カオスとニューロ、分散処理

―現実世界の複雑性とコンピューティング―

豊橋創造大学　情報ビジネス学部学部　今井正文

１．はじめに

一般的なイメージとして、コンピュータは非常に高速に緻密な計算ができると考えられている。高価なスーパーコンピュータなどを用いれば色々なシミュレーションができ、色々な問題を解いてくれる、ある意味で万能な機械のような考えられ方をしている。確かに高速なコンピュータにプログラムすれば、普通の人間ではできないような相当に細かい計算や繰り返し計算が可能である。

一方、普通の一般家庭におけるパソコンの使われ方は、ワープロ、表計算、電子メール、インターネットブラウザを使って調べ物、CD、DVD 鑑賞、さらにはデジタルカメラやデジタルビデオの画像編集、データベースを使って顧客管理や DM、ホームページでの情報発信、ネット販売に応用している場合もある。色々なソフトをインストールしておくと 1 台のコンピュータで色々な機能を果たしてくれるという意味で万能というか多機能で便利だが、何が高速でどのあたりで何万回も計算してくれているのか分かりにくい部分がある。

コンピュータのイメージと実際の利用が異なったり混乱する原因はいくつかあるが、基本的にコンピュータの使い方には大きく何種類かあるにもかかわらずあまり説明されていない事がある。コンピュータの高速な計算能力は元々数値計算やシミュレーション、制御等に使われていたのだが、個人利用のためのパソコンとソフトウエアが発展・普及する段階で利用方法が変わったせいでもある。そういっても現実の世界の問題を、ふと思いついて自宅のパソコンでシミュレーションしてみようと考える人は多くないでしょうけれど。

本日は、この本来の使い方ではあるが一般的ではないコンピュータの使い方、いわゆるシミュレーションについて考えていく。特に、現実問題にコンピュータを利用しようと思ったとき最初にぶつかる壁であるモデルの概念、最近話題になっている複雑性の科学、ソフトコンピューティングといわれる計算手法について順番に考えていく。

最終的には、自分の疑問に思う現実の問題をコンピュータに与え、どうやったら何がおきるのかコンピュータが自動的にシミュレーションして結果を出してくれる。人間は、結果を比較検討しながらさらに良い方法がないか、考えるコンピュータにさらに指示を出すというような使い方について述べる。個人的には、漠然とした相談をしたらプロの人間のようにコンピュータが色々なパターンについて考えて言葉で教えてくれる、SF に出てくるようなコンピュータならもっと便利なのだが。

2. 現実世界とコンピューティング

2.1 コンピューティングの基礎

現実の問題をコンピュータ上で解決しようと考えた場合、色々な問題が出てくる。プログラミングの知識がある学生であっても、その知識とコンピュータを使って何ができるのか良く分かっていないというかやるべきことが分からない学生がいるが、原因は基本的な部分をすっ飛ばしているせいである。まずは、基本的な用語を整理しながら、全体を概観していくことにする。

コンピューティングとは、狭義では計算する、広義ではコンピュータを利用して何らかの処理をすることだから、パソコンを使って行なうことは何でもコンピューティングということになる。では、シミュレーションとはなにか。一般的には、「複雑なシステムの仮想実験（狭義ではコンピュ

ータ上で)を行うこと」という定義になっている。ここでいうシステムは以下のように解釈しておく。「システムとは、（全体として特定の目的を果たす、）相互に結合した多数の構成要素の有機的結合体」コンピュータ上で何らかの問題を扱うという場合、（シミュレーション）モデルを用いて、仮想実験や計算を通して結果を得ることになる。

2.2 モデルとは

システム工学的にいうと現実世界の何かしらの問題を扱う場合には、（厳密でないとしても）モデルを用いて考える。基本は入出力の関係なのだが、以下の様に考えたい対象に対して単純化して関係を整理して考えている。環境を外部システムと対象システムを内部システムという。

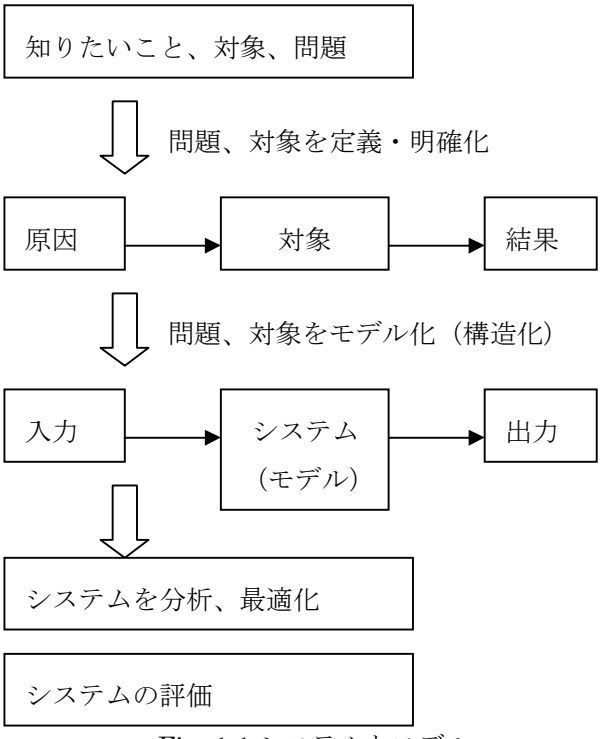


Fig. 1.1 システムとモデル

問題がすぐに単純化できない場合は、問題を部分問題に分割してサブシステム（部分システム）を作り、サブシステムを統合して全体システムを得る。（要素還元的な考え方ともいう。）

2.3 順問題と逆問題

問題には、順問題と逆問題という分類もある。入力を与えて出力を求める演算的な問題を順問題、結果から原因を探ることを逆問題を解くという。コンピュータに応用することを考えると重要な概念で、一般に逆問題は難しくなる。不正確だが簡単な例として足し算を例に以下のように考えるといいかもしれない。

順問題

問題：1+2=?

