認証」という暗号技術を使うことで完全性を満たすことができる．

可用性（Availability）
: 許可された者が必要な時にいつでも情報資産にアクセスできること．具体的には，システムの二重化や冗長化，またはデータのバックアップなどによって，システム障害が発生しないように管理することをいう．

上述したとおり，暗号技術は機密性または完全性の確保に用いられている．また現在では，上記の３つに加えて，新たに３つの要素（責任追跡性，真正性，信頼性）が加えられ，６大要素とも呼ばれている．

2.2　暗号とは

暗号とは，情報を当事者間でのみ理解できるようにあらかじめ決めたルールに従ってまったく違うものに変換することである．一般に当事者間の情報のやり取りにおいて，当事者以外の意図しない第三者にやり取りしている情報の内容をわからないようにするために行うものである．変換する前，つまり暗号化する前の情報は “平文（ひらぶん）” と呼ばれ，変換後の情報を “暗号文” と呼ぶ．逆に暗号化された暗号文を平文に復元することを “復号化” と呼ぶ．また，当事者以外の誰かが暗号文から平文を得ようとする行為は “暗号解読”，または単に “解読” や “暗号解析” とも呼ばれ，復号とは異なった意味となる．例えば，メールのやり取りにおいて，メールの内容を暗号化することは第三者によってメール内容を読まれないようにできるため，機密性を保つことが出来る．暗号の歴史は古く，紀元前の時代から戦争において戦局を優位に導くために用いられていた．現代でも暗号技術は戦争に用いられているが，通信技術の進歩とともに暗号技術も進歩した結果，一般生活者にも広く浸透するようになった．現在，暗号は “古典暗号” と “現代暗号” の 2 種類に分類され，現代暗号はさらに “共通鍵暗号方式” と “公開鍵暗号方式” という 2 種類の暗号方式に分類されている．[9][10]

2.3　古典暗号

古典暗号は暗号アルゴリズムも鍵も秘密にして用いる暗号技術である．ここでいう鍵とは家の鍵などの物理的な意味の鍵ではなく，パスワード情報などのパラメータのことである．単に文字をずらしたり置き換える方法が最も有名であり，歴史上使われていた暗号として，シーザー暗号やヴィジュネル暗号などがあるが，暗号としては弱く，現在では通信のために使われることはない．シーザー暗号は文字をローテーションすることで解読することが出来，ヴィジュネル暗号はカシスキー攻撃によって解読することが出来る．

2.4　現代暗号

現代暗号はコンピュータを用いて発明された暗号技術である．暗号アルゴリズムは公開するが，鍵は秘密にするという特徴がある．古典暗号では文字を暗号化するものだったが，現代では扱うデータ量が膨大なために，コンピュータを使わずに暗号・復号化するのが不可能となっている．コンピュータが操作するのは 0 と 1 のビット列であり，暗号化するのもこのビット列となる．現代暗号は数学的な難しさに安全性の根拠を求めることが多い．その暗号方式には “共通鍵暗号方式” と “公開鍵暗号方式” がある．さらにこの 2 つを組み合わせた “ハイブリッド暗号方式” というのも存在する．

2.4.1　共通鍵暗号方式

共通鍵暗号方式とは暗号化と復号化に同じ鍵を用いる暗号方式であり，「対称暗号方式」や「秘密鍵暗号方式」とも呼ばれている．一般に，公開鍵暗号方式よりも処理時間が高速であるが，鍵の管理が難しいという問題がある．それは暗号化に使う鍵と復号化に使う鍵が同じであるため，情報をやりとりする相手に鍵を配送しなければならないという問題である．これを “鍵配送問題” と呼ぶ．例えば，メール内容を暗号化して暗号文を作成し，これを送信しても受信者は復号化するための鍵がないため，復号化することができない．しかし暗号文と一緒に鍵を送ってしまった場合は，第三者に暗号文と鍵を盗聴され盗聴者によってメール内容を復号化されてしまう恐れがある．このように鍵を送らなければいけないのに，鍵を送ってはいけないという問題が鍵配送問題である．また，共通鍵暗号方式は必要な鍵の数は通信する相手の数の分だけ必要であり，多数の通信相手とやり取りする場合には適していない．通信相手が n 人の場合に必要になる鍵の数は式 (2.1) で求められる．さらに，この暗号方式は暗号アルゴリズムの違いによって “ブロック暗号” と “ストリーム暗号” の 2 種類に分けられる．

$$\text{通信相手が } n \text{ 人の場合に必要な鍵の数} = {}_nC_2[\text{個}] \tag{2.1}$$

ブロック暗号

ブロック暗号はビット数のまとまりを一度に処理する暗号アルゴリズムである．通常は 64 ビットまたは 128 ビット毎の固定ビット毎に暗号化していく．このまとまりを「ブロック」と呼び，ブロックのビット数のことを「ブロック長」と呼ぶ．このため，同じ鍵を用いて暗号化した場合，同じ平文のブロックは常に同じ暗号ブロックとなる．もし平文がブロック長で割り切れない場合，満たない部分を「パディング」という特定のデータを埋め合わせる処理を行うことがある．また，長い平文を暗号化するときはブロック暗号を繰り返して平文全部を暗号化することになる．この繰り返しの方法を「モード」と呼び，暗号強度を高めるために使用する．主なモードとして ECB，CBC，CFB，OFB，CTR の 5 種類がある．ブロック暗号の代表として DES，トリプル DES，AES などが挙げられる．DES とトリプル DES のブロック長は 64 ビットであり，AES は 128 ビット，192 ビット，256 ビットのいずれかから選べる．現在，DES，及びトリプル DES は特殊なハードウェアを用いてブルートフォースアタックを行えば，現実的な時間内に解読できるため，暗号として使用することは推奨されていない．そのため，現在の主流は AES であり，WPA，WPA2，WiMAX などの通信プロトコルに採用されている．

ストリーム暗号

ストリーム暗号はデータの流れ（ストリーム）を順次処理していく暗号アルゴリズムである．通常，1 ビットあるいは 1 バイトの疑似乱数ビット列を生成する．このビット列は「キーストリーム」（keystream）と呼ばれ，暗号化処理はこのキーストリームと平文を排他的論理和で演算することにより実現される．ブロック暗号のようにパディング処理を行うことはなく，暗号文は平文と

同じデータサイズとなる．そのため，ブロック暗号よりも処理が高速である．ブロック暗号がブロック単位での処理のため，どこまで暗号化したかという内部状態を持つ必要がないのに対して，ストリーム暗号はデータを順次処理していくために内部状態を持つという特徴がある．ここで仮に，ストリーム暗号が生成する乱数が真正乱数系列であるならば，その暗号を解読するにはブルートフォースアタックしかないという安全性が保証されている．つまり，どれだけ完全に近い乱数を生成できるかがこの暗号の暗号強度を左右する．一見すると，ブロック暗号よりも有能に見えるストリーム暗号だが，安全評価に関する議論がブロック暗号よりも進んではおらず，安全性の評価方法の開発が求められているという現状がある．ストリーム暗号の代表的として，RC4 が挙げられる．これは通信プロトコルである TKIP や WEP に採用されている．またカオス暗号もこの暗号方式に分類される．

2.4.2　公開鍵暗号方式

公開鍵暗号方式とは暗号化に用いる鍵を公開する公開鍵（public key）と，復号化に用いる鍵を秘密にする秘密鍵（private key）という 2 種類の鍵を用いる暗号方式である．公開鍵と復号鍵が異なるものであることから「非対称暗号方式」とも呼ばれている．公開鍵と秘密鍵は対になっており，数学的に密接な関係があるため，公開鍵と秘密鍵を個別に作ることはできない．公開鍵から秘密鍵を解読することは現在のコンピュータを用いても膨大な時間がかかるため，事実上不可能である特徴がある．さらに，公開鍵で暗号化した暗号文は，ペアになっている秘密鍵でしか復号化できないという性質により，共通鍵暗号方式で問題であった鍵配送問題は解決される．しかし，これですべての問題が解決したかというとそうではなく，今度は入手した公開鍵が本当に正しいものであるかどうかを検証しなければならない問題が残っている．この問題を利用した man-in-the-middle 攻撃というものがある．man-in-the-middle 攻撃とは悪意のある第三者が送信者と受信者の間に入り込み，送信者に対しては受信者になりすまし，受信者に対しては送信者になりすまして，お互いが直接相互に通信できていると信じ込ませることで，やりとりしているメッセージなどを盗聴したり改ざんしたりする攻撃である．つまり，実際はそれぞれが第三者と知らずに通信していることになるため，盗聴や改ざんが可能となってしまうのである．この攻撃は公開鍵暗号方式を用いるだけでは防げず，公開鍵の証明書を用いた「認証」という行為が別途必要になる．公開鍵暗号方式における必要な鍵の数は，自分の公開鍵と秘密鍵の 2 個だけであり，n 人のユーザが通信を行う場合に必要な鍵の数は全部で $2n$ 個となる．このような公開鍵暗号の代表として，RSA 暗号やエルガマル暗号，楕円曲線暗号などがある．ではここで公開鍵暗号方式と共通鍵暗号方式のどちらがより機密性が高いのかという疑問が湧くかもしれないが，単純にどちらかが優れているということはなく，鍵のビット長によって変化する．一般に，同じビット長の鍵を用いた場合は共通鍵暗号方式の方が公開鍵暗号方式よりもブルートフォースアタックに対して強いと言われている．

2.4.3　ハイブリッド暗号方式

前述した共通鍵暗号方式には処理時間が高速であるという長所があり，公開鍵暗号方式には鍵配送問題を解決できるという長所があった．ハイブリッド暗号方式とはこの 2 つの長所を組み合わせた，効率的な暗号化通信を行う方式である．データ自体の暗号化には共通鍵暗号を使用して，大量のデータを高速に処理し，データ暗号化に用いる共通鍵には公開鍵暗号方式を用いて共通鍵を暗号化して配送することで，共通鍵が盗聴されるリスクがある鍵配送問題を解決するというものである．つまり，共通鍵暗号方式の処理速度と処理効率を維持しながら，鍵の配送という問題を解決しているという一石二鳥のような暗号方式である．ハイブリッド暗号方式は，電子メールを暗号化するプロトコルである S/MIME などに採用されている．この暗号方式を行う手順を図 2.1 に示す．ハイブリッド暗号方式によって暗号化通信を行う手段は，図 2.1 のように共通鍵を暗号化して通信相手に送信し共有する第一段階と，共通鍵を用いて暗号化したデータを送受信する第二段階に分けられる．

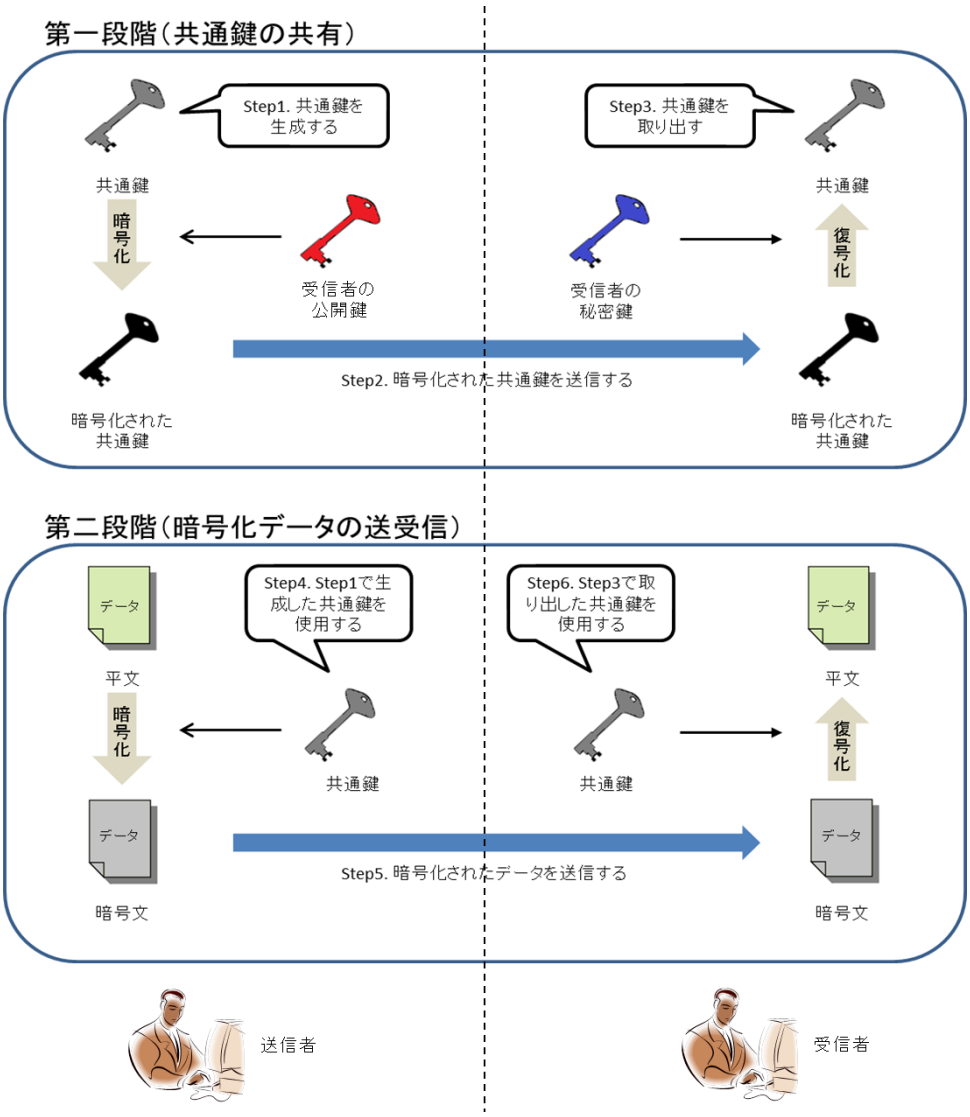

図 2.1　ハイブリッド暗号方式

第 3 章

カオス

3.1　カオスとは

カオスとは，英語で chaos と書き，そのまま日本語に訳せば，無秩序または混沌という意味になるが，ここでいうカオスとは工学分野で研究されている力学系のある現象のことをいう．決定論的な力学系に従って未来の状態が決定されるにもかかわらず，複雑な振動を生み出し，長期予測が不可能であるという特徴があることから，決定論的カオスとも言われ，本研究におけるカオスもこれを指す．カオス分野の研究は 1970 年代に始まり，カオスという現象の発見によって，それまで当たり前だったニュートン力学のように現在の状態とそのモデルがわかれば未来の予測もできるという常識が覆され，工学分野に大きな衝撃を与えた．現在までも多くの研究者がカオスを研究し，カオスの持つ性質を暗号などの分野に応用しているが，カオス自体の定義がまだ統一的な見解には至っておらず，研究者によって様々であるという現状がある．しかし，まとめると以下の５つの性質は共通している見解であるといえよう．

軌道不安定性（orbital instability）

初期変位が指数関数的に伸ばされて，最終的にはアトラクタのサイズまで拡大される．わずかな初期値の違いがその後の振る舞いを大きく変化させる性質であり，これは初期値鋭敏性とも言う．初期変位の伸び率はリアプノフ指数で定量化される．リアプノフ指数については次節で述べる．

長期予測不能性（long-term unpredictability）

軌道不安定性に深く関与しており，無限大の精度で初期状態を観測しない限り，観測誤差が指数関数的に拡大される．その結果，長期予測が事実上不可能となる．しかし，決定論的ダイナミクスには従っているので，良いモデルを作れば短期的な予測は可能である．

有界性（boundedness）

軌道不安定性のみでは，初期変位に誤差が拡大されるのみで発散する．このような現象は線形システムでも起こり得るが，無限に飛んでいくような振る舞いではない．アトラクタとして漸近安全な状態を保つには，非線形折り返しによる再帰運動によって，ある一定の

領域に存在する必要がある．この有界性という性質によって我々はカオスという現象を観測できるともいえる．

アトラクタのフラクタル性，自己相似性（self-similarity）

カオス力学系のアトラクタの幾何学的な構造は，多くの場合自己相似構造（フラクタル構造）を持つ．しかし，完全に同じ構造を繰り返すというわけではない．ここでアトラクタとは力学系における不変集合の近傍の状態点が，時間を経た後にその不変集合に引き込まれるとき，その不変集合のことをいう．自己相似性は，非整数のフラクタル次元で定量化される．

非周期性（nonperiodicity）

時系列信号として観測したときに，非周期的な挙動を示す．このようなアトラクタの軌道は決して交わることはない．

このような性質を持つカオスだが，一般的にはコンピュータを用いて検証することが多い．コンピュータが扱えるのは連続値ではなく離散的な数値であるため，デジタルで扱っている以上は，完全なカオスとは言えず疑似カオス（ディジタルカオス）ということになる．つまり我々はこのような疑似カオスをカオスとして扱っているのである．そのため，上記で述べた非周期性もデジタルで扱う以上はコンピュータが無限大の精度で実数を表現できないことから誤差が発生するため，必ず周期が出現する．

3.2　リアプノフ指数

リアプノフ指数とは，初期値に対する鋭敏な依存性を数値化する特徴量である．２次元以上の力学系については，リアプノフ指数は次元の数だけ存在し，リアプノフスペクトラムと呼ぶ．リアプノフ指数は 1889 年にロシアの数学者コワレフスカヤによって定義され，リアプノフによって一般化された．リアプノフ指数は言い換えれば，近傍点から発した軌道が時間発展とともに離れていく速さとも言える．例として，３次元力学系に初期値として微小球を与えた場合を考える．（図 3.1）最初は球であったものが，１回写像されることにより，縦方向には引き伸ばされ，横方向には押しつぶされた結果，球は楕円体へと変化する．このとき，各方向に対する指数的拡大（または縮小）率，λ_1，λ_2，λ_3 を考えることが出来る．このときの各方向の伸び（または縮み）率 λ_1，λ_2，λ_3 をリアプノフ指数，これらの組 $\{\lambda_1, \lambda_2, \lambda_3\}$ をリアプノフスペクトラムと呼ぶ．このうちの最大リアプノフ指数が正であれば，その系はカオス性があるといえる．

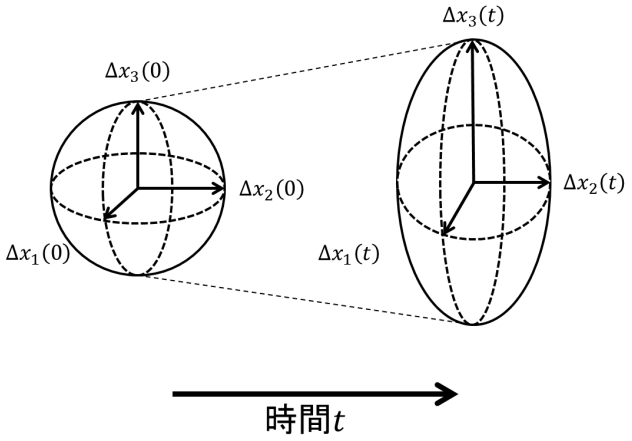

図 3.1　3 次元力学系に与えた初期微小球の時間的発展

3.3　多次元力学系におけるリアプノフスペクトラムの算出方法

この節では多次元系においてリアプノフスペクトラムを算出する方法について記述する．[11] まず，n 次元の離散力学系

$$x(t+1) = f(x(t)), x(t) \in R^n \tag{3.1}$$

を考える．ただし，$x(t)$ は n 次元空間（アトラクタ）内での離散時間 t における状態，f は n 次元非線形写像である．ここで，$x(t)$ における微小変位を $\Delta x(t)$ とすると，

$$x(t+1) + \Delta x(t+1) = f(x(t) + \Delta x(t)) \tag{3.2}$$

となる．さらにテイラー展開をし，線形近似することにより，$x(t)$ における微小変位 $\Delta x(t)$ に関する写像を得る．

$$\Delta x(t+1) = J_t \Delta x(t) \tag{3.3}$$

ここで，J_t は点 $x(t)$ における f のヤコビ行列である．f の第 i 成分を f_i，$x(t)$ の第 j 成分を x_j とすれば，ヤコビ行列は

$$J_t = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \vdots & \ddots & \vdots \\ \cdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \tag{3.4}$$

であり，J_t は時変な線形写像となる．

そして，初期変位として $\Delta x(0)$ を与え，これを J_t によって N 回写像したとすると，

$$\Delta x(N) = J_{N-1} J_{N-2} \cdots J_0 \Delta x(0) \tag{3.5}$$

となる．ここで，式（3.5）に現れるヤコビ行列の N 回積の行列を $M(x(0), N)$ と表すと，

$$M(x(0), N) = \prod_{t=0}^{N-1} J_t \tag{3.6}$$

となる．さらに，行列 $M(x(0), N)$ から次の正定値行列を

$$\Gamma(x(0), N) = [\{M(x(0), N)\}^T \{M(x(0), N)\}]^{\frac{1}{2N}} \tag{3.7}$$

と定義する．ここで T は転置行列である．

行列 $\Gamma(x(0), N)$ は正定値行列なので，この固有値は正の実数となる．この固有値を $\sigma_i(N)$ とおくと，リアプノフスペクトラムは

$$\lambda_i = \lim_{N\to\infty} \frac{1}{N} \log \sigma_i(N) \tag{3.8}$$

と定義される．

力学系が実際にわかっている場合は，リアプノフ指数の推定は比較的簡単ではあるが，それでも正定値行列の定義（式（3.7））通りに計算することは容易ではない．これは，初期変位として与えた $\Delta x(0)$ が固有値の大きな方向に引き伸ばされ，かつ小さい固有値が対応している方向にはどんどん縮んでいくため，数値計算上は安定方向（負）のリアプノフ指数の評価が困難になってしまうからである．つまり，式（3.7）の定義は数値計算には向いていないのである．そこで，実際の数値計算では，1 回写像するごとに $\Delta x(t)$ が伸びた（縮んだ）結果を正規直交化することにより，この問題を避けることにする．