解答：当たり前ですが 1+2=3 で、解は 1 個。（この手の計算はコンピュータの得意分野）

逆問題

問題：?+?=3

解答：0 も含めた正の整数だけなら解の候補は 4 個。

0+3=3, 1+2=3, 2+1=3, 3+0=3

少数を含めると解けなくなる（解の候補が無限になる）。

0+3=3, 0.01+0.29=3, 0.02+0.28=3, …, 1+2=3, …, 2+1=3, …, 3+0=3, …

負の整数を含めると解けなくなる。

1000+(-997)=3, 999+(-996)=3, ... 4+(-1)=3, 3+0=3, 1+2=3, 2+1=3, 3+0=3, 4+(-1)=3, …

一見ばかばかしい話だが、解けないことは確かである。逆問題は、条件や情報があれば解けることがある。上の例の場合、正の整数だけとか、情報があって 1+?=3 なら解けます。コンピュータも逆問題は苦手なので工夫が必要になる。

2.4 モデル化の問題点

また、現実世界を解こうと考えた場合、モデル化そのものにも気をつける必要がある。我々は、多くの人や物事から影響を受け、また、多くの人や物事に対して影響を与える。システムを構成する要素同士が互いに影響を与えるつまり相互作用することによりシステム全体の挙動は決定される。つまり複数の構成要素が相互作用することが重要、言い換えると要素間の関係が重要。

モデル化の問題点
重要でない部分は簡単化して大丈夫か？
サブシステムは統合して全体システム（現実を映したもの）になるか？
基本的な解答
相互作用（要素間の関係）が線形（足し算）ならば大丈夫。関係が非線形（掛け算）の関係は大丈夫とはいえなくなる。

3. 非線形が問題になる理由（複雑性の科学）

線形ならば、答えがあるかどうか求めるのは比較的簡単。（機械であれば安全であると保障できる。微分方程式なら、安定、安定限界、不安定の 3 種類.。）

非線形は大部分の問題が解けない。（解が存在するか分からない、機械であれば安全かどうかいえない）。非線形部分（掛け算の成分）は、値が小さいときは無視できるが、ある程度の大きさになると非常に大きな値になってしまう。カオスと呼ばれる難しい現象が生じやすい。

3.1 方程式の話（線形の場合）

連立方程式：答えは時間で動かない

$\begin{cases} y = x \\ y = -x + 2 \end{cases}$ 答え $\begin{cases} x = 1 \\ y = 1 \end{cases}$

微分方程式：時間 t で答えが動く（物の動きを表現）

$\frac{dx}{dt} = 50$ 積分すると答えは $x = 50t + c$

c=0,t=1 なら x=50

（意味：時速 50km/h の車は、最初の地点を 0 とすると 1 時間後に 50km の場所に移動する）

方程式の話（線形の連立微分方程式の場合）

時刻 *0* における初期状態(initial state) を *x(0)*、時刻 *t*>0 における *x*(*t*)を時間に従って状態空間に表示した軌跡をトラジェクトリ(trajectory)と言う、特に他の軌道を引き込む性質を持つ軌道を特にアトラクタと呼ぶ。連立微分方程式だとアトラクタは、様々な形が存在するが基本は、安定点、安定限界（周期軌道）、不安定点の 3 つである。

安定（平衡点）:ある一点に収束するトラジェクトリを平衡点と呼ぶ。
安定限界（周期軌道）：ループ上になるトラジェクトリの事を周期軌道(periodic orbit)
不安定（不安定平衡点）：ある一点より、発散するトラジェクトリ

以下では、ファン・デル・ポール(van der Pol)発信器より非線形特性を取り除くいたものを例に、安定（平衡点）、安定限界（周期軌道）、不安定（不安定平衡点）を見てゆく。

オリジナル

$$\begin{cases} \frac{dx}{dt} = -y \\ \frac{dy}{dt} = x - i \end{cases}$$

ただし、$i = f(y) = ay(y^2 - b) \quad (a, b > 0)$

非線形部分を取り除いたもの

$$\begin{cases} \frac{dx}{dt} = -y \\ \frac{dy}{dt} = x - ay \end{cases}$$

安定（平衡点）：任意の初期状態からトラジェクトリは原点に巻き込まれる。

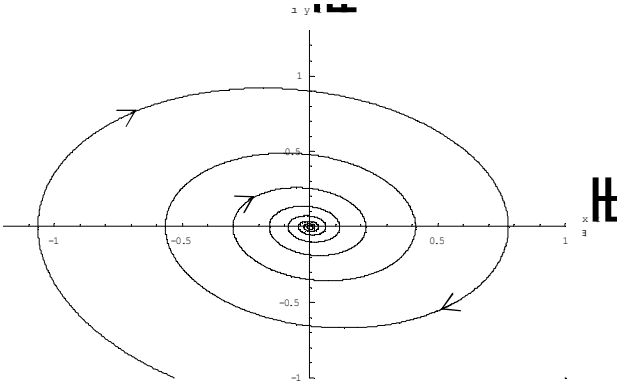


Fig.3.1 安定平衡点(収束)
(*a*=0.2,*b*=-1,*x*(0)=1,*y*(0)=1)

安定限界（周期軌道）：

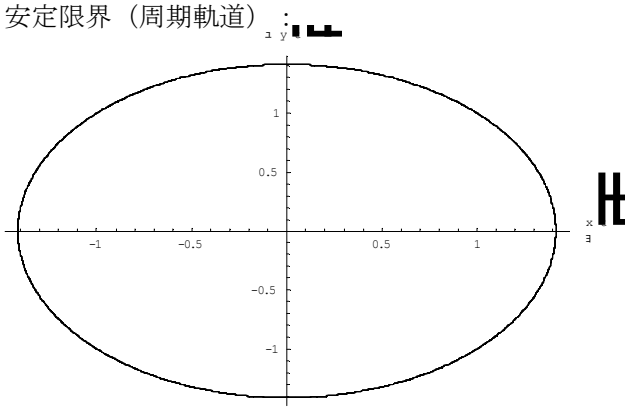


Fig.3.2 安定限界（周期軌道）

(a=0,b=1,$x(0)$=1,$y(0)$=1)

不安定（不安定平衡点）：原点近傍から出発したトラジェクトリは巻き出している。

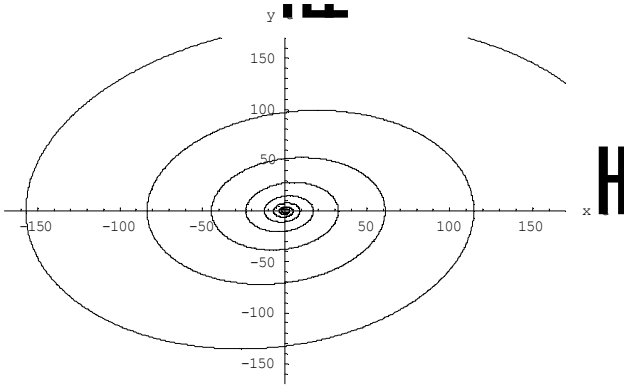


Fig.3.3 不安定平衡点(発散)

(a=0.2,b=-1,$x(0)$=1,$y(0)$=1)

4. 非線形とカオス理論

連立微分方程式が非線形の項を持つと、理論的に解けない上にカオス性を持つ場合がある。「カオスとは規則に従って発生しているにも関わらず、不規則にみえる振る舞いを示す現象の事である。」もっと簡単に言えば、「物事は繰り返すが、全く同じことは二度と起こらない。」となり、自然界にもよく見られる現象です。（従来、自然の中の不規則な現象は規則に従わない確率的な物だと考えられてきたが、一見不規則に見える現象の中に規則に従って発生しているカオスという現象がある。）このように時間と共に複雑な挙動を示す現象をカオスと言い、このカオス特性を持つ時系列データを特にカオス時系列データと言う。

4.1 カオス（非線形）システムを解く

レスラー方程式(Rossler equation)は、式(4.3)の微分方程式で与えられる。非線形性は第 3 式の zx のみの単純な非線形性を有するシステムである。

$$\begin{cases} \dfrac{dx}{dt} = -y - z \\ \dfrac{dy}{dt} = x + ay \\ \dfrac{dz}{dt} = bx - (c - x)z \end{cases} \tag{4.3}$$

Fig 4.1 は、x 軸に $x(t)$、y 軸に $y(t)$、z 軸に $z(t)$ として 3 次元状態空間へプロットしたものである。アトラクタは、z = 0 で平面近傍に拘束されながら原点を中心に渦ができており、時折 z 方向に上がりながら原点の方向へ折り返す運動を繰り返し作成される。視覚的には、貝殻のようなアトラクタとなる。

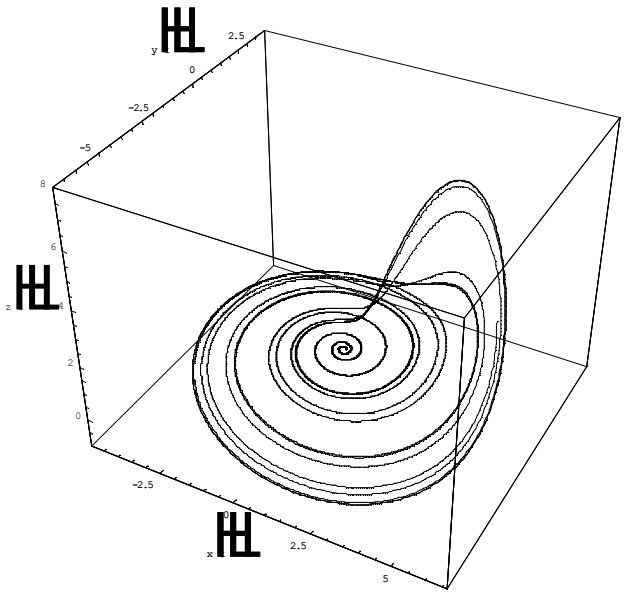


Fig. 4.1 レスラー方程式のアトラクタ

(a = 0.36,b =0.4,c= 4.5,Time=0～100,⊿t=0.02)