まず，式（3.3）における微小変位 $\Delta x(0)$ として，n 次元接空間において，互いに直行する単位ベクトルの組 $u_1(t)$，$u_2(t)$，$\cdots$，$u_n(t)$ を与え，各ベクトルの変化をみる．すなわち，

$$e_i(t+1) = J_t u_i(t) (i = 1, 2, \cdots, n) \tag{3.9}$$

により，$e_i(t+1)(i = 1, 2, \cdots, n)$ を求める．

その結果，$e_i(t+1)$ は安定方向に押しつぶされることになるので，この問題を避けるために式（3.10）のグラムシュミットの直交化を用いて，$e_i(t+1)$ を直交化した $e_i'(t+1)$ を求め，式（3.11）の新たな正規直交系 $u_i(t+1)(i = 1, 2, \cdots, n)$ へと変換する．

$$e_i'(t+1) = e_i(t+1) - \sum_{j=1}^{i-1} (e_i(t+1) \cdot u_j(t+1)) u_j(t+1) \tag{3.10}$$

$$u_i(t+1) = \frac{e_i'(t+1)}{|\, e_i'(t+1)\,|} \tag{3.11}$$

次に，各ベクトル $u_i(t+1)(i=1,2,\cdots,n)$ を式（3.9）によって再び写像する過程を繰り返す．こうして得られる $e_i'(t)$ の系列を用いることで，リアプノフスペクトラム $\lambda_i(i=1,2,\cdots,n)$ は，

$$\lambda_i = \lim_{N\to\infty} \frac{1}{N} \sum_{t=0}^{N-1} \log | e_i'(t) | \tag{3.12}$$

として求めることができる．ここで本研究では $N = 10000$ としている．

3.4　時系列信号からのリアプノフスペクトラム推定方法

定式化された力学系では，式（3.4）のヤコビ行列 J_t を直接計算することができ，これによってリアプノフスペクトラムを求めることができる．しかし，著者らが提案している式は，有限ビット長であり，排他的論理和やオーバーフローを含むため，ダイナミクスが明らかにもかかわらずリアプノフスペクトラムが計算できない．つまり，変調式を偏微分することができないため，この J_t を直接知ることができない．そこで本研究では，佐野，澤田らにより提案された時系列信号からリアプノフ指数を推定する計算方法を用いることにする．[12] この推定方法は，ある微小球内にある任意の軌道上の点が時間 s 後にどのように変化し，微小球がどのくらい変形したのかを測ることによりヤコビアンを推定するものである．

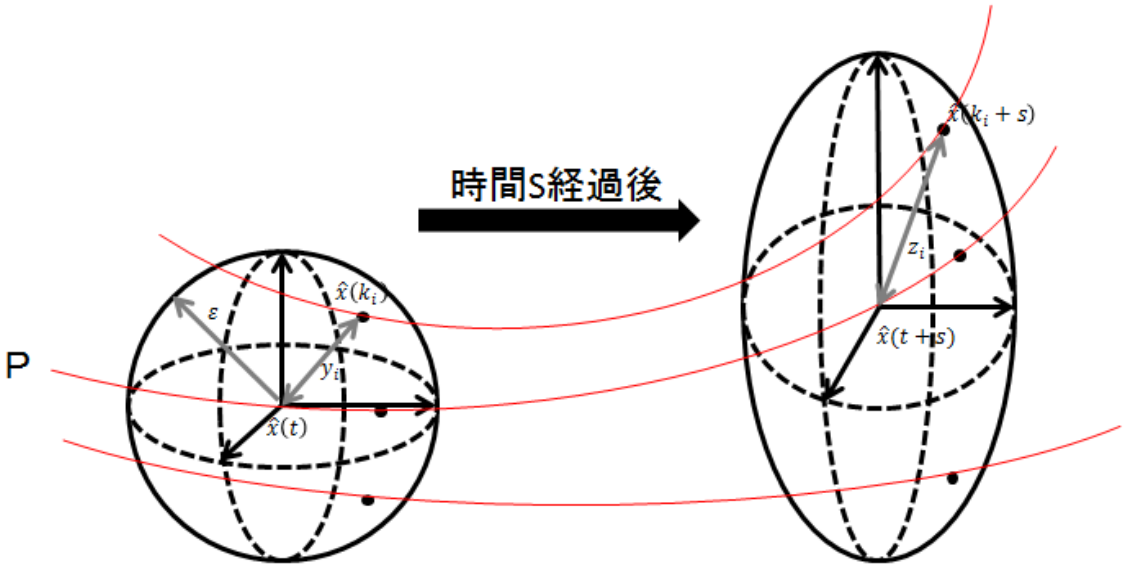

図 3.2　佐野澤田法によるヤコビアン推定の模式図（半径 ϵ 球の時間発展）

図（3.2）は佐野澤田法によるヤコビアン推定の模式図を表したものである．まずアトラクタの軌道 P 上の点を $x(t)$（時刻 t に対応）とする．この点を中心として，微小半径 ϵ の n 次元空間内の超球（以下，ϵ 球とする）を考え，これに入るアトラクタ上の他の点 $x(k_i)$ を M 個 $(i=1,2,\cdots,M)$

選び出す．このとき，$x(t)$ から見た ϵ 球内の M 個の点 $x(k_i)$ に対する変位ベクトル $y_i \in R^n$ は，

$$y_i = x(k_i) - x(t) \tag{3.13}$$

となり，これが式（3.3）の微小変位ベクトル $\Delta x(t)$ の代わりとなる．

次に，時間が s だけ経過した後を考えると，ϵ 球の中心 $x(t)$ は $x(t+s)$ に，ϵ 球内の各点 $v(k_i)$ は $v(k_i+s)$ に，各々変化する．したがって，時間 $t+s$ での変位ベクトル $z_i \in R^n$ は，

$$z_i = x(k_i + s) - x(t + s) \tag{3.14}$$

となる．

対象とする系には決定論的ダイナミクスが存在し，式（3.13），式（3.14）の変位ベクトル y_i，z_i が十分小さい，つまり，十分に小さい半径をとれると仮定する．今，ϵ 球の半径と時間 s が十分に小さいとすると，式（3.13），式（3.14）の y_i と z_i の関係は線形近似可能であり，ある行列 $A(t)$ を用いて，

$$z_i = A(t)y_i \tag{3.15}$$

と近似的に表すことが出来る．この式（3.15）の $A(t)$ は，まさに式（3.3）のヤコビ行列の近似であると考えることが出来る．ここで，式（3.15）の $A(t)$ の決定方法の一つとして，次式で表される距離，

$$S = \sum_{i=1}^{M} | z_i - A_t y_i |^2 \tag{3.16}$$

を最小にするように，即ち，最小二乗法によって $A(t)$ を決定する．
今，$A(t)$ の第 kl 成分を g_{kl} とすると，式（3.16）の距離 S の各 g_{kl} についての極小条件，

$$\frac{\partial S}{\partial g_{kl}} = 0 \tag{3.17}$$

より，

$$A(t)W = C \tag{3.18}$$

$$w_{kl} = \frac{1}{M}\sum_{i=1}^{M} y_{ik}y_{il} \tag{3.19}$$

$$c_{kl} = \frac{1}{M}\sum_{i=1}^{M} z_{ik}y_{il} \tag{3.20}$$

を得る．但し，y_{ik}，z_{ik} は，各々，ベクトル y_i，z_i の第 k 成分を表す．W，C は $m \times m$ の行列で，いわゆる分散・共分散行列である．$M \geq m$ で縮退がなければ，式（3.19），式（3.20）により $A(t)$ を一意に決定することができる．
本研究においては，このアルゴリズムにより求めた $A(t)$ を J_t として用いて，式（3.9）～（3.12）に示した手法に従い計算する．なお，本研究ではカオス暗号システムの変復調式が 3 次元であることから，3 点の近傍点を選べばよいと判断したため，M=3,s=Q10$_{max}$ × 0.03 として計算した．

3.5　アトラクタの分類

N 次元の相空間のアトラクタには N 個の相互に直交するリアプノフ指数が存在し，そのうちの最大の値のものを最大リアプノフ指数という．リアプノフ指数が正ならば，その方向には伸張し，0 ならば伸縮はなく，負であれば収縮する．このようにして求めたリアプノフスペクトラムは符号の組み合わせによってアトラクタが分類できる．
以下にその分類を示す．

カオス（chaos）
: 最大リアプノフ指数が正の場合，少なくとも 1 次元方向に対して発散傾向にあることを意味する．

ハイパーカオス（hyper chaos）
: リアプノフ指数が 2 つ以上正になる状態をハイパーカオスと呼ぶ．

リミットサイクル（limit cycle）
: 最大リアプノフ指数が 0 かつ，0 が 1 つの状態をリミットサイクル呼ぶ．

トーラス（torus）
: 最大リアプノフ指数が 0 かつ，0 が 2 つの場合，2 つの次元に対して周期を持つことから，リング状あるいは球状の平面に張り付くことを意味し，その状態をトーラスという．

固定点・不動点（fixed point）
: 最大リアプノフ指数が負である場合，すべての次元で収束傾向にあることを意味する．平衡点，零点，特異点などとも呼ばれる．

第 4 章

カオス暗号システム

この章では，従来用いられていた連立ボルテラフィルタを適用した従来のカオス暗号システムと，それを改良した提案法のカオス暗号システムについて述べるが，そのシステムに用いらている固定小数点演算と論理演算，そしてボルテラフィルタについてまず述べることにする．

4.1　固定小数点演算

まず，前提としてカオス暗号システムにおける計算は全てコンピュータ上で行われている．固定小数点演算は，整数部に使うビット数と小数部に使うビット数をあらかじめ固定する演算手法である．これに対し，少数点位置が可変で，仮数部と指数部に分けて表現する浮動小数点演算では宣言した数値の大きさにより小数点位置が移動するため広範囲の値を表現できる．一般的に固定小数点演算は浮動小数点に比べて扱える値の範囲が狭いが，計算速度が速く情報落ちが起こらない利点がある．しかし扱える値の範囲が狭いため，この演算手法で計算するとオーバーフローと呼ばれる現象が起きやすくなる．コンピュータは表現できる値の範囲が決まっており，この範囲を超えるとオーバーフローとなり，使用しているコンピュータの環境によっては最小値へと戻ったり，計算エラーが起きてしまう．一般的にシステム開発においてはこのようなオーバーフローは脆弱性を生み出す原因となり得るため，避けるべきである．しかし，われわれはあえて固定小数点演算を採用し，このオーバーフローを積極的に活用している．その理由は２つある．１つは我々はカオス暗号システムを最終的にハードウェアに実装することを考えているため，DSP（Degital Signal Processor）や FPGA（Field Programmable Grid Array）で高速処理しやすいからである．つまり，演算リソースが制限されているため，浮動小数点演算を用いては計算コストが高くなってしまい，ストリーム暗号の利点であるリアルタイム性が失われる可能性があるからである．２つ目の理由は，カオスに必要な有界性を得るためである．従来のカオス暗号システムをそのまま計算しては内部状態変数が発散してしまうのが，固定小数点演算ではオーバーフローが起きても値はある一定の範囲内に収まるため，有界性が実現される．以上の理由から我々は固定小数点演算を採用している．我々が用いているものは，上位 1 ビットを符号ビットとし，下位 10 ビットを小数部として扱う 16 ビット長固定小数点演算（以下，Q10 フォーマットと呼ぶ.）である．加減算の場合は，小数点位置の移動が起こらないため通常の演算を行えばよいが，乗除算の

場合は小数点位置の移動が起きてしまう．そこで，乗算の場合は図 4.1 に示したような小数点の位置を戻す操作が必要になる．これは 16 ビット長の変数を一度 32 ビットに拡張してから演算を行い，32 ビットの結果を得る．そして出てきた 32 ビット長の下位 10 ビットと上位 6 ビットを取り除いて 16 ビット長の演算結果とする．

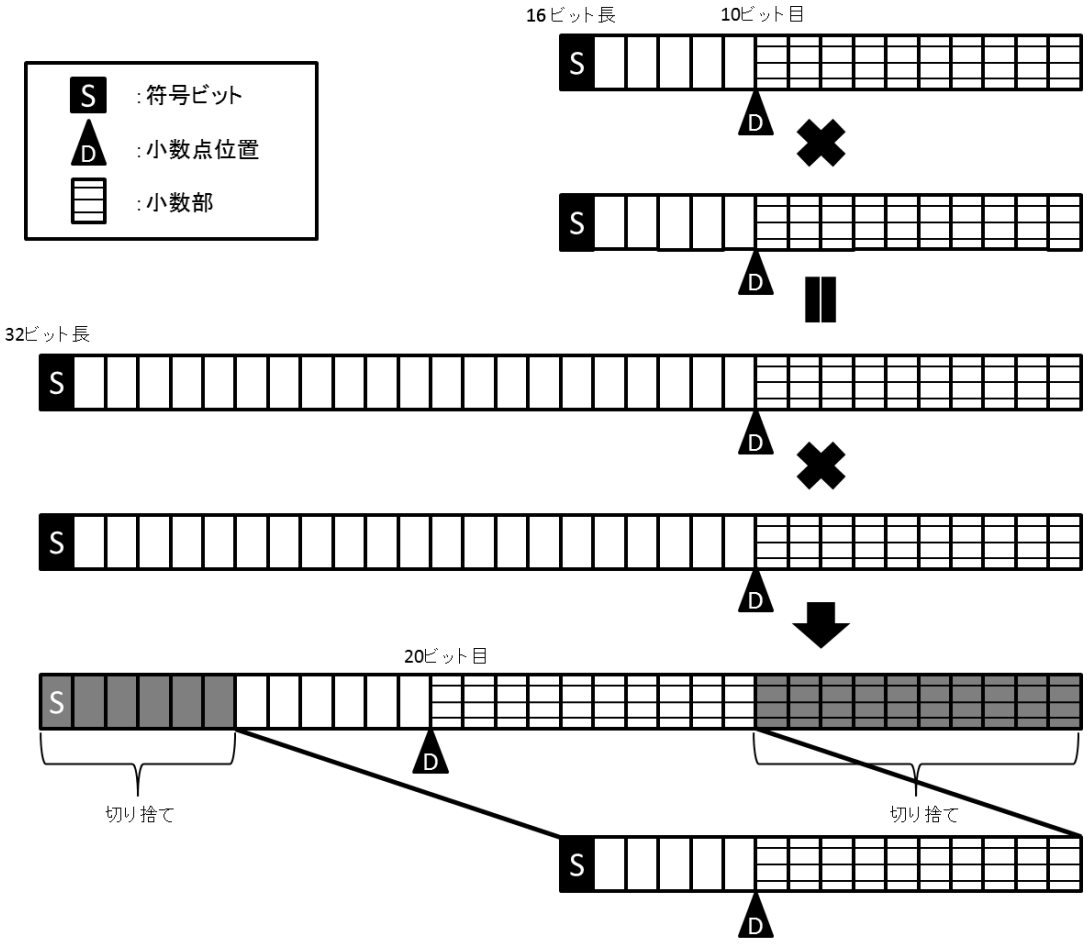

図 4.1　Q10 フォーマットによる乗算手順

除算の場合は，乗算と同じ操作をすると小数部がすべて消えてしまうため，32 ビット長への拡張時に，被除算（割られる数であり分数で言う分子）の下位 20 ビットが小数部となるように小数点の位置を移動する．この操作後に演算を行うことで下位 10 ビットを小数部とする 32 ビット長の演算結果が得られ，最後に上位 16 ビットを取り除き，16 ビット長の結果とする．この演算概要を図 4.2 に示すが，本研究に限って言えば，提案法には除算は用いていないため，この操作が行われることはない．

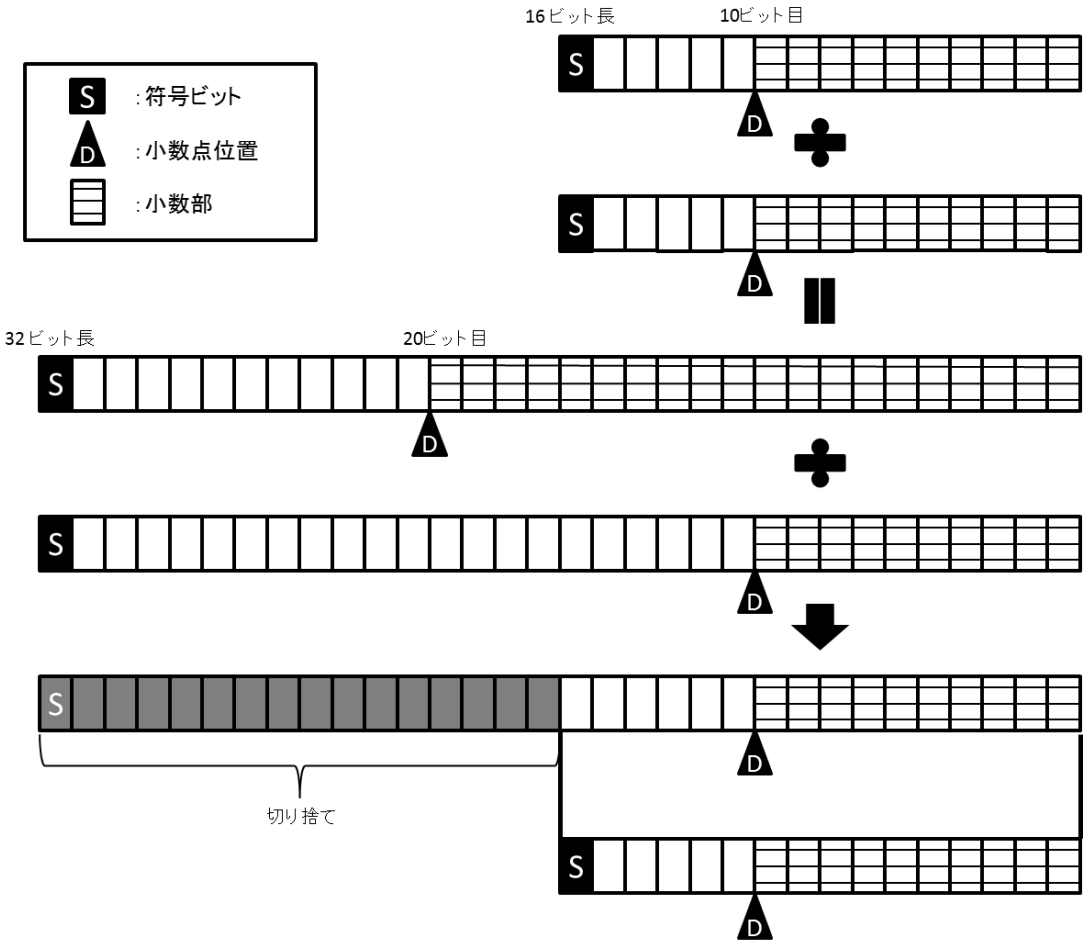

図 4.2　Q10 フォーマットによる除算手順

また，この Q10 フォーマットにおける最小単位 $Q10_{unit}$，最大値 $Q10_{max}$，最小値 $Q10_{min}$ は以下に示す通りとなる．

$$Q10_{unit} = 2^{-10} \simeq 0.000977 \tag{4.1}$$
$$Q10_{max} = 2^{16-10-1} - Q10_{unit} \simeq 31.999023 \tag{4.2}$$
$$Q10_{min} = -2^{16-10-1} \simeq -32 \tag{4.3}$$

4.2　論理演算

論理演算とは’1’（真：True）か’0’（偽：False）かの 2 通りの値しかとらない演算である．本研究では，排他的論理和を用いるため，これについて述べる．

4.2.1　排他的論理和

排他的論理和とは，XOR（exclusive OR）と呼ばれ，入力が異なるとき出力が’1’，入力が同じときは出力が’0’ となる論理演算である． 2 つの入力 A,B を XOR 演算した出力 S を表した真理値

表を表 4.1 に示す．また，排他的論理和は’⊕’ という図記号を使って表される．XOR 演算は，同じ演算を 2 回繰り返すと，元の値に戻るという性質があり，この性質は暗号にも用いられている．例えば，「$A \oplus B$」という演算結果 S に対して，「$S \oplus B$」とすると A に戻るということである．ここで，A が平文，B が暗号鍵，S が暗号文となる．このように普段の生活で使われている暗号アルゴリズムの一部に XOR 演算が使われているということは，これをカオス暗号に応用しても暗号強度を高められるということだと判断できる．

表 4.1　XOR の真理値表

A	B	S
0	0	0
0	1	1
1	0	1
1	1	0

4.3　ボルテラフィルタ

非線形の入出力関係を表現できるシステムはボルテラ級数展開によって表すことが出来，そのボルテラ級数のボルテラ核をディジタルフィルタとして捉えたものがボルテラフィルタである．例えば，2 次非線形成分までで，ボルテラ核が有限の記憶長 N を持つ場合，式（4.4）のように表すことができる．

$$y(n) = \sum_{i=0}^{N-1} h_1(i)x(n-i) + \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} h_2(i,j)x(n-i)x(n-j) \tag{4.4}$$

ここで，$x(n)$ と $y(n)$ はそれぞれ標本化された入力信号と出力信号であり，また $h_1(i)$, $h_2(i,j)$ は 1 次，2 次の離散ボルテラ核である．なお，ボルテラ核は対称性を持っており，ボルテラ核の変数 i, j の順列を任意に入れ換えることができる．例として，2 次ボルテラ核では以下の関係が成立する．

$$h_2(i,j) = h_2(j,i) \tag{4.5}$$

4.3.1　本研究におけるボルテラフィルタ

本研究において従来まで用いられているボルテラフィルタを式（4.6）に示す．ボルテラフィルタが用いられるまでは線形フィルタが用いられていたが，ボルテラフィルタの導入によって，暗号鍵のパラメータが増え，固定値であった写像の偏微分により導かれるヤコビアンを時変とすることができた．このため，暗号強度が増し，暗号を解読されるリスクが減少すると考えられる．

$$VF(x_1, x_2, x_3) = h_0 + \sum_{i=1}^{3} h_i x_i + \sum_{i=1}^{3}\sum_{j=1}^{3} h_{ij} x_i x_j + h_{123} \prod_{i=1}^{3} x_i \tag{4.6}$$