カオス（非線形）システムを解く（同定と予測）

非線形は大部分の問題が解けない（難しい）。制御も難しい(機械であれば安全かどうかいえない)。

できるだけ全要素を観測しないと正確に解析、同定、予測できない。カオス現象（複雑な動き）が生じやすい。⇒一般的な解析的な方法では対処が難しい。

4.2 カオスと短期予測

ここで、対象となる時系列データがカオス性を持つ場合、そのデータの振る舞いは、決定論的な法則に従っている事が考えられる。もし、その非線形的な決定論的規則性を推定する事が出来れば、規則性を失うまでの近未来なら予測する事が可能である。ただし、カオスの特徴の一つである初期値による鋭敏な依存性により、決定論的因果性を失うまでの短期予測しか行えない事に注意が必要である。

また、少ないデータからの推定にはアトラクタの再構成を利用する事が出来る。ある n 次元の力学系の任意の時刻の状態は、n 個の状態変数によって記述することができる。それぞれの状態変数をすべての時刻で状態空間内にプロットすることにより、システムの状態変化を知ることができる。しかし、実際に n 個の状態変数をすべて同時に観測することは大変困難である。最悪、たった 1 変数の時系列データのみしか観測できないということも有り得る。しかし、このような場合でも、計測された 1 変数の時系列データのみから、我々が直接見ることの出来ない本来の力学系に関するデータを得ることができ、アトラクタを観測された時系列データのみから再び構成するという意味で、アトラクタの再構成(reconstruction)と呼ばれる。具体的には、1 変数の観測値 $x(t)$を用いて、時間遅れの大きさを τ と設定する。そこから、式(4.1)のような m 次元による時間遅れベクトルを作成する[9]。

$$v(t) = (x(t), x(t+\tau), \cdots, x(t+(m-1)\tau)) \quad (4.1)$$

この時間遅れベクトル $v(t)$を m 次元状態空間にプロットすることにより、再構成アトラクタを得ることができる。F.Takens によれば、式(4.1)の遅れ時間座標を用いた観測時系列データから再構成状態空間への変換が、再構成次元 mと元の次元 n が $m \geq 2n+1$ であれば、この変換が元のアトラクタの埋め込み(embedding)となることが証明されている。再構成されたアトラクタが元のアトラクタの埋め込みになっている場合、再構成された状態空間内のベクトル $v(t)$と元の状態変数が微分同相となる。これはつまり、再構成されたアトラクタは、元のシステムと同じ動特性を持っていることを意味する。

Fig4.1 の $x(t)$を遅れ時間 τ =10 として再構成し、x 軸に $\tilde{x}(t-2\tau)$、y 軸に $\tilde{x}(t-\tau)$、z 軸に $\tilde{x}(t)$ として状態空間表示したものを Fig4.3 に示す。

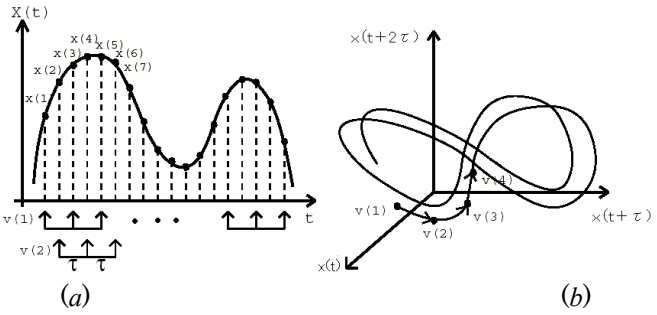


Fig. 4.2 時間遅れ座標系による 1 変数時系列からのアトラクタの再構成

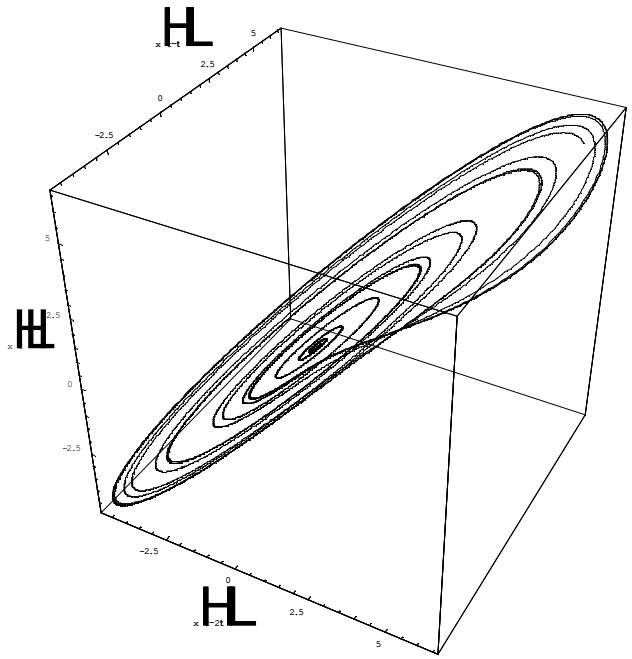


Fig. 4.3 レスラー方程式の再構成アトラクタ
(τ =10⊿t, x 軸に $\tilde{x}(t-2\tau)$,y 軸に $\tilde{x}(t-\tau)$,z 軸に $\tilde{x}(t)$ とした場合)

5. ソフトコンピューティング

コンピュータのスピードや機能の発展に伴って、コンピュータの利用法も広がっている。設計、管理、予測、データベースシステムなど分野によってその使われ方はさまざまであり、使われるアプリケーションも多岐にわたっているが、基本的に人間の作業を支援または代行しようとするものであることは共通している。ところが、コンピュータを利用して何か問題を解こうとするとき、色々と困った現象が起きることが知られている。特に難しいのが以下のようなタイプの問題であるといわれている。