本研究では，ボルテラフィルタ内に含まれている乗算を XOR 演算に置き換えることで，計算処理の高速化を図る新たなボルテラフィルタを提案する．一般にコンピュータ上での計算は乗算よりも論理演算を用いる方が速いため，全体の処理速度は向上すると考えられる．置き換えたことにより，厳密にはボルテラフィルタとは呼べなくなってしまうが，その特性を第 5 章で評価し，有効性を検証する．

4.4　従来法のカオス暗号システム

以前のカオス暗号システムは，カオティックニューロン非線形要素 (Chaotic Neuron Type Nonlininearity）による非線形写像の式と送受信機の両側で同期をとる式から構成され，そこにボルテラフィルタを導入したり，非線形写像の改良，または不要なパラメータを削除しシステムの最適化を図るなどの研究がなされてきた．最近まで研究されていたカオス暗号システムは第 1 式と第 2 式にいくつかのパラメータを削って改良したボルテラフィルタを用いて連立させ，さらに第 1 式の 1 次遅れに非線形写像を組み込み，システムの最適化を図ったもので，本研究ではこれを従来法とする．従来法の変調システムは式（4.7）～（4.9）で構成され，式（4.10）は情報の伝送を示した式で，式（4.11）～（4.13）は復調システムである．また従来法に用いられている非線形写像の式である関数 $g(x)$ を式（4.14）に示し，非線形写像の外形を図（4.3）に示す．これらの式において，$s(n)$ は平文信号であり，暗号文として送信するものは $x_1(n)$ である．そして $r(n)$ は復調信号である．また，$x_1(n)$，$x_2(n)$，$x_3(n)$ はシステムの内部状態変数であり，g_0～g_{456}，h_{11}～h_{66} は暗号鍵となるパラメータである．

- 変調システム

$$x_1(n) = s(n) + \sum_{i=1}^{3}\sum_{j=1}^{3} g_{ij} x_i(n-1) x_j(n-1) + g_{123} \prod_{i=1}^{3} x_i(n-1) - g(g_0 + x_1(n-1)) \tag{4.7}$$

$$x_2(n) = \sum_{i=1}^{3}\sum_{j=1}^{3} h_{ij} x_i(n-1) x_j(n-1) \tag{4.8}$$

$$x_3(n) = x_2(n-1) \tag{4.9}$$

- 情報伝送式

$$x_4(n) = x_1(n) \tag{4.10}$$

- 復調システム

$$r(n) = x_4(n) - \sum_{i=4}^{6}\sum_{j=4}^{6} g_{ij} x_i(n-1) x_j(n-1) - g_{456} \prod_{i=4}^{6} x_i(n-1) + g(g_0 + x_4(n-1)) \tag{4.11}$$

$$x_5(n) = \sum_{i=4}^{6} \sum_{j=4}^{6} h_{ij} x_i(n-1) x_j(n-1) \tag{4.12}$$

$$x_6(n) = x_5(n-1) \tag{4.13}$$

- 非線形写像

$$g(x) = \begin{cases} \frac{(Min-Max)x+(Max-a_1)Min}{Min-a_1} & x_{min} \le x \le a_1 \\ \frac{(Max-Min)x+(a_1 Min-a_2 Max)}{a_1-a_2} & a_1 \le x \le a_2 \\ \frac{Max \times x+a_2 \times Max}{a_2-a_3} & a_2 \le x \le a_3 \\ \frac{(Min-Max)x+(a_3 Max-a_4 Min)}{a_3-a_4} & a_3 \le x \le a_4 \\ \frac{Min \times x+a_2 \times Min}{a_4-a_5} & a_4 \le x \le a_5 \\ \frac{(Min-Max)x+(a_5 Max-a_6 Min)}{a_5-a_6} & a_5 \le x \le a_6 \\ \frac{(Max-Min)x+(a_6 Min-Max^2)}{a_6-Max} & a_6 \le x \le x_{max} \end{cases} \tag{4.14}$$

なお，$g_{11} = g_{44}$，$g_{12} = g_{45}$，$g_{13} = g_{46}$，$g_{21} = g_{54}$，$g_{22} = g_{55}$，$g_{23} = g_{56}$，$g_{31} = g_{64}$，$g_{32} = g_{65}$，$g_{33} = g_{66}$，$g_{123} = g_{456}$，$h_{11} = h_{44}$，$h_{12} = h_{45}$，$h_{13} = h_{46}$，$h_{21} = h_{54}$，$h_{22} = h_{55}$，$h_{23} = h_{56}$，$h_{31} = h_{64}$，$h_{32} = h_{65}$，$h_{33} = h_{66}$，$x_1(n) = x_4(n)$，$x_2(n) = x_5(n)$，$x_3(n) = x_6(n)$ である．

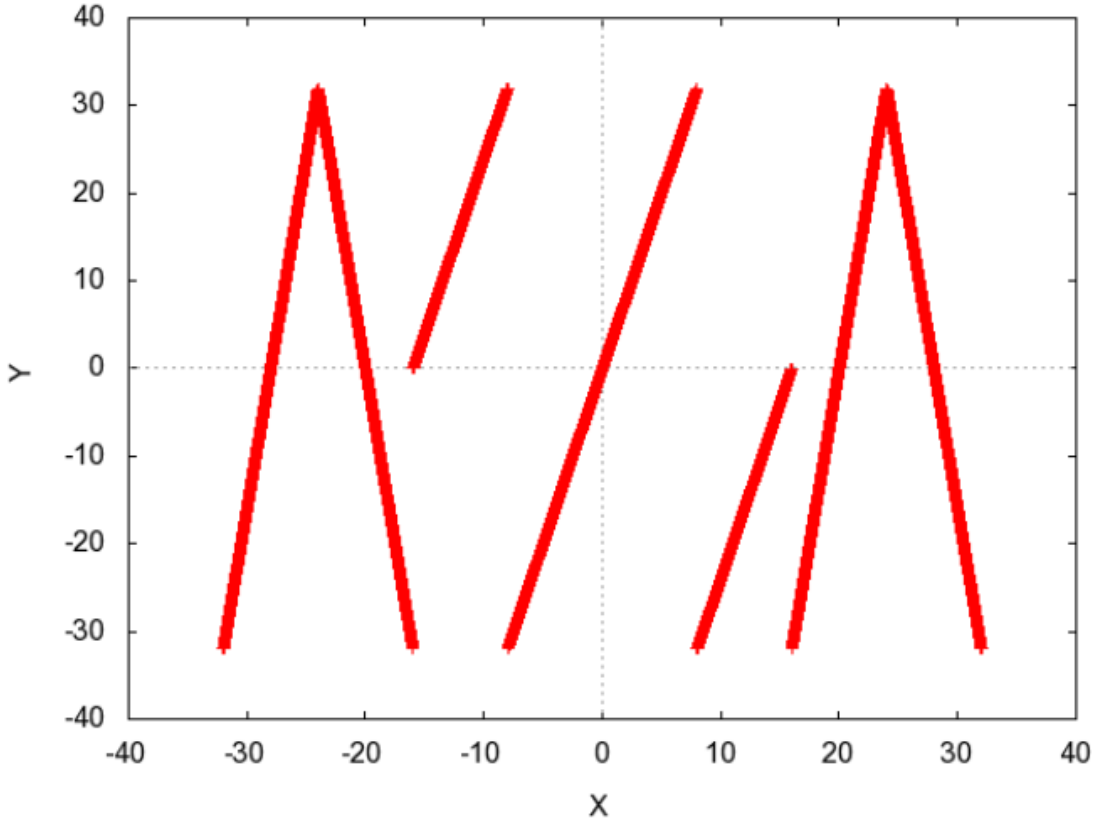

図 4.3　非線形写像の外形

4.5　提案法のカオス暗号システム

本研究では，従来法に用いられているボルテラフィルタ内の乗算を XOR 演算に置き換えることで，処理速度の向上を図った新たなシステムを提案する．さらに，暗号鍵のビット数を増やすため，パラメータ $g_1, g_2, g_3, h_1, h_2, h_3, h_{123}, \theta_1, \theta_2$ を新たに増やした．また，従来用いられていた非線形写像は計算コストが高いことから提案法では用いないこととする．ここで，すべてのパラメータを XOR 演算したものは，後述する DIEHARD TEST によってランダム性がないことがわかっている．そこで，各パラメータを一つずつ XOR 演算したシステムを DIEHARDTEST の結果が良かったパラメータだけを採用し，組み合わせたものが提案法である．提案法の変調システムを式（4.15）～（4.17）に，情報伝送の式を式（4.18）に，復調システムを式（4.19）～（4.21）に示す．これらの式において，$s(n)$ は平文信号であり，暗号文として送信するのは $x_1(n)$ となる．そして $r(n)$ は復調信号である．また，$x_1(n)$，$x_2(n)$，$x_3(n)$ はシステムの内部状態変数であり，g_1～g_{456}，h_1～h_{456}，θ_1，θ_2 は暗号鍵となるパラメータである．

- 変調システム

$$\begin{aligned} x_1(n) = {} & s(n) + \{g_1 x_1(n-1)\} + \{g_2 x_2(n-1)\} + \{g_3 \oplus x_3(n-1)\} \\ & + \{g_{11} \oplus x_1(n-1) \oplus x_1(n-1)\} + \{g_{12} \oplus x_1(n-1) \oplus x_2(n-1)\} + \{g_{13} \oplus x_1(n-1) \oplus x_3(n-1)\} \\ & + \{g_{21} \oplus x_2(n-1) \oplus x_1(n-1)\} + \{g_{22} x_2(n-1)^2\} + \{g_{23} \oplus x_2(n-1) \oplus x_3(n-1)\} \\ & + \{g_{31} x_3(n-1) x_1(n-1)\} + \{g_{32} \oplus x_3(n-1) \oplus x_2(n-1)\} + \{g_{33} x_3(n-1)^2\} \\ & + \{g_{123} x_1(n-1) x_2(n-1) x_3(n-1)\} + \theta_1 \end{aligned} \tag{4.15}$$

$$\begin{aligned} x_2(n) = {} & \{h_1 x_1(n-1)\} + \{h_2 x_2(n-1)\} + \{h_3 \oplus x_3(n-1)\} \\ & + \{h_{11} \oplus x_1(n-1) \oplus x_1(n-1)\} + \{h_{12} \oplus x_1(n-1) \oplus x_2(n-1)\} + \{h_{13} \oplus x_1(n-1) \oplus x_3(n-1)\} \\ & + \{h_{21} x_2(n-1) x_1(n-1)\} + \{h_{22} x_2(n-1)^2\} + \{h_{23} x_2(n-1) x_3(n-1)\} \\ & + \{h_{31} \oplus x_3(n-1) \oplus x_1(n-1)\} + \{h_{32} x_3(n-1) x_2(n-1)\} + \{h_{33} \oplus x_3(n-1) \oplus x_3(n-1)\} \\ & + \{h_{123} \oplus x_1(n-1) \oplus x_2(n-1) \oplus x_3(n-1)\} + \theta_2 \end{aligned} \tag{4.16}$$

$$x_3(n) = x_2(n-1) \tag{4.17}$$

- 情報伝送式

$$x_4(n) = x_1(n) \tag{4.18}$$

- 復調システム

$$\begin{aligned} r(n) = {} & x_4(n) - \{g_4 x_4(n-1)\} - \{g_5 x_5(n-1)\} - \{g_6 \oplus x_6(n-1)\} \\ & - \{g_{44} \oplus x_4(n-1) \oplus x_4(n-1)\} - \{g_{45} \oplus x_4(n-1) \oplus x_5(n-1)\} - \{g_{46} \oplus x_4(n-1) \oplus x_6(n-1)\} \\ & - \{g_{54} \oplus x_5(n-1) \oplus x_4(n-1)\} - \{g_{55} x_5(n-1)^2\} - \{g_{56} \oplus x_5(n-1) \oplus x_6(n-1)\} \\ & - \{g_{64} x_6(n-1) x_4(n-1)\} - \{g_{65} \oplus x_6(n-1) \oplus x_5(n-1)\} - \{g_{66} x_6(n-1)^2\} \\ & - \{g_{456} x_4(n-1) x_5(n-1) x_6(n-1)\} - \theta_1 \end{aligned} \tag{4.19}$$

$$
\begin{aligned}
x_5(n) = {} & \{h_4 x_4(n-1)\} + \{h_5 x_5(n-1)\} + \{h_6 \oplus x_6(n-1)\} \\
& + \{h_{44} \oplus x_4(n-1) \oplus x_4(n-1)\} + \{h_{45} \oplus x_4(n-1) \oplus x_5(n-1)\} + \{h_{46} \oplus x_4(n-1) \oplus x_6(n-1)\} \\
& + \{h_{54} x_5(n-1) x_4(n-1)\} + \{h_{55} x_5(n-1)^2\} + \{h_{56} x_5(n-1) x_6(n-1)\} \\
& + \{h_{64} \oplus x_6(n-1) \oplus x_4(n-1)\} + \{h_{65} x_6(n-1) x_5(n-1)\} + \{h_{66} \oplus x_6(n-1) \oplus x_6(n-1)\} \\
& + \{h_{456} \oplus x_4(n-1) \oplus x_5(n-1) \oplus x_6(n-1)\} + \theta_2
\end{aligned} \tag{4.20}
$$

$$
x_6(n) = x_5(n-1) \tag{4.21}
$$

なお，$g_1 = g_4$，$g_2 = g_5$，$g_3 = g_6$，$g_{11} = g_{44}$，$g_{12} = g_{45}$，$g_{13} = g_{46}$，$g_{21} = g_{54}$，$g_{22} = g_{55}$，$g_{23} = g_{56}$，$g_{31} = g_{64}$，$g_{32} = g_{65}$，$g_{33} = g_{66}$，$g_{123} = g_{456}$，$h_1 = h_4$，$h_2 = h_5$，$h_3 = h_6$，$h_{11} = h_{44}$，$h_{12} = h_{45}$，$h_{13} = h_{46}$，$h_{21} = h_{54}$，$h_{22} = h_{55}$，$h_{23} = h_{56}$，$h_{31} = h_{64}$，$h_{32} = h_{65}$，$h_{33} = h_{66}$，$h_{123} = h_{456}$，$x_1(n) = x_4(n)$，$x_2(n) = x_5(n)$，$x_3(n) = x_6(n)$ である．

第 5 章

検証

本章では，提案したカオス暗号システムを実際に運用するうえで必要になるであろう暗号強度を以下の４つの項目で特性を評価する．

カオス性の検証

本研究で提案した暗号システムがカオス性を有しているかどうかをリアプノフスペクトラムを求めることで検証する．リアプノフ指数が高ければ，わずかな初期値の差で素早く異なった軌道に入ることになり，暗号鍵が解読されるリスクが少なくなると考えられる．

ランダム性の検証

暗号システムが生成した信号（暗号文）に規則性が見られた場合，それは暗号鍵を解読されるリスクへと繋がってしまう．生成した信号がどれだけ規則性がないか（乱数に近い，または周期がとても長い）を DIEHARD TEST と呼ばれる乱数検定により，暗号システムのランダム性を検証する．

係数感度の検証

パラメータが正しい値以外で復調できてしまった場合，そのパラメータは暗号鍵として適切ではない．暗号鍵としてふさわしいパラメータがどれだけあるか，パラメータミスマッチングによって判別し，検証する．

暗号化処理速度の検証

本研究の最大の目的である暗号化処理速度を測定し，検証する．

また，これらの検証において設定したパラメータの値を以下に示す．また，入力信号や内部状態変数の初期値はともに 1.0 とした．これらの初期値は特に記載しない限り変更しないものとする．

- 従来法

$$g_0 = 1.1, g_{11} = 1.1, g_{12} = 1.1, g_{13} = 1.1, g_{21} = 1.1, g_{22} = 1.1, g_{23} = 1.1, g_{31} = 1.1,$$
$$g_{32} = 1.1, g_{33} = 1.1, g_{123} = 1.1, h_{11} = 1.1, h_{12} = 1.1, h_{13} = 1.1, h_{21} = 1.1, h_{22} = 1.1,$$
$$h_{23} = 1.1, h_{31} = 1.1, h_{32} = 1.1, h_{33} = 1.1 \tag{5.1}$$

- 提案法

$$\theta_1 = 0.1, \theta_2 = 0.1, g_1 = 1.1, g_2 = 1.1, g_3 = 1.1, g_{11} = 1.1, g_{12} = 1.1, g_{13} = 1.1,$$
$$g_{21} = 1.1, g_{22} = 1.1, g_{23} = 1.1, g_{31} = 1.1, g_{32} = 1.1, g_{33} = 1.1, g_{123} = 1.1, h_1 = 1.1,$$
$$h_2 = 1.1, h_3 = 1.1, h_{11} = 1.1, h_{12} = 1.1, h_{13} = 1.1, h_{21} = 1.1, h_{22} = 1.1, h_{23} = 1.1,$$
$$h_{31} = 1.1, h_{32} = 1.1, h_{33} = 1.1, h_{123} = 1.1 \tag{5.2}$$

5.1　カオス性の検証

リアプノフスペクトラムを求めることにより，提案法のカオス性の有無を調べる．リアプノフ指数が 1 つ正であるならば，カオス性を有していると判断でき，２つ正であるならばハイパーカオスであるといえる．また，リアプノフ指数が高い値を示せば，わずかな初期値の違いですぐに異なる軌道に入るため，軌道予測が困難になり，暗号鍵が解読されるリスクが少なくなると考えられる．以下に従来法，提案法の各パラメータを 0.1 ずつ変化させた場合のリアプノフスペクトラムを図（5.1）～（5.48）に示す．赤線が第 1 リアプノフ指数，緑線が第 2 リアプノフ指数，青線が第 3 リアプノフ指数である．

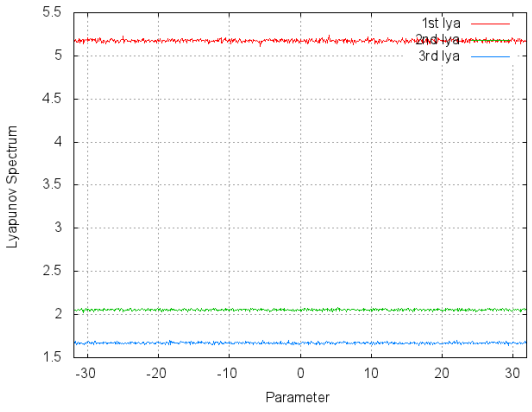

図 5.1　従来法 g_0 のリアプノフスペクトラム

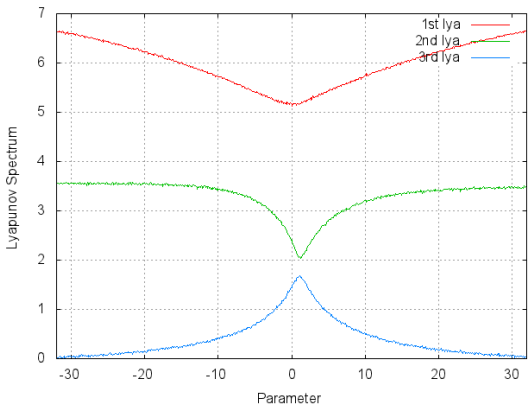

図 5.2　従来法 g_{11} のリアプノフスペクトラム

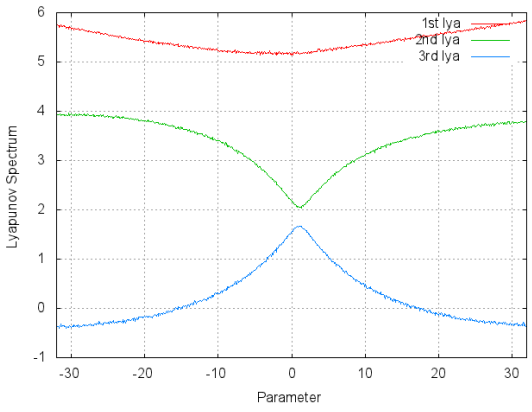

図 5.3　従来法 g_{12} のリアプノフスペクトラム

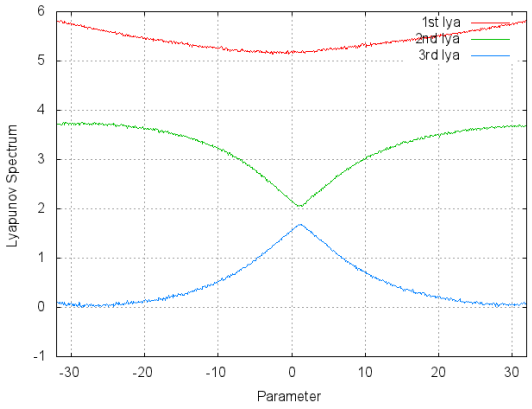

図 5.4　従来法 g_{13} のリアプノフスペクトラム

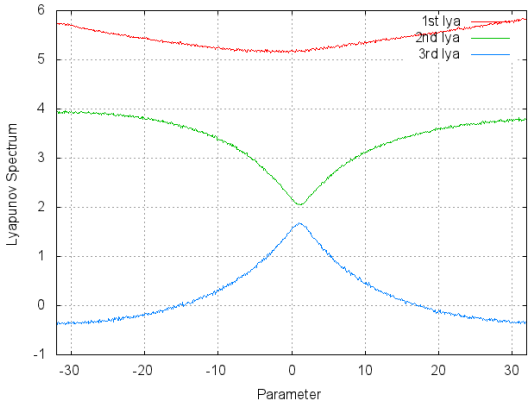

図 5.5　従来法 g_{21} のリアプノフスペクトラム

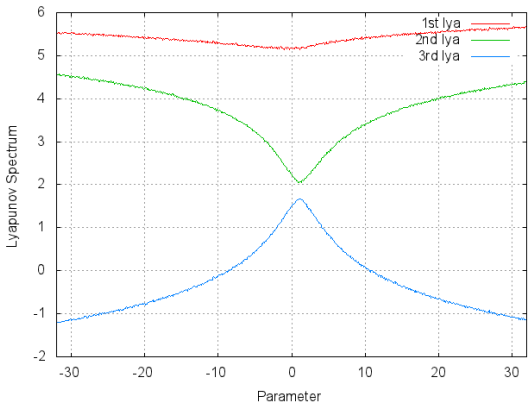

図 5.6　従来法 g_{22} のリアプノフスペクトラム

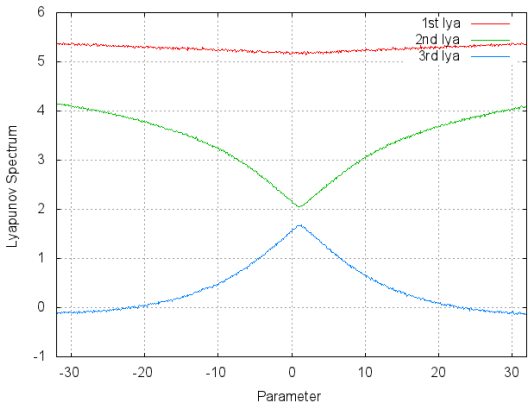

図 5.7　従来法 g_{23} のリアプノフスペクトラム

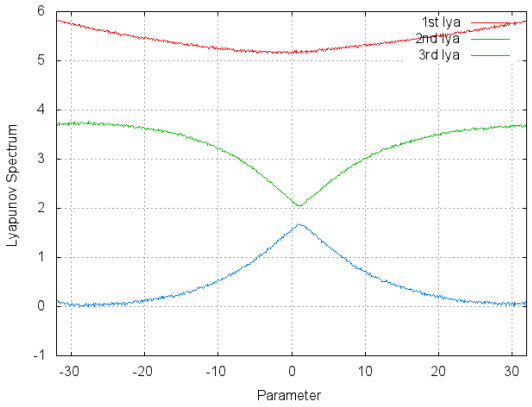

図 5.8　従来法 g_{31} のリアプノフスペクトラム

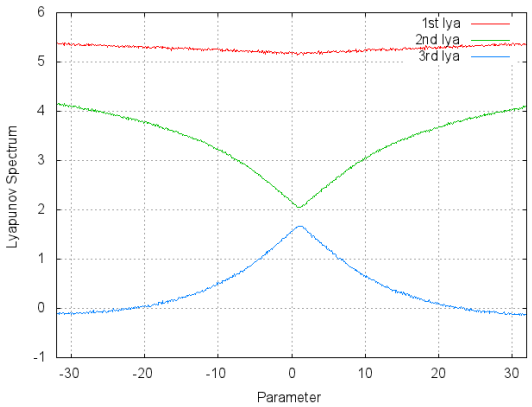

図 5.9　従来法 g_{32} のリアプノフスペクトラム

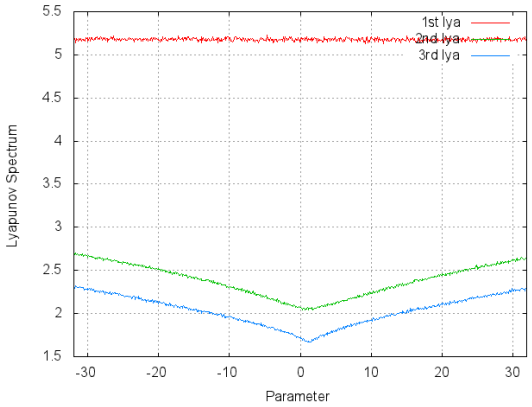

図 5.10　従来法 g_{33} のリアプノフスペクトラム

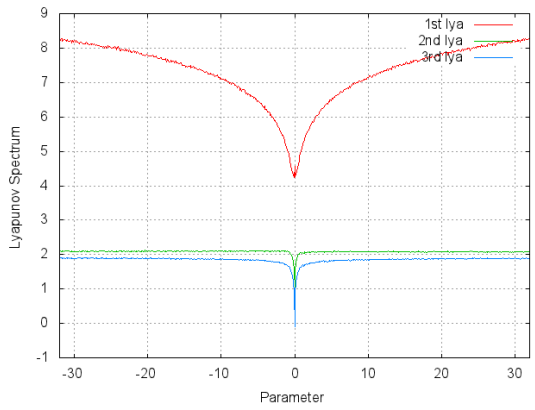

図 5.11　従来法 g_{123} のリアプノフスペクトラム

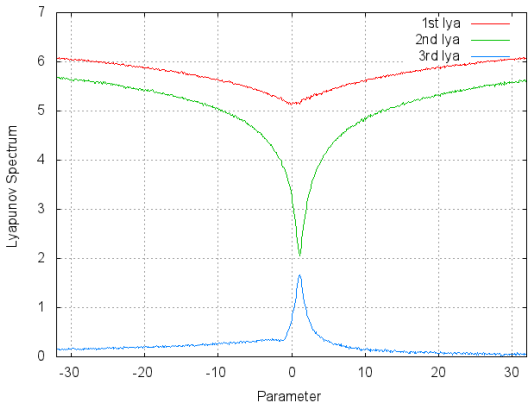

図 5.12　従来法 h_{11} のリアプノフスペクトラム

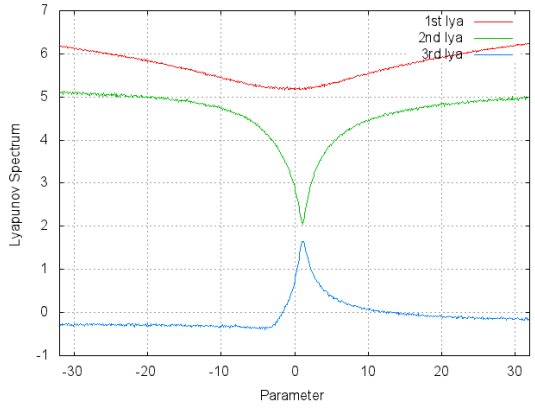

図 5.13　従来法 h_{12} のリアプノフスペクトラム

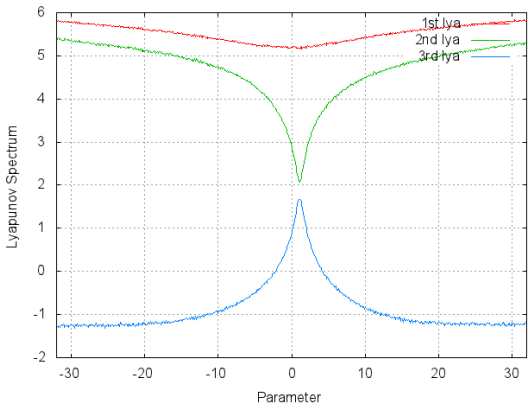

図 5.14　従来法 h_{13} のリアプノフスペクトラム

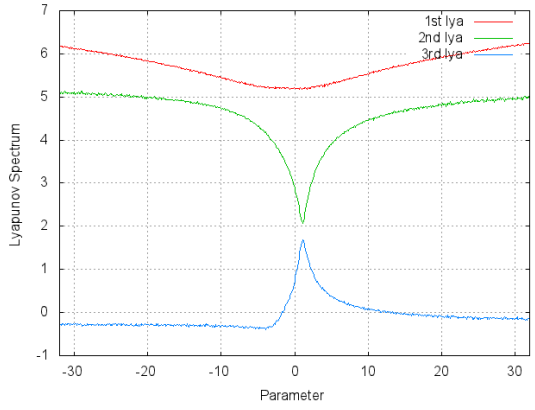

図 5.15　従来法 h_{21} のリアプノフスペクトラム

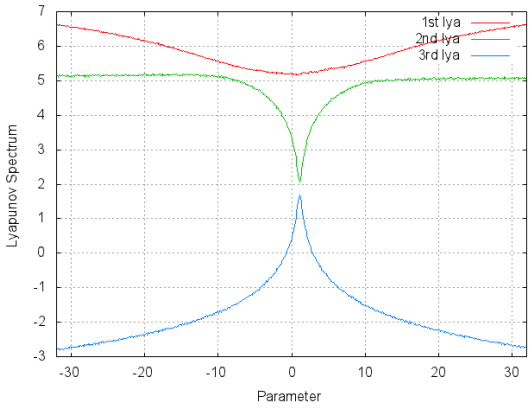

図 5.16　従来法 h_{22} のリアプノフスペクトラム

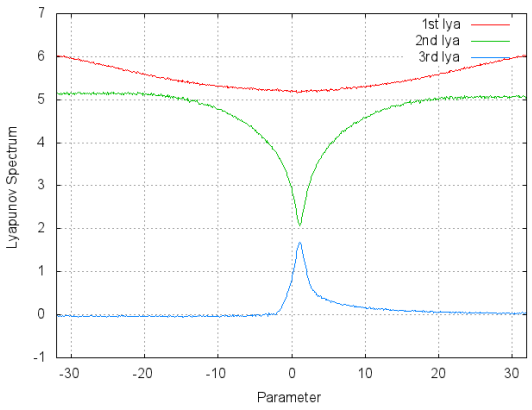

図 5.17　従来法 h_{23} のリアプノフスペクトラム

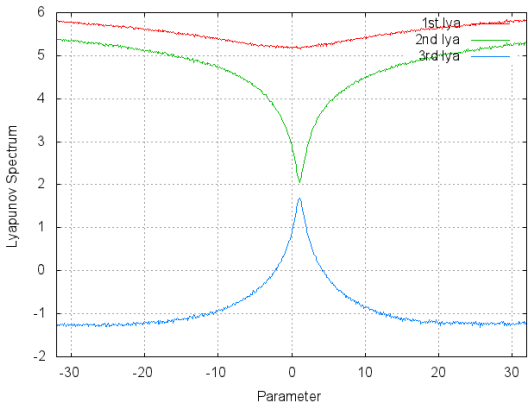

図 5.18　従来法 h_{31} のリアプノフスペクトラム

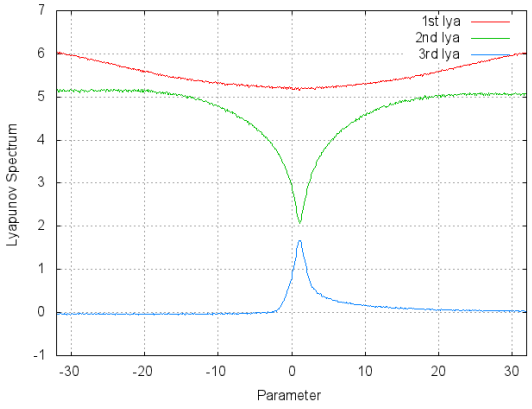

図 5.19　従来法 h_{32} のリアプノフスペクトラム

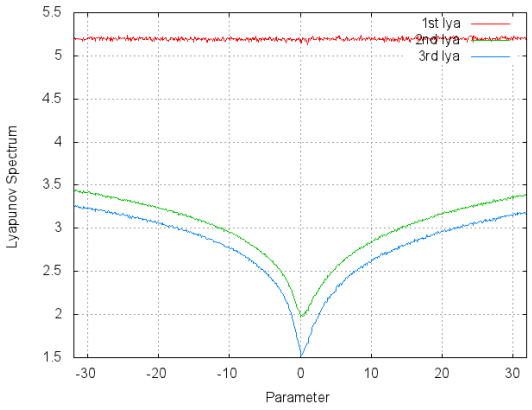

図 5.20　従来法 h_{33} のリアプノフスペクトラム

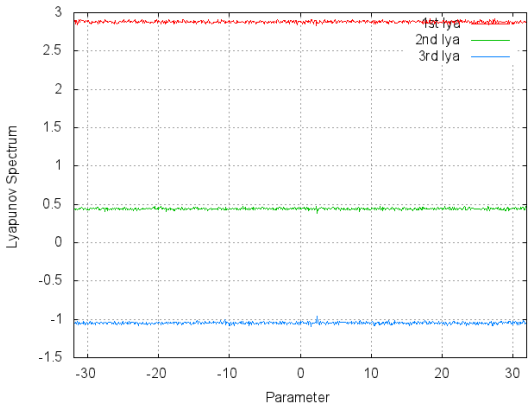

図 5.21　提案法 θ_1 のリアプノフスペクトラム

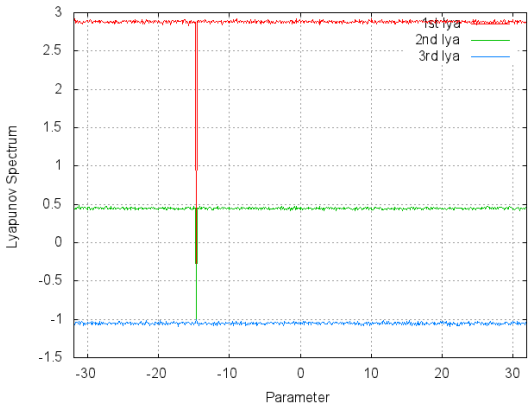

図 5.22　提案法 g_1 のリアプノフスペクトラム

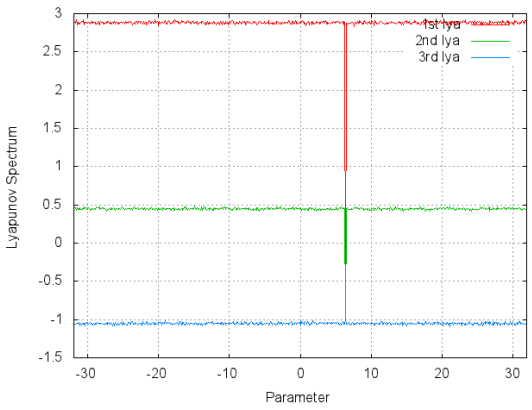

図 5.23　提案法 g_2 のリアプノフスペクトラム

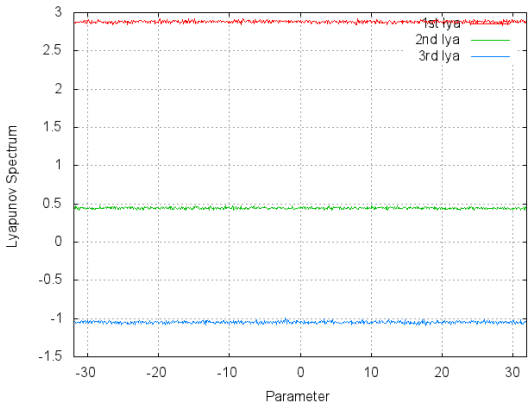

図 5.24　提案法 g_3 のリアプノフスペクトラム

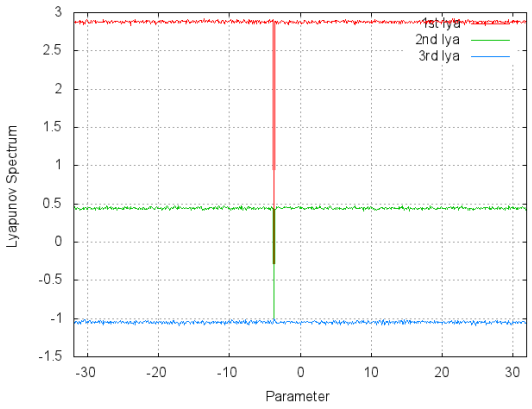

図 5.25　提案法 g_{11} のリアプノフスペクトラム

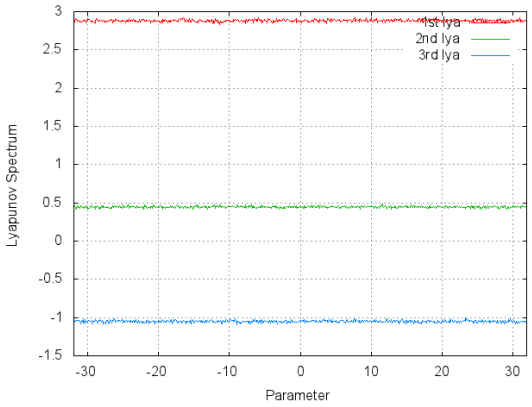

図 5.26　提案法 g_{12} のリアプノフスペクトラム

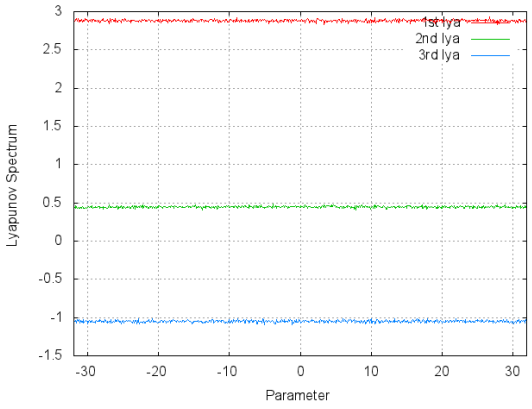

図 5.27　提案法 g_{13} のリアプノフスペクトラム

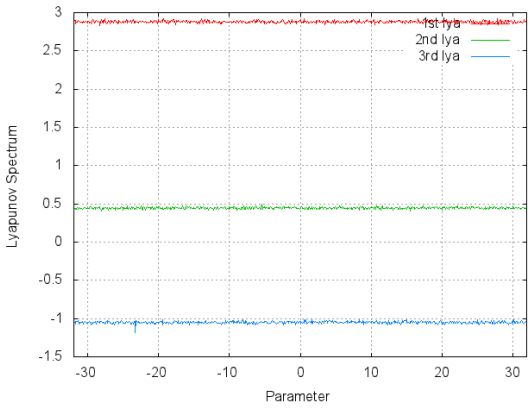

図 5.28　提案法 g_{21} のリアプノフスペクトラム

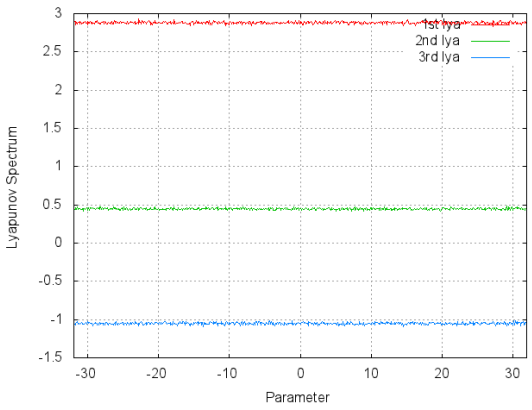

図 5.29　提案法 g_{22} のリアプノフスペクトラム

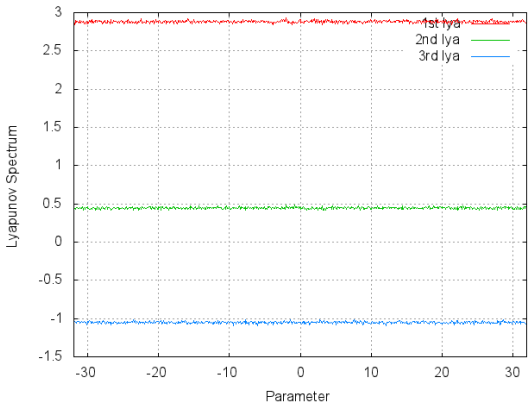

図 5.30　提案法 g_{23} のリアプノフスペクトラム

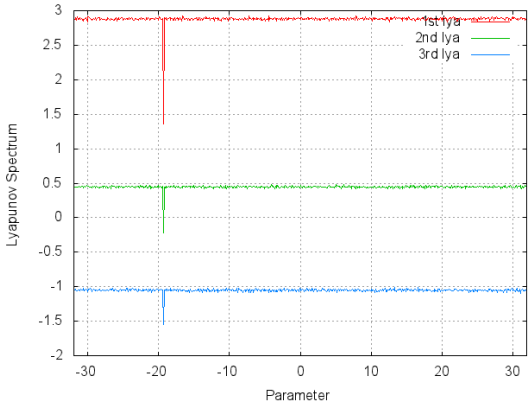

図 5.31　提案法 g_{31} のリアプノフスペクトラム

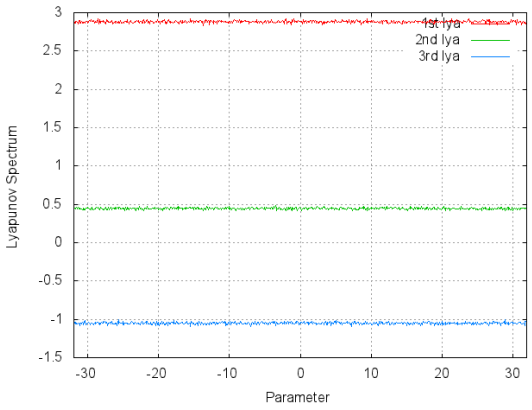

図 5.32　提案法 g_{32} のリアプノフスペクトラム

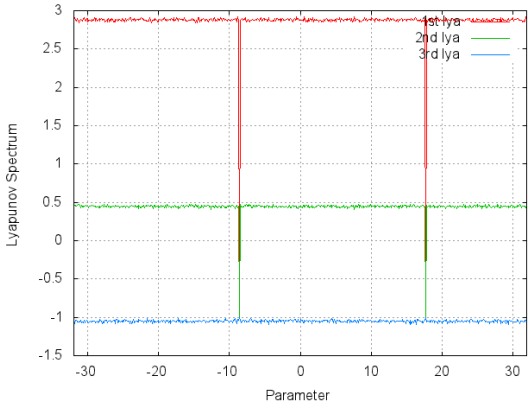

図 5.33　提案法 g_{33} のリアプノフスペクトラム

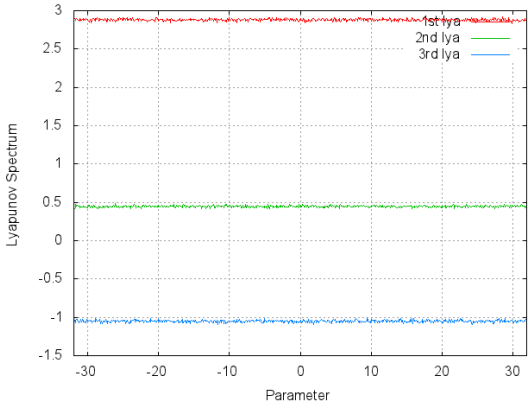

図 5.34　提案法 g_{123} のリアプノフスペクトラム

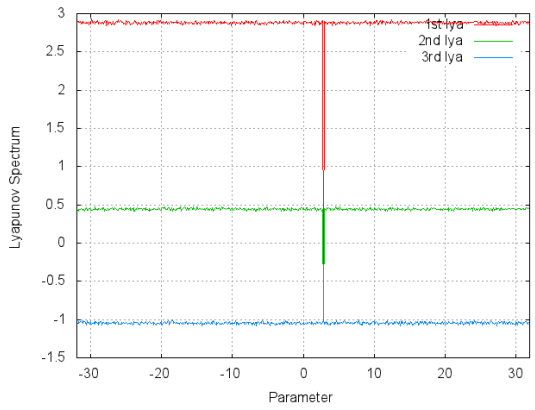

図 5.35　提案法 θ_2 のリアプノフスペクトラム

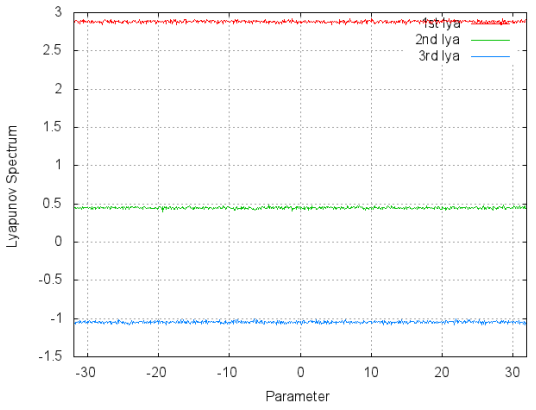

図 5.36　提案法 h_1 のリアプノフスペクトラム

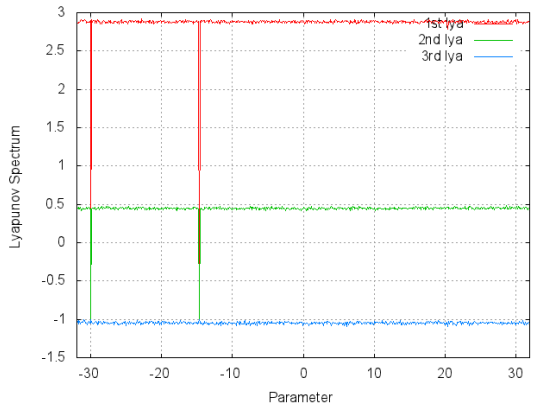

図 5.37　提案法 h_2 のリアプノフスペクトラム

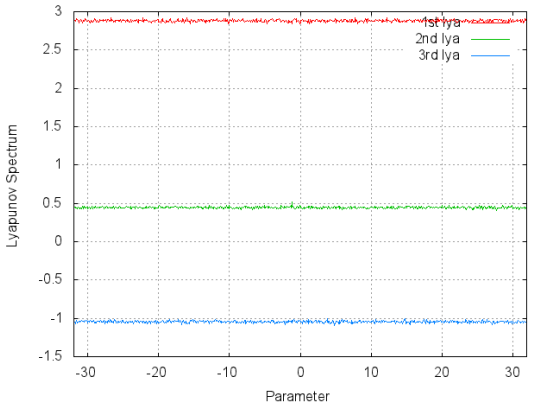

図 5.38　提案法 h_3 のリアプノフスペクトラム

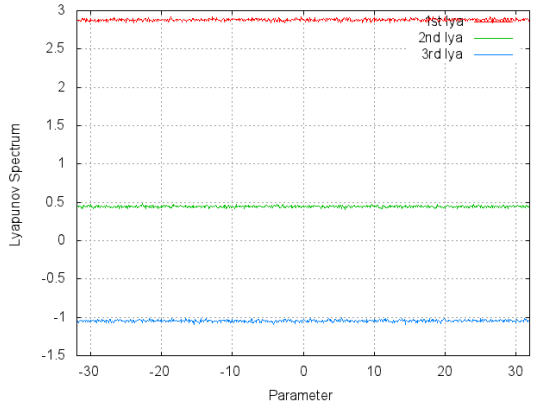

図 5.39　提案法 h_{11} のリアプノフスペクトラム

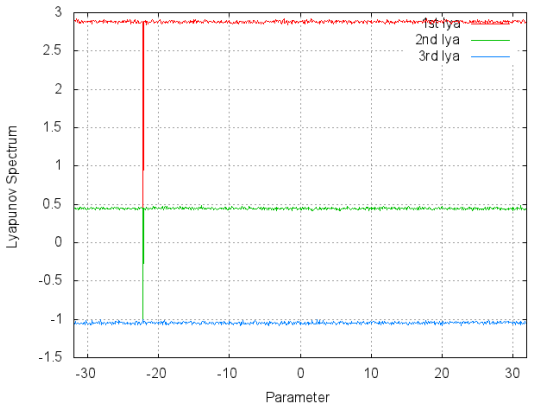

図 5.40　提案法 h_{12} のリアプノフスペクトラム

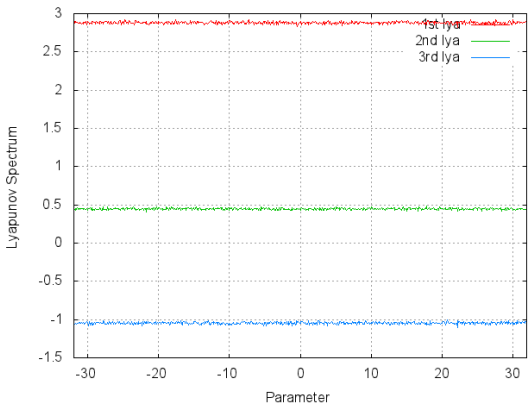

図 5.41　提案法 h_{13} のリアプノフスペクトラム

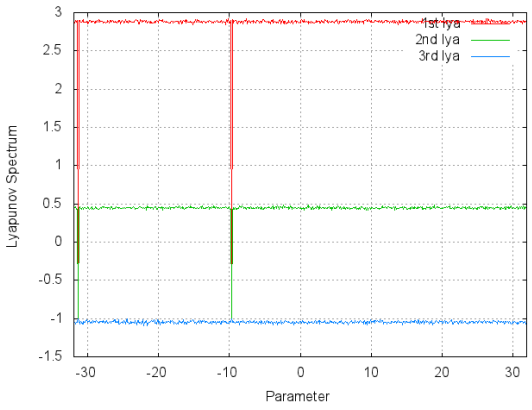

図 5.42　提案法 h_{21} のリアプノフスペクトラム

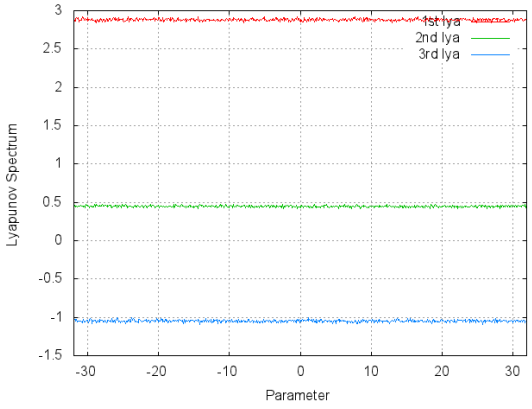

図 5.43　提案法 h_{22} のリアプノフスペクトラム

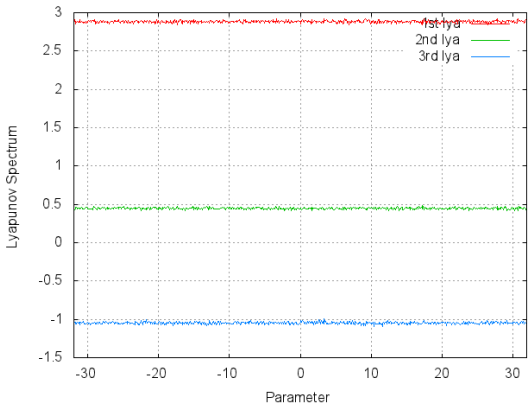

図 5.44　提案法 h_{23} のリアプノフスペクトラム

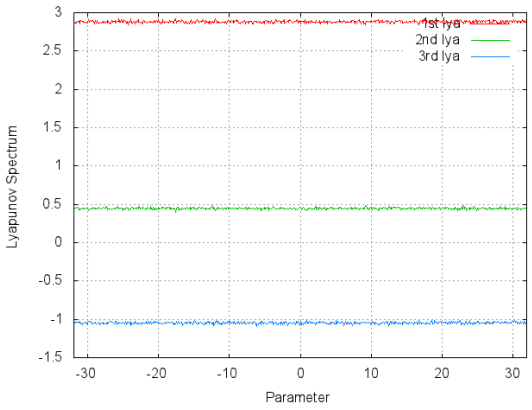

図 5.45　提案法 h_{31} のリアプノフスペクトラム

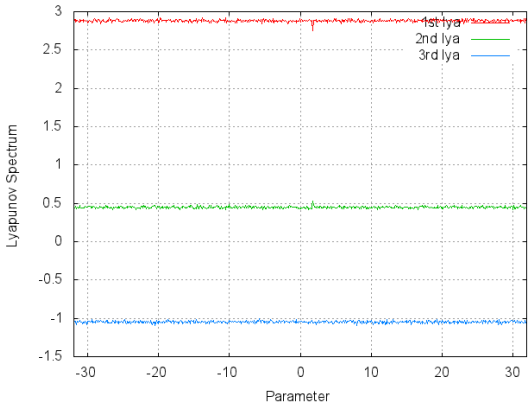

図 5.46　提案法 h_{32} のリアプノフスペクトラム

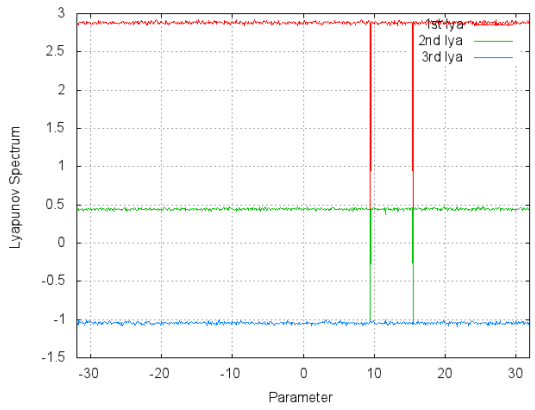

図 5.47　提案法 h_{33} のリアプノフスペクトラム

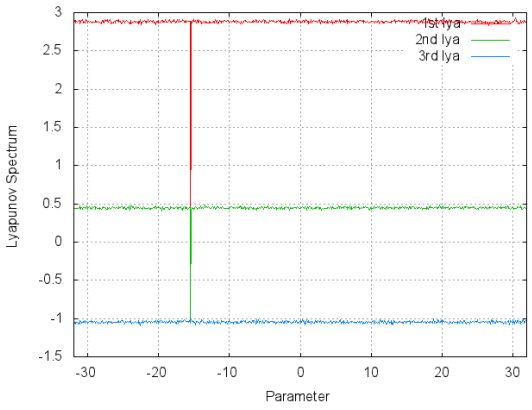

図 5.48　提案法 h_{123} のリアプノフスペクトラム

5.1.1　考察

従来法は変調式からヤコビ行列を求め，提案法は時系列からヤコビ行列を求めた．従来法は，多くのパラメータですべてのリアプノフ指数が正となった．３つのリアプノフ指数が正であるのは何かということについては執筆現在において定義されていないが，どのような状態かというと，これはすべての次元方向に伸び続けているということになり，発散傾向にあるということである．しかし当研究室では固定小数点演算を用いて有界性を得ており，かつ変復調が正しく行えているということから，これは即ち収束を得ており，カオスであると主張している．この結果に対して，提案法はパラメータ g_{31} の一部といくつかのパラメータで見られるマイナス無限大の点（縦線が現れている部分）以外は２つのリアプノフ指数が正，１つのリアプノフ指数が負を表している．これはハイパーカオスであり，パラメータ g_{31} の一部に関してはカオスであると言える．マイナス無限大の点に関しては写像後に近傍点が球内に含まれていなかったことによるゼロ除算が原因であり，この場合は，新たに球の半径サイズを大きく変更することによってこのような状態をを防ぐことができると考えられる．そのため，この点に関しては無視しても問題ないと判断する．そのため定義されていない従来法に比べて，提案法はハイパーカオスであるため，カオス性を有していると考えられる．

5.2　DIEHARD TEST によるランダム性の検証