「規模が小さければ人間にも解けるが、規模が大きくなると実質的に解けなくなる」

「理論上はある計算手順で解けることが分っているが厳密には実用的な時間で解くことができない」

具体的な問題としては、巡回セールスマン問題の他にスケジューリングの問題、自動設計などが有名である。これらの問題は、基本的には厳密に正しい解（厳密解、最適解）を求めることができてもそれが実用的ではない問題である、逆にいえば、それに近い値（近似解、満足解）を求めることしかできない問題であるともいえます。

コンピュータは人間が定めた動作を高速に繰り返すという目的から開発され、発展してきたが、現代においてもスピードは十分と言い切れない部分がある。一方、人間の代わりに考えるようなコンピュータ（例えば人工知能、人工生命）を期待する部分がある。ソフトコンピューティングはこれらに答えようとする方法のひとつと考えられている。ソフトコンピューティングに属する理論を簡単に説明すると以下のようになる。

ハードコンピューティング　→　厳密な計算（計算時間やコストが増大する傾向がある）

ソフトコンピューティング　→　過度の厳密性を避ける事により実用性を向上させる。

代表的な理論：ファジィ理論、ニューロコンピューティング、遺伝的アルゴリズム、etc

ファジィ理論：人間の言語のような曖昧さを取り扱う事ができる（柔軟な制御などが得意）

ニューロコンピューティング：神経回路網の応用、学習や予測を行うことができる。理論的には何でも学習可能

遺伝的アルゴリズム（GA：Genetic Algorithms）：生物進化のメカニズム利用、解の発見等に有効

カオス理論：自然界にみられる複雑な現象を取り扱う。非線形理論

5.1 神経回路網とニューラルネット

ニューロ（以下ニューロと呼ぶ）とは、人間の脳を構成する神経回路網をニューロンとそのつながりであるニューロとしてコンピュータ上に再現し、問題に適用しようとする新しいアルゴリズムである。長所としては、与えられた数値データから学習や認識を行う。非線形関係や対象の構造が未知でも学習できる可能性を持つことが挙げられる。（応用例：パターン認識、音声認識や文字認識、家電製品等の制御など。）短所としては、学習や認識の過程、または学習した結果の構造の解析が困難である。つまり、ニューロの持つ非線形関係や対象の構造が未知でも学習できる能力を用いて、カオス時系列データの予測を行うことを考える。（詳細な説明はパワーポイントの資料を参照）

5.2 遺伝的アルゴリズム

遺伝的アルゴリズム（GA:Genetic Algorithms）とは、遺伝と自然淘汰という生物学的な仕組みからヒントを得て考え出された探索手法（答えを探すための方法）の一つである。「遺伝的アルゴリズムとは、ダーウィンの進化論の概念を用いて、厳密に求める事ができない問題の答えを見つけようとするアルゴリズム（計算の手順）である」と言い換えることもできる。

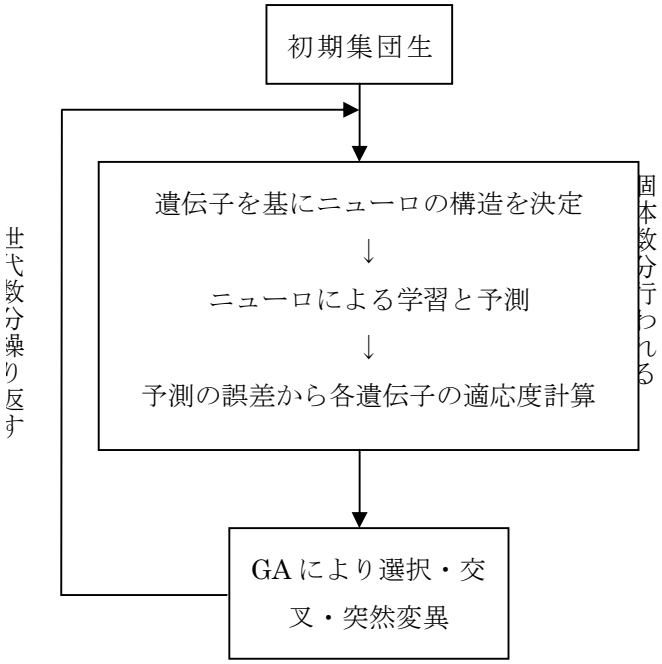
ここでは、「各都市間の距離が設定された n 個の全ての都市を 1 回ずつ訪れるものとしたとき、最小距離で巡回する経路を探す」という巡回セールスマ

ン問題（TSP:Traveling Salesman Problem）を用いて、遺伝的アルゴリズムの有用性について示していく。この TSP は、もちろん都市の数が少なければ人間でも、少々数が多くてもパソコンを使えば解けるが、都市の数が大きくなると計算量が膨大になって正確に解くことができない問題として有名である。（詳細な説明はパワーポイントの資料を参照）

5.3 GA-ニューロ

ニューロは、与えられた数値データから非線形関係や対象の構造が未知でも（つまりカオスでも）学習できるが、入力層や中間層の数によって（つまりはニューラルネットの構造）によって性能が変化してしまう問題がある。

一方、GA は遺伝と自然淘汰という生物進化のメカニズムを利用して、コンピュータ上で擬似生命を繁殖させて答えを探す。つまり、生物の適応のように、条件に適応した解が見つけることが得意であった。ここでは、これらを互いに補完するものとしてニューロのパラメータ調整や構造決定を GA で行なう GA-ニューロという考えについて述べていく。