ここでは DIEHARD TEST という乱数検定をランダム性の評価に用いることにする．DIEHARD TEST はフロリダ州立大学統計学部の Marsaglia 博士により開発された乱数検定である．[13] このテストは 18 種類の統計テストから構成されており，1 つまたは複数の $p-value$ を出力する．この $p-value$ に明確な基準は設けられていないが，本研究では，$p-value$ が

$$0.001 < p-value < 0.999$$

となるとき，そのテスト項目は合格とする．そして 18 種類すべてのテストに合格した場合，その 2 進系列はランダム性が高いと判断する．テストに必要な出力系列は 80M ビットである．以下にそれぞれのテストの概要を示す．

- バースデイ空間検定（The Birthday Spacings Test）
  0 と 1 からなる乱数列を 32 ビット単位で分割し，各 32 ビットから 24 ビットの Z を取り出した新たな列 $Z_1, Z_2, \ldots, Z_{1024}$ を作る．Z_i を小さい順に並べてその 1024 個の間隔の値 $\{Y_i\}$ を求める．Y_i を数直線上にプロットしていき，それまでプロットした値と重なった回数 s を数え，その偏りを調べる．ポアソン分布統計量からの $p-value$（9 個）と KS 検定による $p-value$ を出力する．

- OPEN5 検定（The Overlapping 5 - Permutation Test）
  0 と 1 からなる乱数列を 160 ビットの部分列とし，32 ビット谷の 5 文字組みとみなしたときの，5 文字組みの大小関係により，120 個のクラスに割り当て，その度数分の分布が一様になっているかを χ^2 検定に手検証する．χ^2 統計量からの $p-value$（2 個）を出力する．

- 31 × 31 の 2 値行列ランク検定（The Binary Rank for 31 × 31 Matrices）
  0 と 1 からなる乱数列を 160 ビットの部分列から 31 × 31 の 2 値行列を構成し，各部分列のランクに応じて 4 個のクラスに割り当て，その度数を χ^2 検定にて検定しランクの偏りを調べる．χ^2 統計量からの $p-value$ を出力する．

- 32 × 32 の 2 値行列ランク検定（The Binary Rank for 32 × 32 Matrices）
  上記の検定と同様の手法で 32 × 32 の 2 値行列を構成し，ランクの偏りを調べる．χ^2 統計量からの $p-value$ を出力する．

- 6 × 8 の 2 値行列ランク検定（The Binary Rank for 6 × 8 Matrices）
  0 と 1 からなる乱数列の 1564 ビットの部分列から 6 × 8 の 2 値行列を構成し，各部分列のランクに応じて 3 個のクラスに割り当て，その度数を χ^2 検定に手検定しランクの偏りを調べる．χ^2 統計量からの $p-value$ を出力する．

- ビット列検定（The Bitstreamk Test）
  0 と 1 からなる乱数列から 20 ビットの数値を 2^{21} 個作り，イリ度も出現しなかった数値の個数の偏りを調べる．尚，20 ビットの数値は，1 つ目が（X_1，X_2，...，X_{20}），2 つ目が（X_2，X_3，...，X_{21}）のように 1 ビットずつずらして重ねながらつくっていく．正規分布統計量からの $p-value$（20 個）を出力する．

- OPSO 検定（The Overlapping - Prairs - Sqarse - Occupancy Test）
  0 と 1 からなる乱数列を 32 ビット単位で分割し，各 32 ビットから 10 ビットの Y を取り出した新たな列 Y_1，Y_2，... をつくる．（Y_1，Y_2），（Y_2，Y_3），... と連続する 2 つの 10 ビットの組 2^{21} 個作り，一度も出現しなかった 2 つの 10 ビットの組の個数の偏りを調べる．正規分布統計量からの $p-value$（23 個）を出力する．

- OQSO 検定（The Overlapping - Quadruples - Squares - Occupancy Test）
  0 と 1 からなる乱数列を 32 ビット単位で分割し，各 32 ビットから 5 ビットの Y を取り出した新たな列 Y1，Y2，…をつくる．（Y1，Y2，Y3，Y4），（Y2，Y3，Y4，Y5），... と連続する 4 つの 5 ビットの組 2^21 個作り，一度も出現しなかった 4 つの 5 ビットの組の個数の偏りを調べる．正規分布統計量からの $p-value$（28 個）を出力する．

- DNA 検定（The DNA Test）
  0 と 1 からなる乱数列を乱数列を 32 ビット単位で分割し，各 32 ビットから 2 ビットの Y を取り出した新たな列 Y_1，Y_2，... をつくる．（Y_1，Y_2，...，Y_{10}），（Y_2，Y_3，...，Y_{11}），... と連続する 10 個の 2 ビットの組み 2^21 個作り，一度も出現しなかった 10 個の 2 ビットの組みの個数の偏りを調べる．正規分布統計量からの $p-value$（31 個）を出力する．

- 8 ビット中の文字数検定（Count - the - 1’s Test on Stream of Bytes）
  0 と 1 からなる乱数列の 40 ビットの部分列を核バイト中の 1 の個数に応じて各バイトを文字に置き換えて，その 5 文字組み 3625 個のクラスに割り当て，その度数を，χ^2 検定にて検定し，文字の出現頻度の偏りを調べる．χ^2 統計量からの $p-value$ を出力する．

- 特定位置の 8 ビット中の文字数検定（Count - the 1’s Test for Specific of Bytes）
  0 と 1 からなる乱数列の 160 ビットの部分列から特定位置の 8 バイトを選び，各バイト中の 1 の個数に応じて各バイトを文字に置き換えて，その 5 文字組み 3625 個のクラスに割り当て，その度数を χ^2 検定にて検定し，文字の出現頻度の偏りを調べる．χ^2 統計量からの $p-value$ を出力する．

- 駐車場検定（The Parking Lot Test）
  0 と 1 からなる乱数列を 32 ビットで分割し，各 32 ビットの Y を $u = 10Y/2^{32}$ と変換す

る．変換した乱数列の 2 文字の組み合わせを 2 次元座標上の点とみなし，12000 個の点をプロットしていく．プロットした点とこれまでにプロットされた点との距離を計算し，全ての点との距離が 1 より大きいときは，成功として成功回数の偏りを調べる．χ^2 統計量からの $p-value$（10 個）と KS 検定による $p-value$ を出力する．

- 最小距離検定（The Minimum Distance Test）
  0 と 1 からなる乱数列を 32 ビットで分割し，各 32 ビットの Y を $u = 10Y/2^{32}$ と変換する．変換した乱数列の 2 文字の組み合わせを 2 次元座標上の点とみなし，8000 個の点をプロットしていく．全ての点の組み合わせの中から距離が最小となる 2 点の距離を計算し，最小距離の偏りを調べる．χ^2 統計量からの $p-value$（20 個）と KS 検定による $p-value$ を出力する．

- 3DSPHERES（The 3D - Sphere Test）
  0 と 1 からなる 乱数列を 32 ビットで分割し，各 32 ビットの Y を $u = 10Y/2^{32}$ と変換する．変換した乱数列の 3 文字の組み合わせを 3 次元座標上の点とみなし，4000 個の点をプロットしていく．全ての点の組み合わせの中から距離が最小となる 2 点の距離を計算し，最小距離の偏りを調べる．χ^2 統計量からの $p-value$（20 個）と KS 検定による $p-value$ を出力する．

- スクイーズ検定（The Squeeze Test）
  0 と 1 からなる乱数列を 32 ビットで分割し，各 32 ビットの Y を $u = 10Y/2^{32}$ と [0，1] の少数に変換する．k の初期を $k = 2^{31}$ とし，$k \leftarrow [k \times u]$ を繰り返し，$k = 1$ となるまでの繰り返し回数に応じて各部分列を 43 個のクラスに割り当て，その度数を χ^2 検定にて検証する．χ^2 統計量からの $p-value$ を出力する．

- 重なりのある検定（The Overlapping Sums Test）
  0 と 1 からなる乱数列を 32 ビット単位で分割し，各 32 ビットから新たな列 $Y_1, Y_2, \ldots$ を作る．$(Y_1, Y_2, \ldots, Y_{100}), (Y_2, Y_3, \ldots, Y_{101}), \ldots$ と連続する 100 個の組みを作り，それぞれの組の和を計算し，求めた和が一様に分布しているかを調べる．KS 検定による $p-value$（10 個）と，この 10 個の $p-value$ への KS 検定による $p-value$ を出力する．

- 連の検定（The Runs Test）
  区間 [0，1] 上の分布する乱数列を上昇連，下降連で分割し，部分列の長さでクラスを定義する．部分列の長さに応じてクラスに割り当て，その度数を χ^2 検定にて検定し，連の偏りを調べる．尚，乱数列が 0 と 1 からなる乱数の場合は，区間 [0，1] に分布する乱数列への変換が必要である．KS 検定による $p-value$（2 個）を出力する．

- クラップス検定（The Craps Test）

0 と 1 からなる乱数列を 32 ビットで分割し，各 32 ビットの Y を $u = [Y/2^{32} \times 6]+1$ と変換する．u を *Craps* におけるサイコロの値とみなして *Craps* を 2000 回行い，1 回の勝負がつくまで繰り返し回数に応じて各部分列を 21 個のクラスに割り当て，その度数を χ^2 検定にて検定する．また，確立の偏りも調べる．正規分布統計量からの $p-value$（勝った回数に対する）と χ^2 統計量からの $p-value$（1 回の勝負がつくまでの繰り返し回数）を出力する．

提案法のカオス暗号システムはボルテラフィルタ内の乗算を一部 XOR 演算に置き換えるものであった．前述したとおり，すべての乗算を XOR 演算に置き換えたものは DIEHARD TEST によってランダム性がないと判断されている．なぜそのような結果になるのか，生成した暗号文の内部状態を確認したところ，だんだん値が小さくなっていき，最終的にはゼロ信号に収束してしまったものであった．このままでは暗号としては使えないため，どのパラメータを XOR 演算するべきなのかをこのテストによって調べることにした．具体的には，各パラメータを一つずつ XOR 演算したものに DIEHARD TEST を行い，結果が良かったパラメータだけを選別し，それらを組み合わせて最適化を図ったものが提案式である．そこで，まずは提案式の結果について示す前に，提案式の導出過程に至るまでの各パラメータを XOR 演算した結果を表 5.1〜5.4 に示す．結果より，各パラメータの合格率に一度も 0% を含んでおらず，かつ平均値が 60% 以上だったものであるパラメータ $g_3, g_{11}, g_{12}, g_{13}, g_{21}, g_{23}, g_{32}, h_3, h_{11}, h_{12}, h_{13}, h_{31}, h_{33}, h_{123}$ の合計 14 個のパラメータを XOR 演算すべきと判断し，これを提案法とした．そして，従来法と提案法における DIEHARD TEST の結果を表 5.5，表 5.6 に示す．なお，このテストに限っては平文と内部状態変数の初期値はすべて 0 とし，結果の値は各パラメータを最小値から最大値まで 1.0 ずつ変化させながらテストを行い，その平均値を載せた．つまり，パラメータ 1 つに対して 65 回テストを行っているということである．

表 5.1　各パラメータを XOR 演算したときの DIEHARD TEST の結果 ($g_1, g_2, g_3, g_{11}, g_{12}, g_{13}, g_{21}$)

	g_1	g_2	g_3	g_{11}	g_{12}	g_{13}	g_{21}
g_1	3.0769	63.0769	69.2308	76.9231	73.8462	63.0769	61.5385
g_2	50.7692	0.0000	49.2308	55.3846	67.6923	63.0769	50.7692
g_3	72.3077	60.0000	36.9231	72.3077	70.7692	72.3077	70.7692
g_{11}	72.3077	61.5385	76.9231	41.5385	69.2308	64.6154	58.4615
g_{12}	55.3846	61.5385	70.7692	80.0000	64.6154	55.3846	66.1538
g_{13}	67.6923	61.5385	70.7692	67.6923	66.1538	75.3846	67.6923
g_{21}	60.0000	67.6923	61.5385	69.2308	61.5385	64.6154	58.4615
g_{22}	47.6923	0.0000	52.3077	64.6154	72.3077	64.6154	53.8462
g_{23}	60.0000	58.4615	60.0000	66.1538	69.2308	76.9231	64.6154
g_{31}	69.2308	56.9231	60.0000	66.1538	56.9231	64.6154	58.4615
g_{32}	75.3846	58.4615	61.5385	78.4615	75.3846	73.8462	73.8462
g_{33}	60.0000	64.6154	60.0000	83.0769	56.9231	69.2308	75.3846
g_{123}	64.6154	67.6923	60.0000	60.0000	61.5385	75.3846	55.3846
θ_1	15.3846	0.0000	43.0769	43.0769	55.3846	61.5385	66.1538
h_1	53.8462	60.0000	70.7692	63.0769	66.1538	64.6154	66.1538
h_2	44.6154	0.0000	53.8462	66.1538	73.8462	72.3077	60.0000
h_3	61.5385	69.2308	63.0769	67.6923	73.8462	64.6154	69.2308
h_{11}	70.7692	69.2308	66.1538	55.3846	70.7692	61.5385	67.6923
h_{12}	67.6923	61.5385	69.2308	76.9231	67.6923	63.0769	69.2308
h_{13}	61.5385	52.3077	75.3846	66.1538	66.1538	76.9231	50.7692
h_{21}	72.3077	52.3077	58.4615	67.6923	61.5385	58.4615	72.3077
h_{22}	56.9231	0.0000	53.8462	67.6923	64.6154	72.3077	49.2308
h_{23}	64.6154	60.0000	60.0000	76.9231	66.1538	58.4615	58.4615
h_{31}	64.6154	63.0769	58.4615	72.3077	80.0000	66.1538	58.4615
h_{32}	63.0769	58.4615	56.9231	80.0000	55.3846	67.6923	75.3846
h_{33}	63.0769	69.2308	63.0769	60.0000	58.4615	67.6923	64.6154
h_{123}	63.0769	56.9231	53.8462	66.1538	63.0769	73.8462	72.3077
θ_2	15.3846	0.0000	44.6154	66.1538	53.8462	66.1538	58.4615
平均値	57.0330	48.3517	60.0000	67.0330	65.8242	67.0879	63.3516

横：XOR 演算したパラメータ

縦：各パラメータの合格率 [%]

表 5.2　各パラメータを XOR 演算したときの DIEHARD TEST の結果 (g_{22}, g_{23}, g_{31}, g_{32}, g_{33}, g_{123})

	g_{22}	g_{23}	g_{31}	g_{32}	g_{33}	g_{123}
g_1	64.6154	69.2308	72.3077	66.1538	64.6154	60.0000
g_2	0.0000	64.6154	70.7692	61.5385	76.9231	47.6923
g_3	73.8462	64.6154	60.0000	69.2308	61.5385	58.4615
g_{11}	67.6923	69.2308	63.0769	49.2308	66.1538	63.0769
g_{12}	0.0000	100	0.0000	100	0.0000	0.0000
g_{13}	58.4615	67.6923	69.2308	70.7692	63.0769	56.9231
g_{21}	66.1538	63.0769	64.6154	61.5385	61.5385	58.4615
g_{22}	0.0000	60.0000	70.7692	72.3077	61.5385	64.6154
g_{23}	60.0000	67.6923	66.1538	70.7692	63.0769	55.3846
g_{31}	63.0769	81.5385	60.0000	81.5385	69.2308	72.3077
g_{32}	66.1538	49.2308	60.0000	63.0769	78.4615	49.2308
g_{33}	63.0769	63.0769	75.3846	67.6923	60.0000	60.0000
g_{123}	63.0769	69.2308	64.6154	61.5385	64.6154	0.0000
θ_1	0.0000	60.0000	67.6923	66.1538	60.0000	0.0000
h_1	58.4615	63.0769	69.2308	64.6154	53.8462	0.0000
h_2	0.0000	64.6154	69.2308	63.0769	60.0000	0.0000
h_3	72.3077	53.8462	76.9231	58.4615	66.1538	0.0000
h_{11}	66.1538	73.8462	67.6923	63.0769	73.8462	0.0000
h_{12}	69.2308	63.0769	75.3846	69.2308	50.7692	0.0000
h_{13}	66.1538	58.4615	60.0000	73.8462	64.6154	0.0000
h_{21}	70.7692	63.0769	61.5385	70.7692	60.0000	0.0000
h_{22}	0.0000	60.0000	78.4615	73.8462	67.6923	0.0000
h_{23}	61.5385	60.0000	72.3077	70.7692	72.3077	0.0000
h_{31}	69.2308	58.4615	63.0769	58.4615	60.0000	0.0000
h_{32}	69.2308	67.6923	67.6923	60.0000	66.1538	0.0000
h_{33}	67.6923	49.2308	60.0000	63.0769	69.2308	0.0000
h_{123}	73.8462	70.7692	55.3846	64.6154	73.8462	0.0000
θ_2	0.0000	64.6154	70.7692	61.5385	60.0000	0.0000
平均値	49.6703	65.0000	64.7253	67.0330	62.4725	23.0769

横：XOR 演算したパラメータ

縦：各パラメータの合格率 [%]

表 5.3　各パラメータを XOR 演算したときの DIEHARD TEST の結果 (h_1, h_2, h_3, h_{11}, h_{12}, h_{13}, h_{21})

	h_1	h_2	h_3	h_{11}	h_{12}	h_{13}	h_{21}
g_1	44.6154	56.9231	61.5385	58.4615	64.6154	67.6923	76.9231
g_2	55.3846	0.0000	46.1538	60.0000	60.0000	69.2308	53.8462
g_3	66.1538	69.2308	58.4615	61.5385	63.0769	67.6923	58.4615
g_{11}	63.0769	55.3846	63.0769	76.9231	69.2308	67.6923	58.4615
g_{12}	61.5385	61.5385	58.4615	100	67.6923	53.8462	0.0000
g_{13}	67.6923	61.5385	60.0000	69.2308	67.6923	73.8462	70.7692
g_{21}	55.3846	60.0000	72.3077	69.2308	53.8462	66.1538	64.6154
g_{22}	58.4615	0.0000	55.3846	69.2308	64.6154	70.7692	69.2308
g_{23}	67.6923	58.4615	58.4615	66.1538	80.0000	64.6154	67.6923
g_{31}	69.2308	66.1538	73.8462	80.0000	72.3077	69.2308	66.1538
g_{32}	70.7692	55.3846	63.0769	75.3846	58.4615	67.6923	67.6923
g_{33}	69.2308	63.0769	70.7692	70.7692	66.1538	75.3846	73.8462
g_{123}	75.3846	66.1538	56.9231	66.1538	66.1538	63.0769	70.7692
θ_1	15.3846	0.0000	44.6154	64.5385	73.8462	64.6154	50.7692
h_1	4.6154	63.0769	63.0769	53.8462	73.8462	83.0769	66.1538
h_2	64.6154	0.0000	50.7692	66.1538	60.0000	64.6154	64.6154
h_3	66.1538	70.7692	41.5385	67.6923	69.2308	47.6923	67.6923
h_{11}	67.6923	61.5385	73.8462	63.0769	70.7692	69.2308	66.1538
h_{12}	66.1538	60.0000	67.6923	64.6154	63.0769	70.7692	60.0000
h_{13}	67.6923	66.1538	72.3077	70.7692	63.0769	61.5385	67.6923
h_{21}	64.6154	52.3077	66.1538	64.6154	67.6923	69.2308	56.9231
h_{22}	55.3846	0.0000	46.1538	63.0769	66.1538	61.5385	56.9231
h_{23}	67.6923	69.2308	60.0000	63.0769	66.1538	75.3846	60.0000
h_{31}	64.6154	60.0000	63.0769	67.6923	67.6923	67.6923	55.3846
h_{32}	60.0000	66.1538	63.0769	67.6923	63.0769	66.1538	61.5385
h_{33}	61.5385	72.3077	55.3846	64.6154	64.6154	56.9231	61.5385
h_{123}	70.7692	61.5385	66.1538	60.0000	64.6154	69.2308	53.8462
θ_2	18.4615	0.0000	50.7692	63.0769	60.0000	64.6154	64.6154
平均値	58.5714	49.1758	60.1099	67.3077	65.9890	66.7583	61.1539

横：XOR 演算したパラメータ

縦：各パラメータの合格率 [%]

表 5.4　各パラメータを XOR 演算したときの DIEHARD TEST の結果 (h_{22}, h_{23}, h_{31}, h_{32}, h_{33}, h_{123})

	h_{22}	h_{23}	h_{31}	h_{32}	h_{33}	h_{123}
g_1	49.2308	67.6923	72.3077	73.8462	75.3846	61.5385
g_2	0.0000	60.0000	72.3077	66.1538	55.3846	56.9231
g_3	64.6154	67.6923	69.2308	70.7692	63.0769	67.6923
g_{11}	70.7692	64.6154	58.4615	64.6154	69.2308	66.1538
g_{12}	0.0000	0.0000	100	0.0000	63.0769	100
g_{13}	73.8462	67.6923	64.6154	61.5385	69.2308	64.6154
g_{21}	53.8462	61.5385	69.2308	70.7692	70.7692	67.6923
g_{22}	0.0000	63.0769	70.7692	58.4615	56.9231	76.9231
g_{23}	60.0000	64.6154	61.5385	70.7692	69.2308	69.2308
g_{31}	60.0000	64.6154	58.4615	56.9231	58.4615	78.4615
g_{32}	63.0769	66.1538	63.0769	72.3077	60.0000	70.7692
g_{33}	56.9231	61.5385	67.6923	67.6923	66.1538	70.7692
g_{123}	60.0000	70.7692	56.9231	67.6923	66.1538	67.6923
θ_1	0.0000	66.1538	66.1538	63.0769	52.3077	67.6923
h_1	66.1538	67.6923	53.8462	70.7692	58.4615	53.8462
h_2	0.0000	64.6154	67.6923	56.9231	61.5385	63.0769
h_3	70.7692	60.0000	76.9231	73.8462	69.2308	53.8462
h_{11}	55.3846	69.2308	69.2308	64.6154	61.5385	61.5385
h_{12}	55.3846	67.6923	69.2308	58.4615	49.2308	64.6154
h_{13}	52.3077	69.2308	78.4615	58.4615	72.3077	58.4615
h_{21}	61.5385	72.3077	61.5385	56.9231	52.3077	67.6923
h_{22}	0.0000	66.1538	60.0000	64.6154	70.7692	61.5385
h_{23}	66.1538	63.0769	69.2308	66.1538	60.0000	53.8462
h_{31}	52.3077	63.0769	60.0000	69.2308	73.8462	69.2308
h_{32}	63.0769	70.7692	70.7692	73.8462	58.4615	67.6923
h_{33}	66.1538	69.2308	58.4615	69.2308	55.3846	63.0769
h_{123}	66.1538	70.7692	70.7692	70.7692	70.7692	67.6923
θ_2	0.0000	73.8462	66.1538	72.3077	56.9231	56.9231
平均値	45.9890	64.0659	67.2528	63.9560	63.0769	66.0440

横：XOR 演算したパラメータ

縦：各パラメータの合格率 [%]

表 5.5　DiehardTest 合格率 [%]

パラメータ	従来法	提案法	パラメータ	従来法	提案法
g_0	58.4615		θ_2		56.9231
g_1		56.9231	h_1		64.6154
g_2		78.4615	h_2		64.6154
g_3		66.1538	h_3		56.9231
g_{11}	66.1538	67.6923	h_{11}	80.0000	55.3846
g_{12}	60.0000	75.3846	h_{12}	63.0769	72.3077
g_{13}	72.3077	69.2308	h_{13}	60.0000	67.6923
g_{21}	61.5385	75.3846	h_{21}	70.7692	70.7692
g_{22}	61.5385	66.1538	h_{22}	78.4615	61.5385
g_{23}	67.6923	81.5385	h_{23}	63.0769	64.6154
g_{31}	63.0769	75.3846	h_{31}	66.1538	67.6923
g_{32}	70.7692	81.5385	h_{32}	61.5385	61.5385
g_{33}	60.0000	66.1538	h_{33}	67.6923	55.3846
g_{123}	64.6154	73.8462	h_{123}		70.7692
θ_1		66.1538	平均値	65.8462	67.5275

表 5.6　DiehardTest 結果

	従来法	提案法
最低合格率 [%]	58.4615	55.3846
最高合格率 [%]	80.0000	81.5385
平均合格率 [%]	65.8462	67.5275
パラメータ数	20	28
合格率 70 %以上のパラメータ数	5	10

5.2.1　考察

提案法は従来法よりも約 1.68% の向上が見られた．つまり XOR 演算は乗算と同じかそれ以上の複雑性を生み出しているということである．XOR 演算するパラメータを選定する際に見られた合格率 0% はおそらくはカオスの窓と呼ばれる無秩序の中（非周期部分）に突如として現れる秩序（周期が発生している）の状態であると思われる．

5.3　パラメータミスマッチングによる係数感度の検証

本研究ではパラメータの値をそのまま暗号鍵として用いている．もし正しい値以外で復調できてしまった場合は暗号鍵として適切であるとは言えない．パラメータミスマッチング（係数感度）はパラメータを暗号鍵として使用するのに有効なものであるかどうかを検証するものである．この検証では，各パラメータの値が取り得るすべての値で変復調し，復調後の信号と元の信号との誤差を算出することにより行う．その式を式（5.3）に示す．

$$D = -\frac{1}{L}\sum_{n=0}^{L-1}|r(n) - s(n)| \tag{5.3}$$

この式において，$D = 0$（誤差が 0）となった時に正しく復調できているということになる．L は検証に用いたサンプル数で本研究では $L = 1000$ とした．また，$r(n)$ は復調信号，$s(n)$ は元信号である．従来法，提案法のパラメータミスマッチングの結果を図（5.49）～（5.96）に示し，まとめたものを表 5.7 に示す．