5.3.1 直接予測の結果

まず、データに対して直接、ニューロを適用する。15 期を入力とし、1 期先の予測を行った。ニューロのパラメータを Table.5.1 に示す。10 回予測を行い得られた結果から誤差の平均値と、10 回の試行において誤差が最良となった結果を Table.5.2 に、最良となった予測値のグラフを Fig.5.1 に示す。

Table.5.1　直接予測のニューロのパラメータ

入力層のユニット数	中間層のユニット数	出力層のユニット数	学習回数	学習係数
15	10	1	1000	0.1

Table.5.2　誤差の平均値と最小値

	平均値	最小値
Rmse	0.06757	0.05884

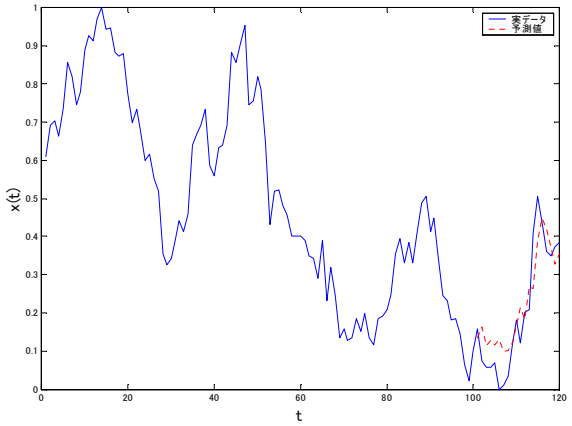


Fig.5.1　直接予測における実データと予測値

5.3.2 再構成空間における予測の結果

3 次元に再構成を行い、再構成空間内での 5 期を入力とし、1 期先の予測を行う。ニューロのパラメータを Table.5.3 に示す。

結果の良かった遅れ時間 τ=11 の、再構成空間における実データと予測値のグラフを Fig.5.2 に示す。Fig.5.2 より 1 期先の予測値を取り出した場合の実データと予測値を Fig.5.3 に示す。