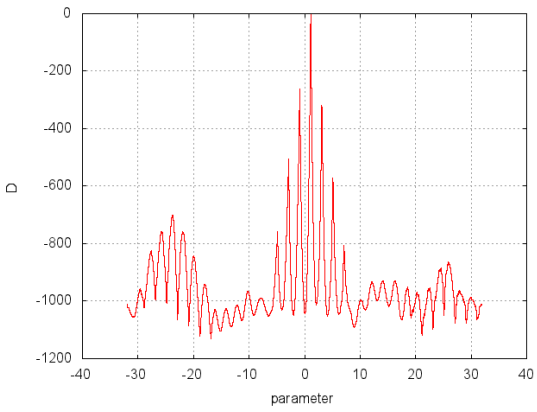

図 5.49　従来法 g_1 の係数感度

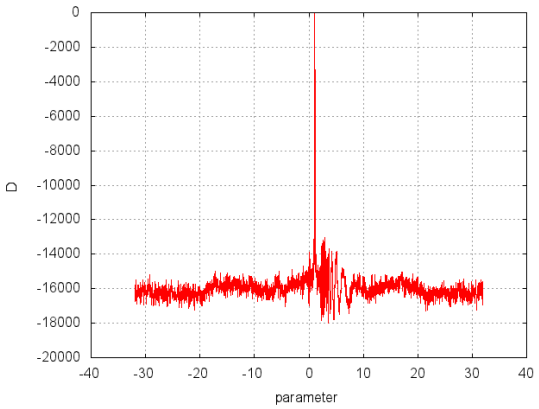

図 5.50　従来法 g_{11} の係数感度

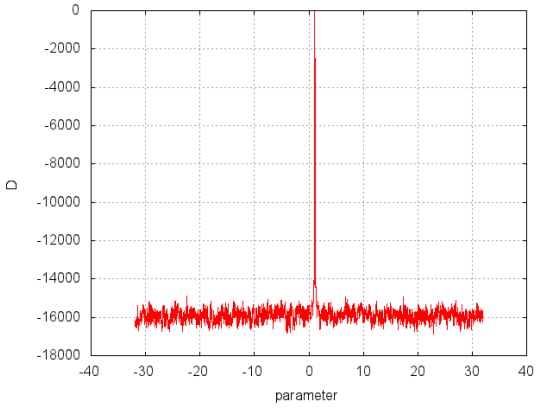

図 5.51　従来法 g_{12} の係数感度

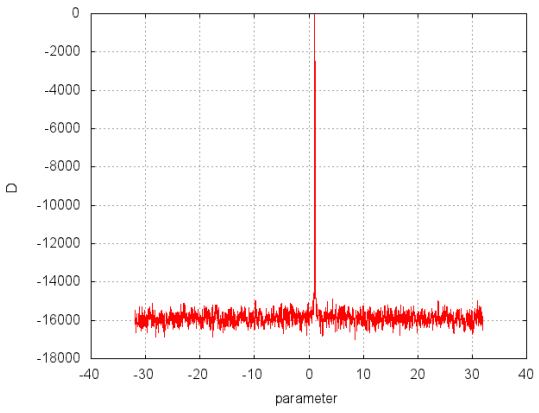

図 5.52　従来法 g_{13} の係数感度

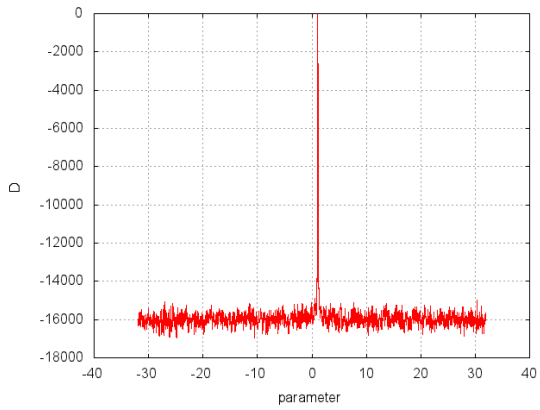

図 5.53　従来法 g_{21} の係数感度

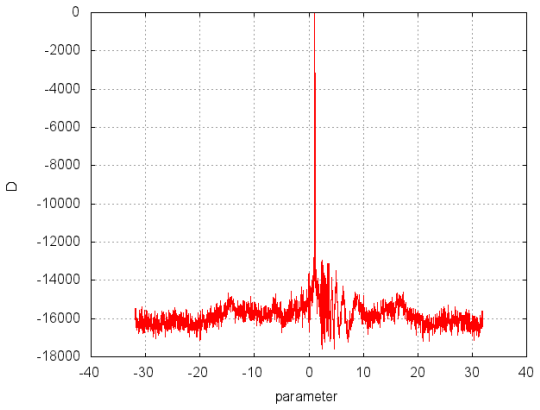

図 5.54　従来法 g_{22} の係数感度

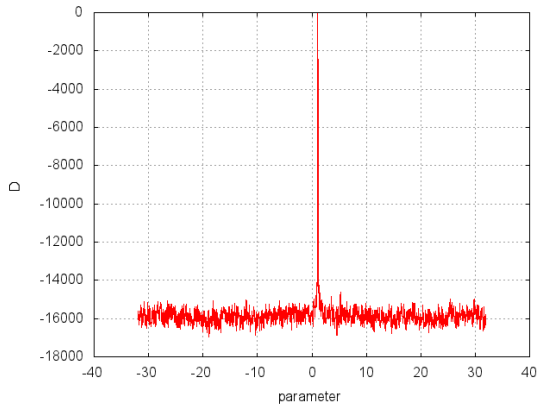

図 5.55　従来法 g_{23} の係数感度

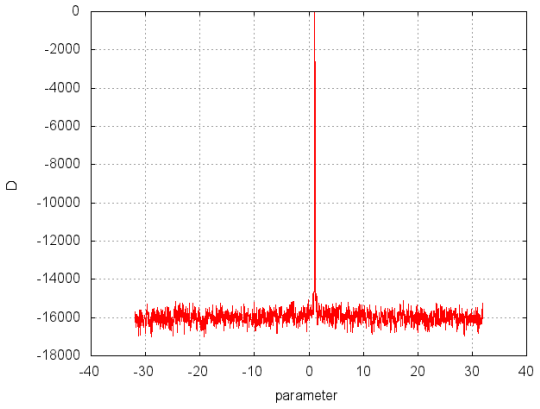

図 5.56　従来法 g_{31} の係数感度

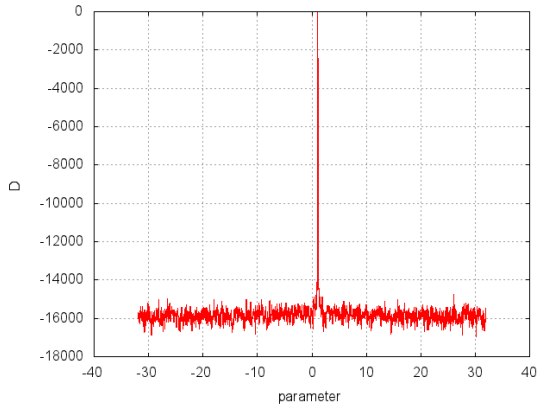

図 5.57　従来法 g_{32} の係数感度

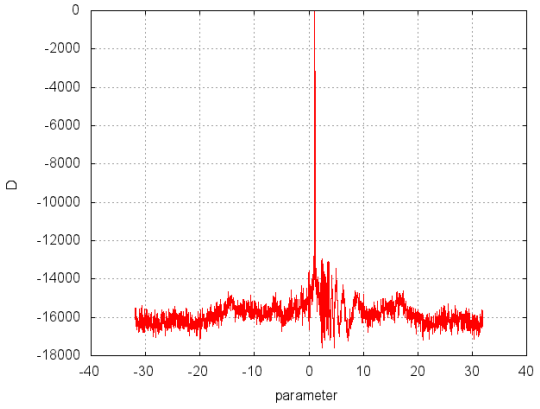

図 5.58　従来法 g_{33} の係数感度

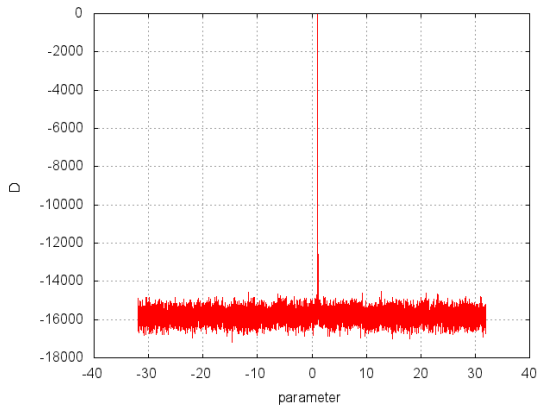

図 5.59　従来法 g_{123} の係数感度

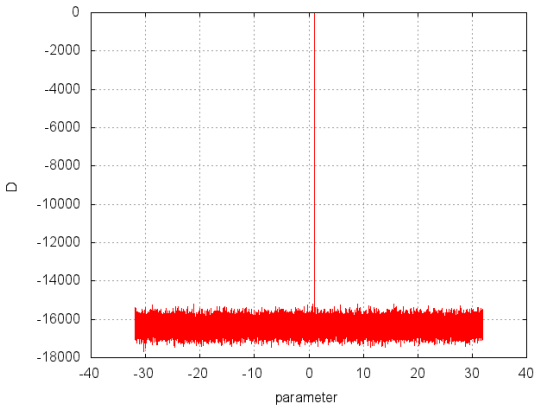

図 5.60　従来法 h_{11} の係数感度

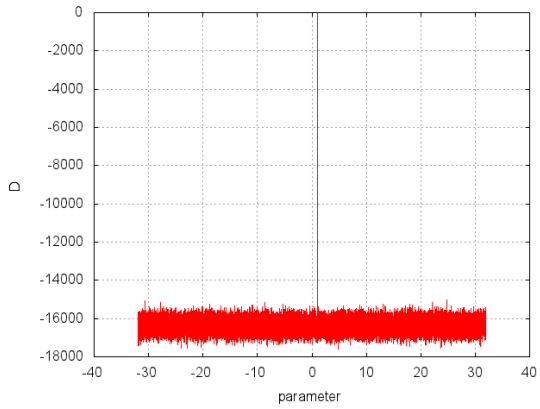

図 5.61　従来法 h_{12} の係数感度

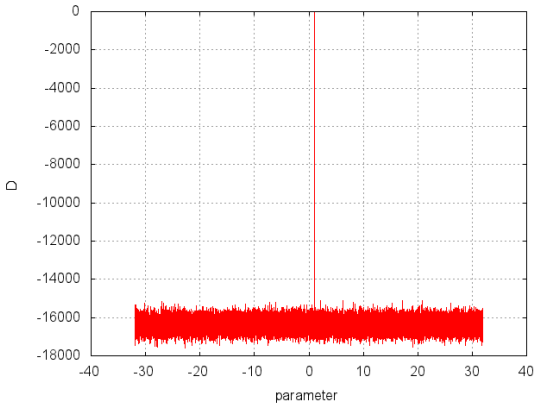

図 5.62　従来法 h_{13} の係数感度

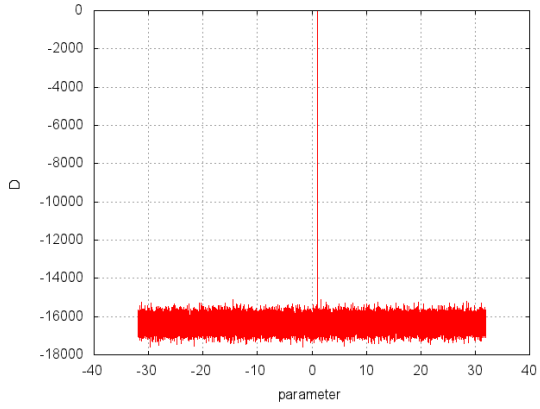

図 5.63　従来法 h_{21} の係数感度

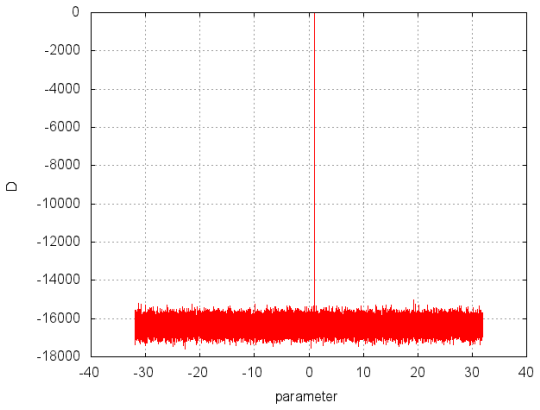

図 5.64　従来法 h_{22} の係数感度

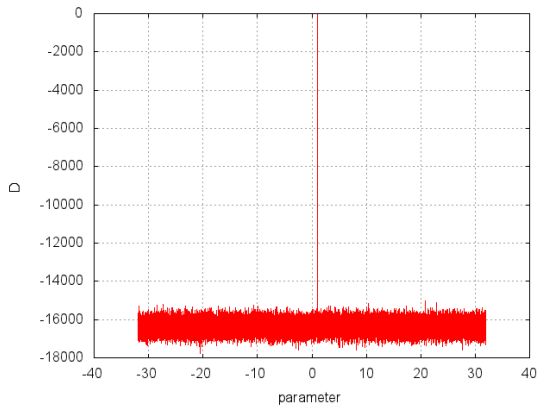

図 5.65　従来法 h_{23} の係数感度

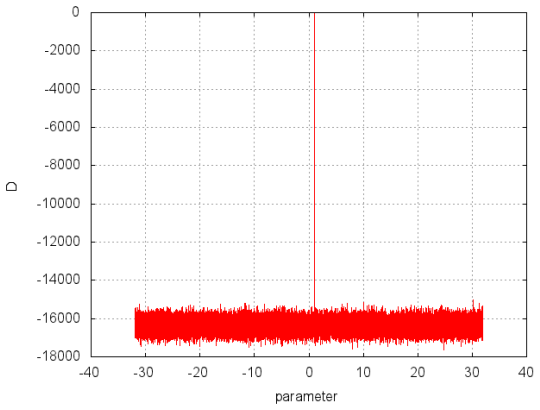

図 5.66　従来法 h_{31} の係数感度

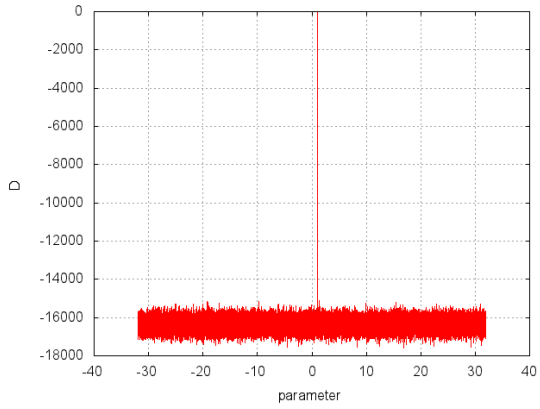

図 5.67　従来法 h_{32} の係数感度

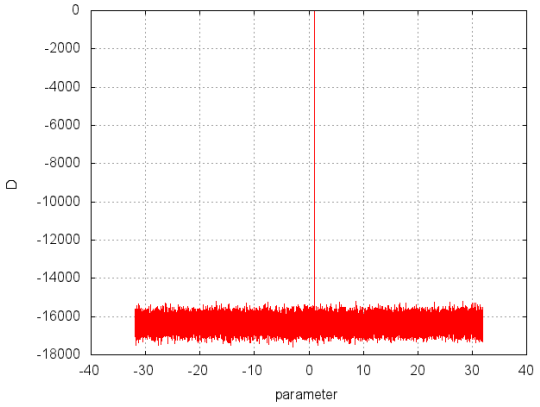

図 5.68　従来法 h_{33} の係数感度

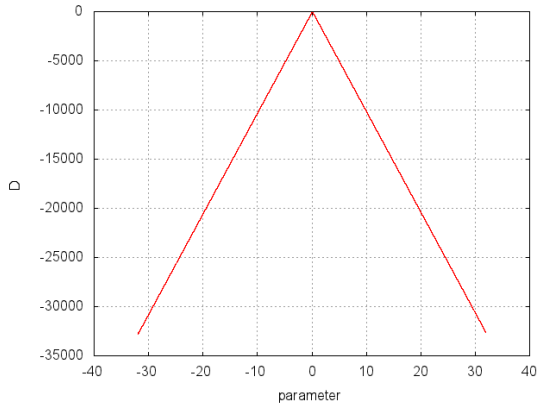

図 5.69　提案法 θ_1 の係数感度

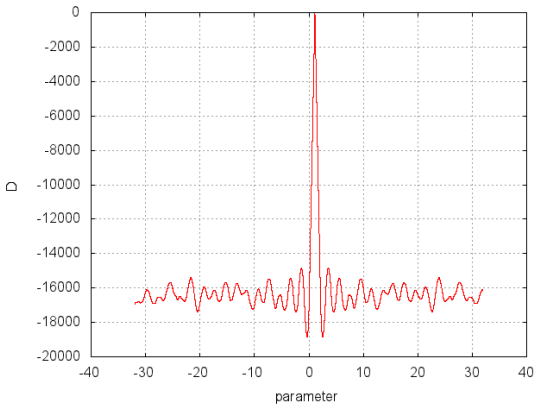

図 5.70　提案法 g_1 の係数感度

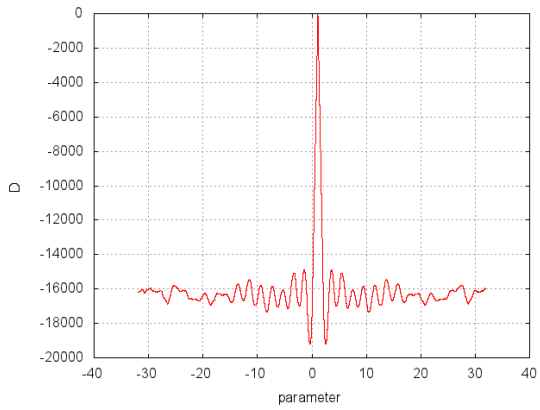

図 5.71　提案法 g_2 の係数感度

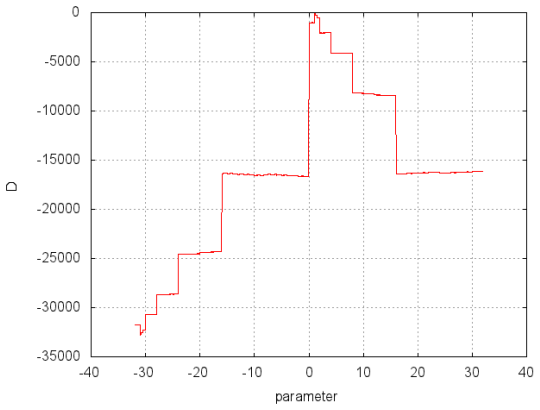

図 5.72　提案法 g_3 の係数感度

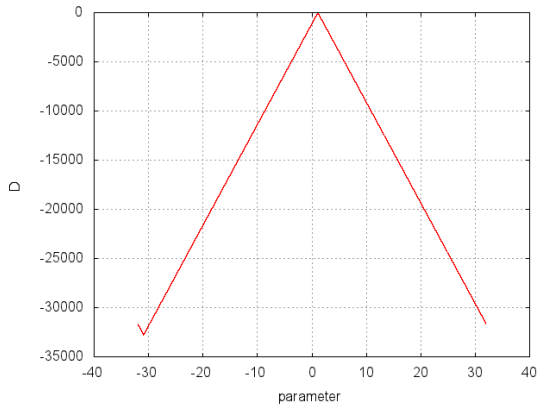

図 5.73　提案法 g_{11} の係数感度

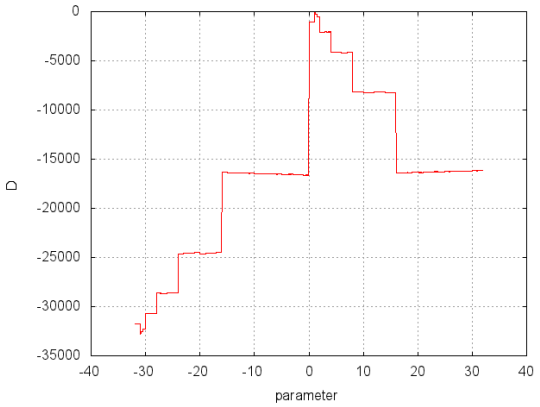

図 5.74　提案法 g_{12} の係数感度

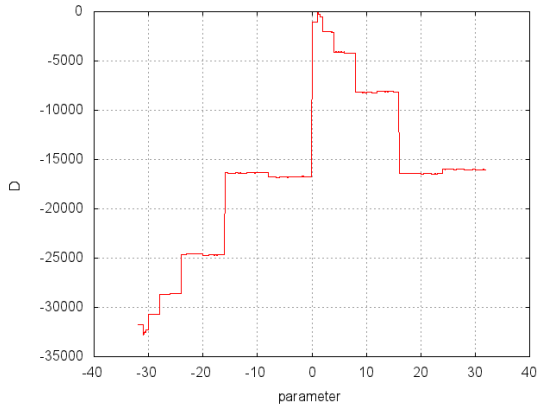

図 5.75　提案法 g_{13} の係数感度

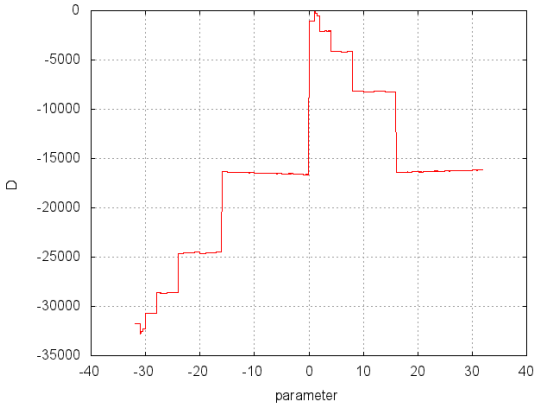

図 5.76　提案法 g_{21} の係数感度

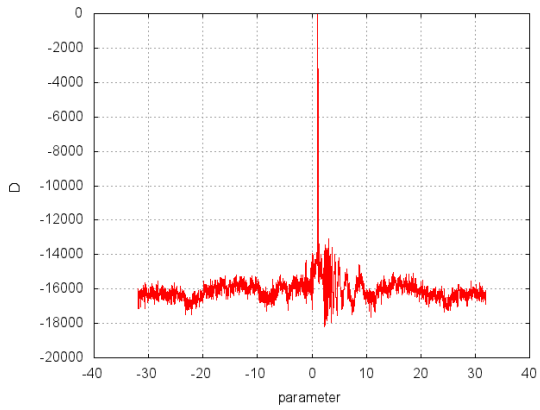

図 5.77　提案法 g_{22} の係数感度

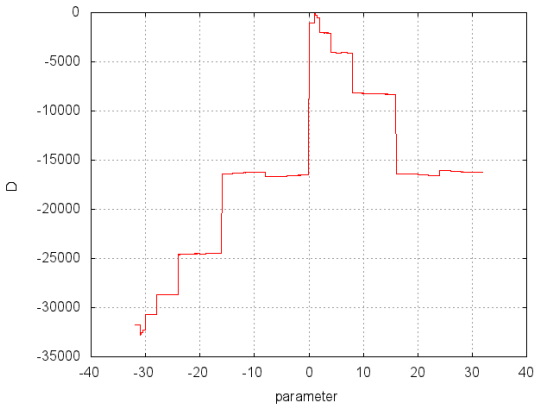

図 5.78　提案法 g_{23} の係数感度

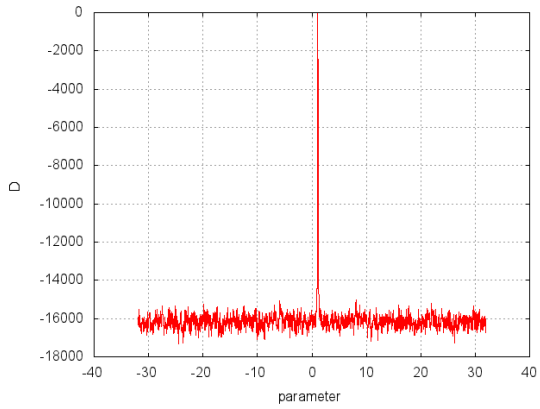

図 5.79　提案法 g_{31} の係数感度

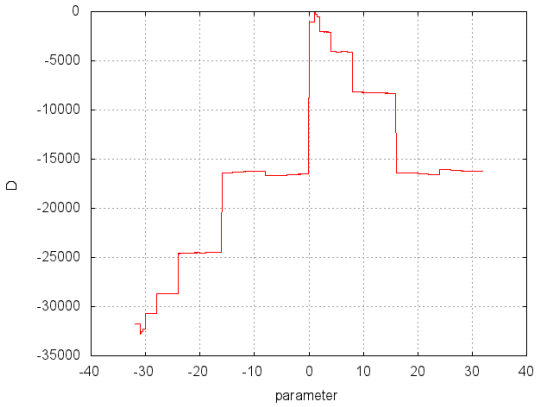

図 5.80　提案法 g_{32} の係数感度

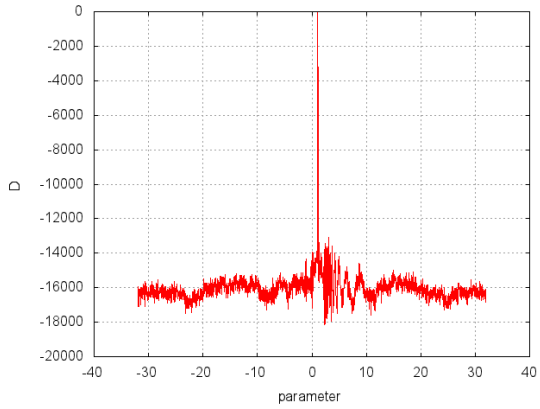

図 5.81　提案法 g_{33} の係数感度

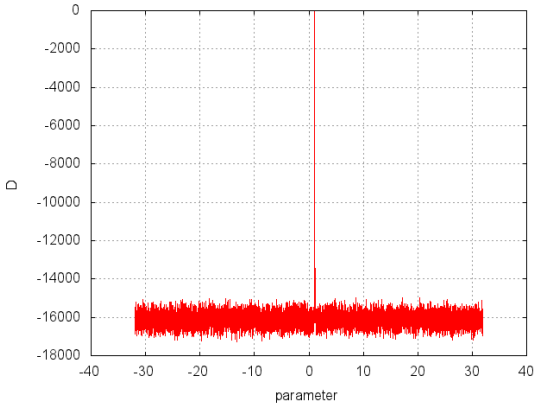

図 5.82　提案法 g_{123} の係数感度

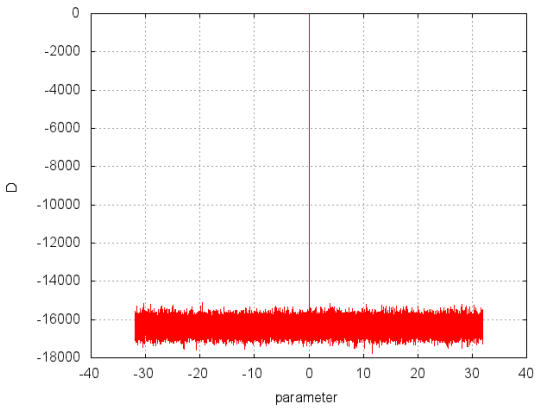

図 5.83　提案法 θ_2 の係数感度

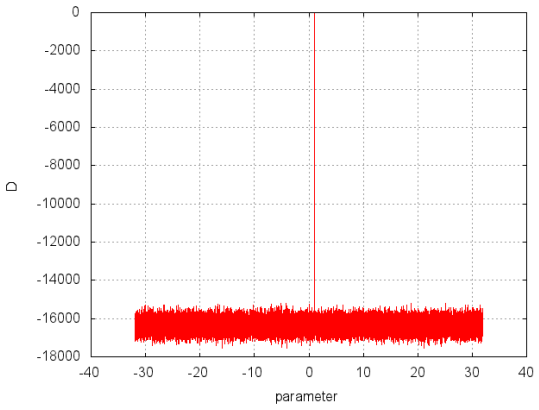

図 5.84　提案法 h_1 の係数感度

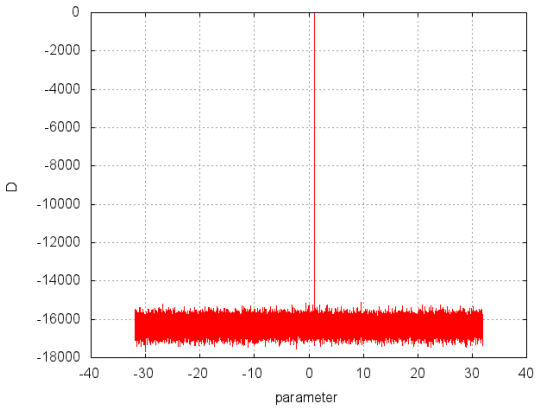

図 5.85　提案法 h_2 の係数感度

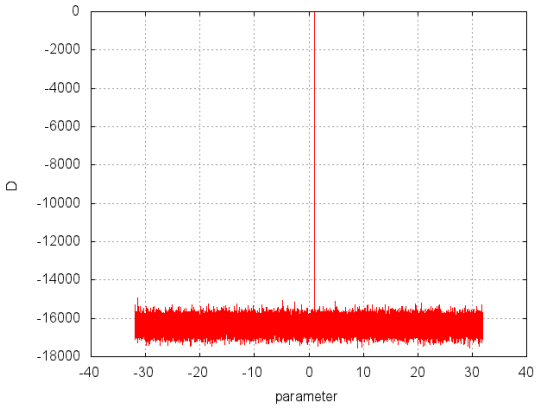

図 5.86　提案法 h_3 の係数感度

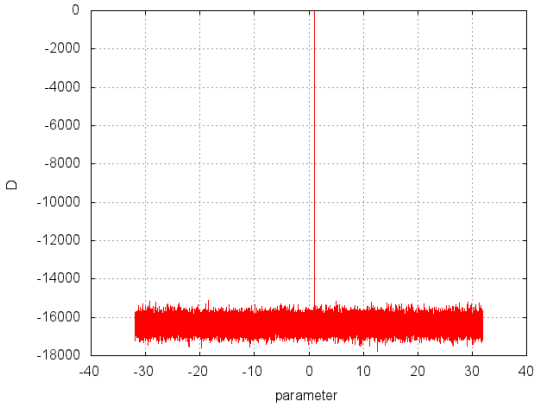

図 5.87　提案法 h_{11} の係数感度

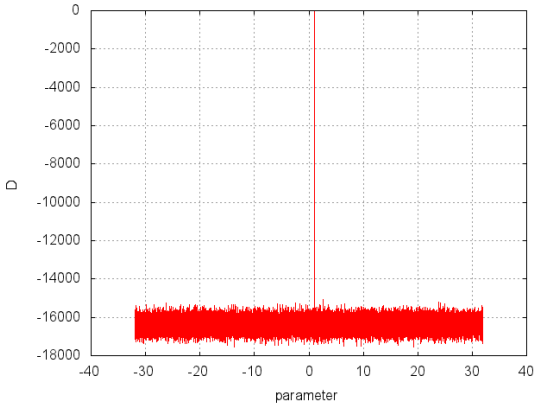

図 5.88　提案法 h_{12} の係数感度

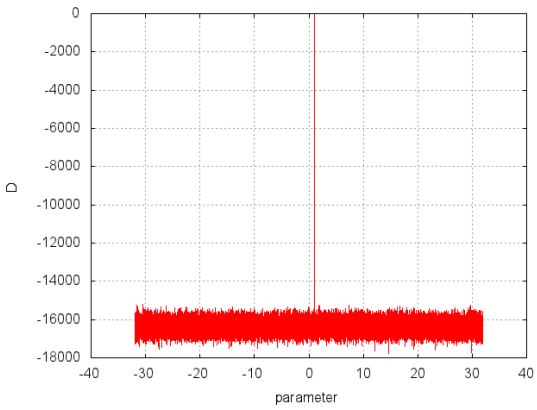

図 5.89　提案法 h_{13} の係数感度

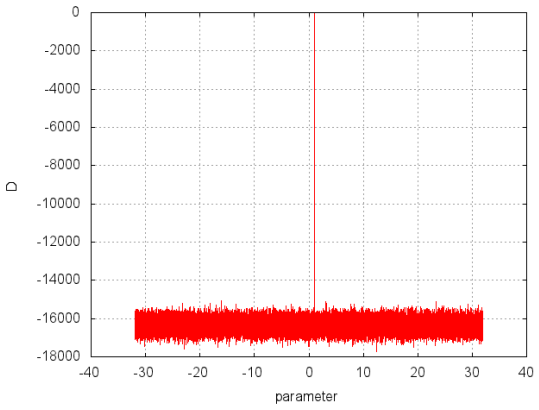

図 5.90　提案法 h_{21} の係数感度

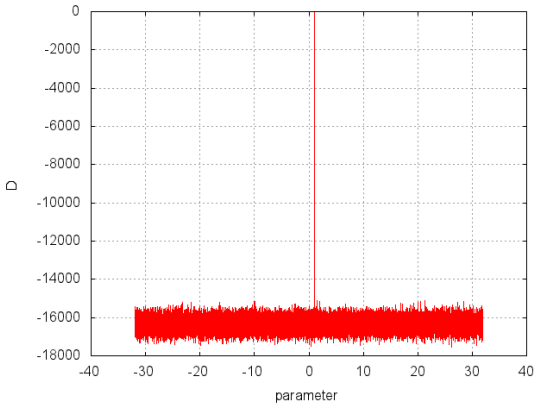

図 5.91　提案法 h_{22} の係数感度

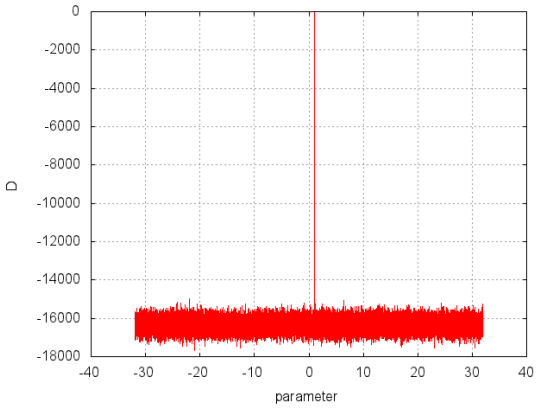

図 5.92　提案法 h_{23} の係数感度

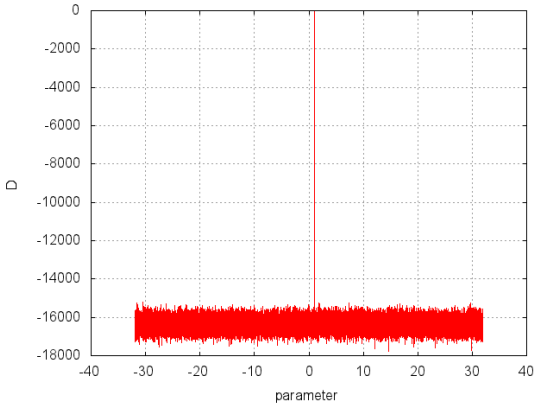

図 5.93　提案法 h_{31} の係数感度

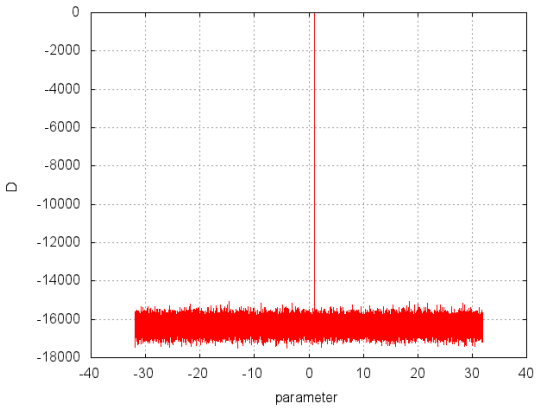

図 5.94　提案法 h_{32} の係数感度

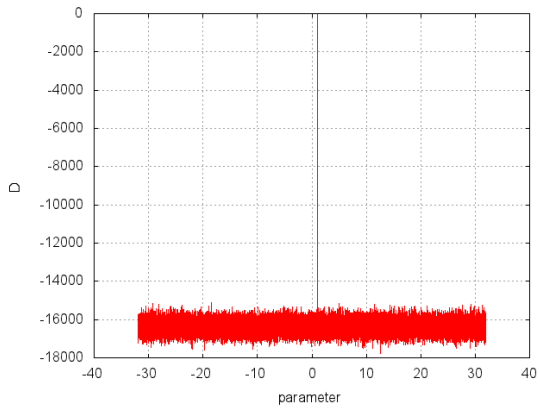

図 5.95　提案法 h_{33} の係数感度

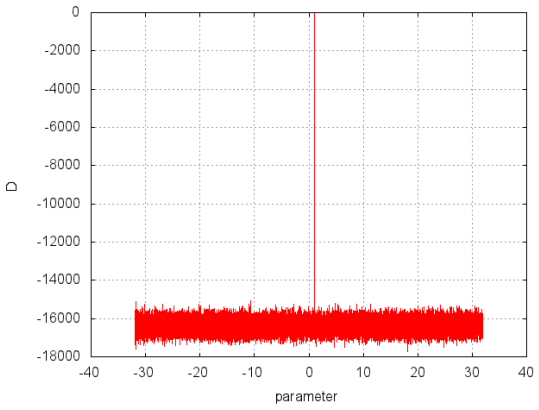

図 5.96　提案法 h_{123} の係数感度

表 5.7　パラメータミスマッチングの結果

	従来法	提案法
暗号鍵の鍵長 [bit]	320	448
有効な暗号鍵の鍵長 [bit]	256	256
有効な暗号鍵の割合 [%]	80.0	57.1
第 1 式の有効な暗号鍵の鍵長 [bit]	112	32
第 2 式の有効な暗号鍵の鍵長 [bit]	144	224

※パラメータは 1 個につき 16 ビットである．

5.3.1　考察

従来法，提案法共に 16 個のパラメータが暗号鍵として有効であると判断した．暗号鍵として有効なものは正しい値付近でも誤差（D の値）が均等に分布しているものが望ましい．有効でない暗号鍵の特徴として，従来法に関しては g_1 のように規則的な波形を示してしまうもの，g_{11}, g_{22}, g_{33} のように正しい値付近で波形が乱れてしまうものは暗号鍵としてふさわしくないと判断した．提案法に関しても同様で，θ_1, g_{11} のように三角波となってしまうもの，g_1, g_2 のように規則的な波形を示したり g_{22}, g_{33} のように正しい値付近で波形が乱れるもの，さらに $g_3, g_{12}, g_{13}, g_{21}, g_{23}, g_{32}$ のように階段状の波形になっているものは暗号鍵としてふさわしくないと判断した．その結果，提案法は従来法に比べて付与したパラメータの数の割に暗号鍵として使用可能であると判断された割合は少ない結果となった．特に，第 1 式に付与したパラメータはわずか 2 つしか有効ではないと判断されており，さらに細かく見ていくと，従来法に関しても第 1 式の結果より第 2 式の方が良いことがわかる．考えられる原因としては非線形関数を取り除いたことによる結果であるのが原因だと思われる．

5.4　暗号化処理速度の検証

この検証では，従来法と提案法の各暗号システムに対して 100MB のデータを暗号化させ, 暗号化に要した時間を測定する. 暗号化は 100 回行い, その平均をとる. 測定した時間から式（5.4）により 1 秒間当たりの暗号化速度を算出する.

$$\text{暗号化速度 } [Mbps] = (100[MB] \times 8[bits] \div \text{暗号化処理時間 } [s]) \tag{5.4}$$