Table.5.3 再構成空間におけるニューロパラメータ

入力層のユニット数	中間層のユニット数	出力層のユニット数	学習回数	学習係数
15	10	3	1000	0.1

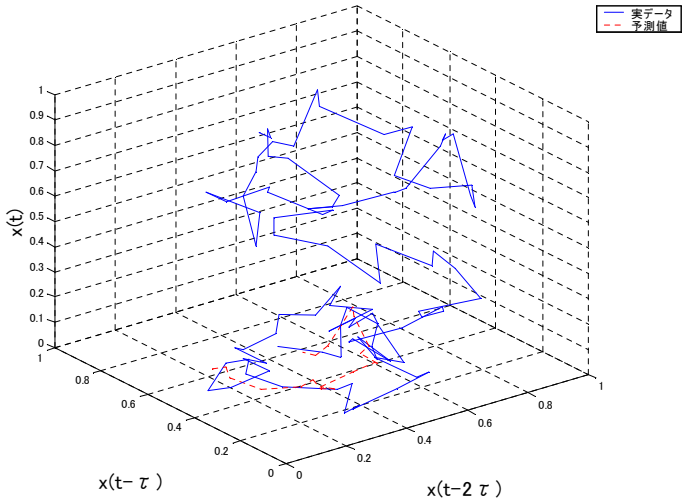


Fig.5.2　遅れ時間 τ =11 の場合の実データと予測値

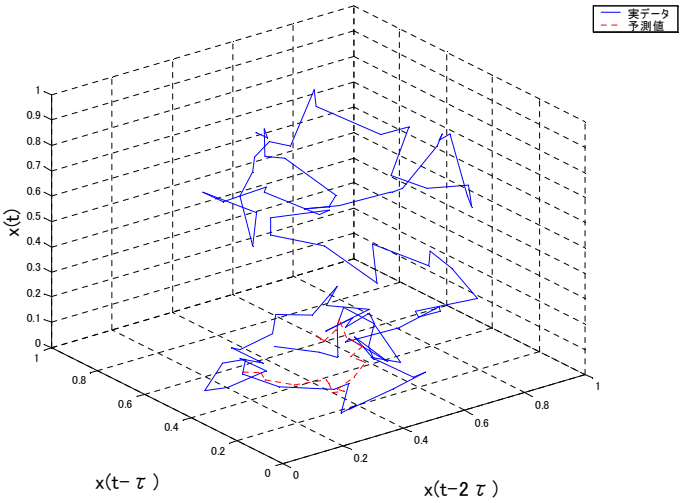


Fig.5.4 再構成空間における予測値
（学修回数 1000、遅れ時間 τ =11）

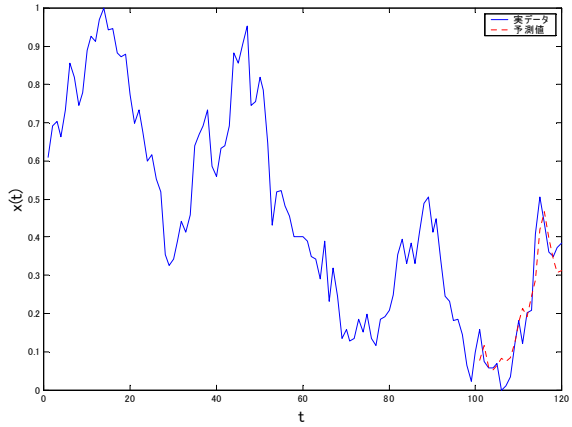


Fig.5.3　遅れ時間 τ =11 の場合の実データと 1 期先の予測値

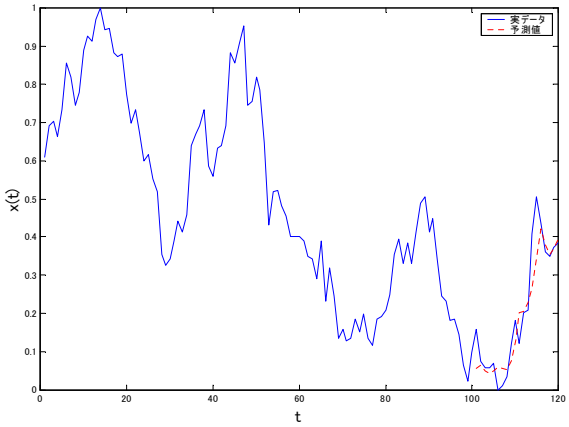


Fig.5.5 実データと 1 期先の予測値

5.3.3 GA-ニューロによる予測の結果

GA-ニューロによって予測した場合、最も結良かった再構成空間における予測値と 1 期先の値を学習回数ごとに Fig.5.4,5.5 に示す。

GA-ニューロの計算結果をまとめたも Table.5.4 に示す。比較のために次元数と遅れを変化させながら予測を行なったものを Fig.5 に示す。Table.5.4 より、直接予測や GA を利用い再構成空間における予測に対して改善していとがわかる。特に Fig.5.6,5.7 との比較よりその遅れ時間や空間の次元数に対して、ニューロ造を最適化できていることがわかる。

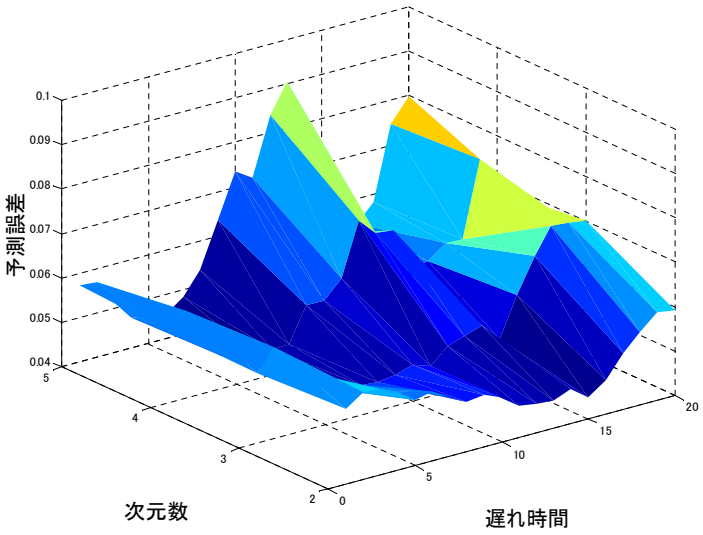


Fig.5.6 次元数と遅れ時間による誤差(学習回数 1000)

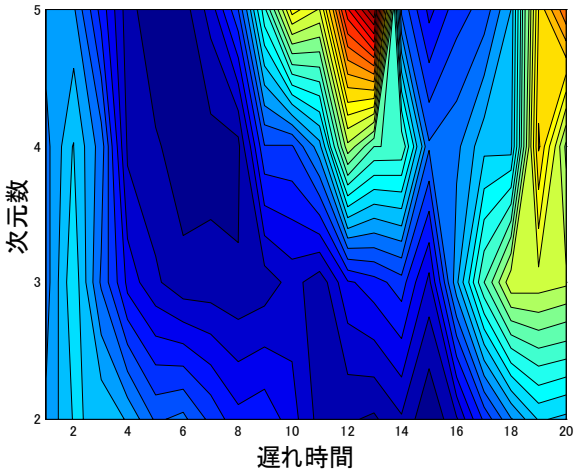


Fig.5.7 次元数と遅れ時間による誤差

Table.5.4 再構成空間における予測(学習回数 1000)

遅れ時間	最良の個体の最良値			最良個体平均誤差
	入力層	中間層	誤差	
1	39	3	0.05254	0.05605
2	27	9	0.05843	0.06079
3	48	10	0.05117	0.05497
4	48	10	0.04728	0.05019
5	39	7	0.04330	0.04814
6	45	16	0.04271	0.04681
7	39	10	0.04392	0.04691
8	48	6	0.04503	0.04659
9	18	16	0.04354	0.04703
10	27	11	0.04623	0.04841
11	27	5	0.04141	0.04695
12	18	14	0.04715	0.05048
13	18	4	0.04688	0.05156
14	12	7	0.04857	0.05299
15	9	3	0.04199	0.04885
16	3	15	0.05615	0.05772
17	3	9	0.06335	0.06546
18	18	6	0.06730	0.07103
19	12	3	0.06814	0.07074
20	9	6	0.06573	0.07014

6.多段システムと分散処理

Ant System は M.Dorigo らによって提案された ACO メタヒューリスティクスに基づくアルゴリズムであるが、Ant System において ant は組み合わせ最適化問題のような分散的な空間で並列的に行動するエージェントである。フェロモンの揮発性の特徴を利用して最適経路を維持しつつ動的な変化に適応するモデルが構築できる。TSP に対する Ant System では、複数の ant を各都市に配置し次の都市までの距離と経路に置かれているフェロモン量に従って、未訪問都市から確率的に選択しながら全ての都市を巡回して経路を生成する。

この節では TSP における Ant System のパラメータ設定の難しさを解決するために、Ant System のパラメータを GA で探索する GA-Ant System を提案し、Ant System パラメータを最適化することを考える。GA-Ant System の動作イメージを Fig.6.1 に示す。

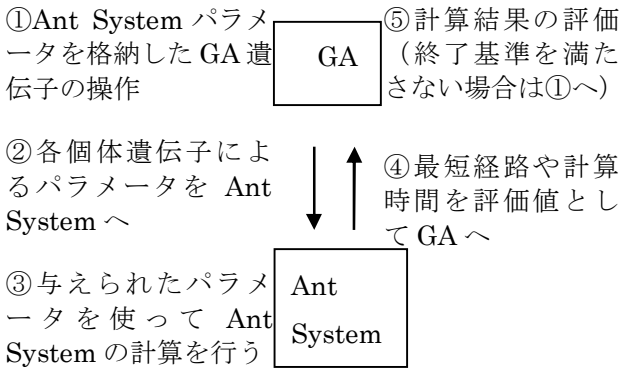


Fig.6.1 GA-Ant System の動作イメージ

6.1 分散処理

分散処理には様々な方式があるが、その一つであるグリッドコンピューティングとは、インターネット等の広域ネットワークを通じて複数のコンピュータ資源を 1 つの仮想的なコンピュータとして扱うものである 。

グリッドコンピューティングでは、ネットワーク経由で接続されたコンピュータを役割によりブローカーノードと計算ノードと呼ぶ。ブローカーノードは問題をジョブに分割し計算ノードに割り当てる。計算ノードは割り当てられたジョブの計算を行い、結果をブローカーノードに返す。グリッドコンピューティングの動作イメージを Fig.6.2 に示す。

前節の Fig.6.1 と Fig6.2 を比較すると、システムの構造が類似しており、GA-Ant System は分散処理化し易いことが分かる。多段システムは他にもあるが同様に分散処理化することが出来る。

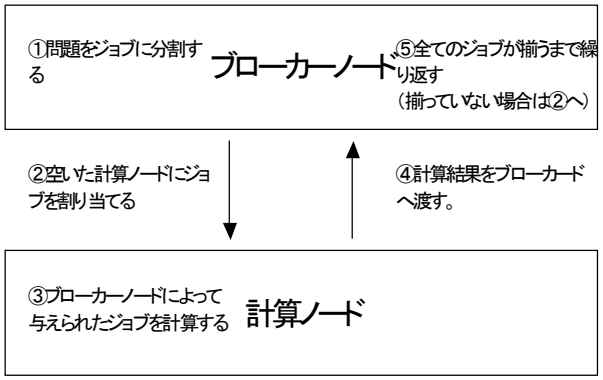


Fig.6.2 分散処理のイメージ

100 都市（10×10 のグリッド都市）の TSP に対して GA-Ant System を適用し、さらに GA-Ant System の処理を分散させ最短経路の探索を行う。このときの GA を使って変動させるパラメータ、またその探索範囲を Table6.1 に示す。GA-Ant System のパラメータを Table6.2 に示す。1 台から 4 台の計算ノードを用いた場合の 10 回ずつの処理時間の平均を Table6.3 に示す。

Table6.1 パラメータの探索範囲

	ant 数	P	Q	α	β
数	100～400	0.2～0.8	1～4	1～4	1～4
刻み	100	0.2	1	1	1

Table6.2 GA-AS のパラメータ

GA				AS
遺伝子数	遺伝子長	世代数	突然変異率	ステップ数
20	10	5	0.5	24

Table6.3 処理時間の平均

台数	1 台	2 台	3 台	4 台
平均	14127.7	8086.6	5368.1	4558.7
比率	100%	57%	38%	32%

7.コンピューティングと適応

生物は、全体を理解して一番良いものを選んでいるわけではない。実際の生物は、それぞれに限られた情報をもとに個々が良いと思うもの、自分に適していると判断した方法に従って行動しているだけである。結果的にたまたま状況に適した個体が生き延びていくことになる。つまり、個々が多様であるがゆえに全体として適応していけるといえる。変化する環境では、今良い事が次も良い事である保障はない。構成要素自体も変化し続けなければならないといえる。このような環境では、GA-ニューロのような機能に加えて、(1)多数が相互作用することが重要、(2)多様でなければ適応できない、(3)構成要素は相互作用で変化する、などの能力が必要になると考えられる。具体的には（マルチ）エージェントシステムが適用可能ではないかといわれている。

8.まとめ

経営情報のような複合的分野では対象自体が複雑で、従来の方法では計算できない場合がある。特に非線形な場合は、モデルの構造化そのものが難しく、カオス特性を持つ場合は色々な意味で注意が必要となる。伝統的なハードコンピューティング（厳密な解を求めるための計算）に対して、ソフトコンピューティングという概念が提案されている。理由は、現実の世界が不精密、不確実であり従来の方法ではうまく計算できないことによると考えられている。複雑なものをそのまま考える「複雑性の科学」やソフトコンピューティングが現在のところ適用可能で有用な手法とされている。また、人工生命やエージェントシステムという分野も適応と言う観点から、将来的には（今現在進行形で）有用な技術であるといえる。

参考文献

[1] 大内東、山本雅人、川村秀憲、柴肇一、高柳俊明、當間愛晃、遠藤聡志：「生命複雑系からの計算パラダイム」森北出版 pp.8-41(2003)

[2] Masafumi IMAI: 「Consideration Concerning Application of Ant-System in Path Planning Problem」流通と情報(Japan Society for Logistics and Information), Vol.3, No.1, pp.9-16 (2007)