その結果を表 5.8 に示した．なお，測定環境を表 5.9 に示す.

表 5.8　暗号化速度検証

	従来法	提案法
暗号化速度 [Mbps/sec]	552.555	1028.68
従来法に対する暗号化速度 [%]	100.00	186.17

表 5.9　測定環境

OS	Windows7 64bit OS
CPU	Core i5-3570K 3.40GHz
メモリ	8.00GB
IDA（統合開発環境）	Microsoft Visual Studio Express 2013 for Windows Desktop

5.4.1　考察

提案法は従来法に比べて約 1.86 倍速いという結果になった．これはやはり計算時間のかかる非線形関数を除去し，乗算を XOR 演算に置き換えたからであるといえる．従来法が掛け算回数最大 42 回に対して，提案法は乗算回数 21 回，XOR 演算回数 27 回と従来法に比べて乗算回数が約半分に減ったことが理由であると考えられる.

第 6 章

総括

6.1　まとめ

本研究では，ランダム性を維持しつつ高速に暗号化可能な新たなカオス暗号システムとして連立ボルテラフィルタに論理演算を用いたカオス暗号を提案した．特性評価の結果を順に見ていくと，まずリアプノフスペクトラムではほぼすべてのパラメータが第 1 と第 2 リアプノフ指数が正の値を示し，第 3 リアプノフ指数が負の値を示した．これは提案したカオス暗号システムがハイパーカオスであることを示している．ランダム性の検証においては，従来法よりも高い結果となり，合格率が 70 %を超えたパラメータも従来法の倍になった．また，本研究の目的である暗号化処理速度の検証では従来法よりも大幅に改善され，約 1.86 倍の処理効率を得ることができた．これはやはりコンピュータが計算コストの高い，乗算よりも論理演算の方が早く計算できるということである．しかし，係数感度の検証においては従来法と同じ暗号鍵の鍵長となった．暗号鍵として有効なパラメータの割合で比較すると従来法よりも悪い結果と言える．こちらは今後の課題としたい．係数感度の検証結果はあまり良いものではないが総合的に判断しても，論理演算をカオス暗号に用いることは有効であり，実験から提案法のカオス暗号システムは高いランダム性と高速な暗号化速度を兼ね備えていると証明した．よって，今後はこの手法を取り入れた新たな暗号システムの開発に期待できるだろう．

6.2　今後の展望

今回，提案したカオス暗号システムは総合的に判断してみても従来法より優れたものとなっているが，実際にシステムとして実現するにはまだ多くの課題が残されている．以下にその課題をまとめる．

第 1 式は複雑化すべきではない

係数感度の結果から，第 2 式のパラメータの方が第 1 式よりも良い結果を示している．経験上，第 1 式をいくら複雑化してもあまり結果が良くなることはなかった．なので，今後この研究を行う者は第 2 または第 3 式を複雑化することをおすすめする．なお，その際に非線形性を保てているかをしっかりと検証しなければならない．

XOR 演算がランダム性を生み出す根拠

数学的な根拠に基づいて，XOR 演算が乗算よりも複雑性を生み出す理由をさらに検討すべきである．

計算効率の向上

今回，時系列からリアプノフ指数を測定した．測定には 2 日近くかかりかなりの計算量を必要とした．そこで GPU を用いた並列プログラミングを行い，計算効率の向上を目指すべきであると思う．しかし，計算効率が向上するということは暗号を解読しようとする者に対しても同じ条件となりうるので，この点は対策が必要である．

ハードウェアへの実装

実際に FPGA 等に搭載し，特性を評価することで実際に運用が可能なのか確かめる必要がある．

周期性の確認

暗号としてどのくらいの強度があるのか確かめるためにカオス暗号の周期がどのくらいであるかを確かめる必要がある．

アトラクタの確認

カオス暗号の式が生み出すアトラクタを確認し，リアプノフ指数だけに頼らず，本当にカオス性があるのかどうかさらなる検証をするべきである．

適切なパラメータの選定方法の確立

今回用いた従来法，提案法のパラメータは以前から設定されてきたものであったが，違うパラメータで検証したとき，その検証結果は全く異なるものになってしまう．本来ならば，すべてのパラメータの組み合わせを試し，一番最適なものを設定するべきなのだが，その組み合わせは膨大なものとなり天文学的な時間がかかるため今の技術では検証することが事実上不可能なものとなっている．だからこそ暗号として強いとも言えるが，検証していない以上は脆弱性の原因ともなり得る．このパラメータの選定方法の確立をしなければならない．

参考文献

[1] H. Kamata, Y. Umezawa, M. Dobashi, T. Endo, and Y. Ishida. Private communications with chaos based on the fixed-point computation. Trans. IEICE, Vol. E83-A, No. 6, pp. 1238-1246, Jun 2000.

[2] K.Iwata, T.Nakamura, and H.Kamata. Chaotic modulator with volterra filter for cipher. IEICE, Proceeding of NOLTA, pp. 216-219, Sep 2007.

[3] 岩田一馬，入倉裕之，鎌田弘之．ボルテラフィルタを用いたカオス秘密通信に関する研究．第 20 回 回路とシステムワークショップ

[4] 川西義明，渡辺大貴，中山有洋，佐藤泰智，吉田大希，鎌田弘之，”非線形関数と Volterra フィルタを組み合わせたカオスシステムの研究” 電子情報通信学会技術研究報告. vol.110, No.465,pp47-52,NLP2010-171,2011.

[5] H,Watanabe, T.Sato, A.Nakayama, D.Yoshida, Y.Kawanishi, and H.Kamata, ”Study on the characteristic analysis and evaluation of digital chaotic system operated by the fixed point arithmetic”, Proceedings of 2012 RISP Internatinal Workshop on Nonlinear Circuits, Communications and Signal Processing, vol.5PM1-1/2, ppp.377-380,2012.

[6] 鶴岡泰明，田中大貴，尾崎泰孝，鎌田弘之，固定小数点演算により発生する特性の関数近似とカオス暗号に与える効果に関する研究．第 27 回 回路とシステムワークショップ

[7] Y. Kawanishi, D. Yoshida, H. Watanabe, A. Nakayama, T. Sato and H. Kamata. ”Chaotic Modem System using Nonlinear Map with Simultaneous Volterra Filters”, JSST2011, 2011.

[8] H.Watanabe, D.Yoshida, A.Nakayama, T.Sato, Y.Kawanishi, and H. Kamata. Parameter setting for chaotic cipher using simultaneous volterra filters. on the 24th workshop on circuits and systems in Awaji, pp. 22-a1-2-2, Aug 2011.

[9] 結城浩 著，暗号技術入門 不思議の国のアリス，2008 年

[10] JON ERICKSON 著，村上雅章 訳，HACKING:美しき策謀 脆弱性攻撃の理論と実際 第 2 版，2011 年

[11] 合原一幸, 池口徹, 山田泰司, 小室元政. カオス時系列解析の基礎と応用. 産業図書, 東京, 2000 年

[12] M.Sano and Y.Sawada, “Measurement of the Lyapunov spectrum from a chaotic time series,” Physical Review Letters, Vol 55, No. 10, pp.1082-1085, 1985.

[13] G. Marsaglia. Diehard, 1998.

謝辞

本研究において，多大なるご指導とご鞭撻を賜りました明治大学理工学部電気電子生命学科教授 鎌田 弘之 先生に心より厚く御礼申し上げます．

さらに，研究の助言だけでなく，学会の付き添いもしていただきました，明治大学理工学部電気電子生命学科助教 神山 恭平氏に深く感謝致します．

また，本研究を進める上で多くの助言や相談に乗っていただいたカオス班の後輩 鶴岡 泰明君に心より感謝致します．

そして，本研究を行うにあたり多くの助言を頂いた，電気電子生命学科 複合情報処理研究室の大学院生諸氏，及び 4 年生諸氏にこの場を借りて感謝致します．

最後に 18 年間にも長きにわたり学生生活を支えてくださった両親に感謝致します．

2015 年 2 月　吉日

明治大学大学院　理工学研究科　電気工学専攻

複合情報処理研究室

2 年 51 組 61 番