組織の進化アルゴリズムモデル

荒木孝治

1．序

1980年代後半以降，組織やネットワークといった複雑な特性を持つシステムを積極的にコンピュータを用いて分析・制御する領域が開発されてきた。この流れが様々な分野で受容されるに従い，カオスや複雑性といった用語が一般的なことばとして広く定着しつつある。

旧来の研究では理論的な解析可能性が重視され，ともすれば分析対象やその枠組みに対して強い制約が置かれる傾向にあった。それに対して上記の動きの特徴的なところは，コンピュータを集中的に利用し，シミュレーション技術を援用することにより，より制約の少ない，現実的な問題設定のもとで自由度の大きい分析が可能になることにある。そして，それが近年のコンピュータの能力の飛躍的な向上と相俟って大きく開花してきた。社会科学の分野でも人工知能・ニューラルネットワーク・進化アルゴリズムといったツールや，マシンラーニングといったそれらを包括する関連諸分野の実りある結果をどん欲に取り入れて様々な研究やシステムへの実装が行われている[1)]。

Simon (1996) の論考をまつまでもなく，経営学においても経営現象にお

1）たとえばSimon (1996) は，自然物・現象を研究する科学に対比して「人工物」の科学（the science of the artificial）を定義し，ここで記述した理論やツールの見取り図を与えている。

ける複雑性を分析することは大きな意味を持つ。組織やそれらの柔軟な結合体であるネットワーク組織は，制約のある資源の中で情報や物をやりとりしながら日々学習し，進化するダイナミズムを持つ複雑なシステムだからである。組織は，進化だけでなく，エージェント，創発性，分散化，不確実性，協力，合理性の限界，不完全な情報，満足原理といった特性によって彩られている[2)]。

現在，複雑性を視野に入れて，進化や学習といった組織やシステムのダイナミックな特性を分析に取り入れることのできるツールとして，カオス，ゲーム，セルオートマトン，ニューラルネットワーク，進化アルゴリズムといった諸理論がある。本稿では，これらのうち，進化アルゴリズムと呼ばれる研究領域で開発されてきた理論・ツールを組織の学習や進化の分析に取り入れたとき，経営学においてどのような視野が開かれるかを試みとして提示したい。多数の経済・経営エージェントが相互作用する中で複雑な特性を発現する複合体としてシステム・組織をとらえ，進化アルゴリズムを応用したシミュレーションモデルが示す振る舞いが，現実の組織が持つ特性にかなり近いことを明らかにする。

進化アルゴリズムの代表的なものに遺伝的アルゴリズム (Genetic Algorithms) と遺伝的プログラミング (Genetic Programming) がある。これらをシミュレーションに利用することにより，合理性に限界を持つエージェントの集合体として組織を表現し，それが複雑な環境のもとで自ら，および組織が置かれている環境条件を学習しながら漸進的に進化するというモデリングの新たな可能性の領域を開拓することが期待できる。社会科

2）これらの複雑性を解明するために欠くことのできない方法として，モデルの構築から解析に至るまでの一連のプロセスにコンピュータを集中的に利用する方法であるコンピュテーショナル・アプローチある。従来，経営学において複雑な特性を持つシステムに関する分析が困難であった原因の一つに，分析者が共通に持つことのできるツールや言語の不在，すなわち数学的あるいはコンピュータにもとづく分析ツールの不在を挙げることができるが，この問題もこのアプローチを用いることによりある程度解消することが期待できる。

学の分野に進化アルゴリズムを適用した先駆的な研究としてArifovic (1994) やChen and Yeh (1996), Edmonds and Moss (1996) がある。

進化アルゴリズムを用いたモデリングでは，エージェントたちは環境とのインタラクションの中で，環境に対する信念あるいはモデルを形成しながら進化する。そのとき組織は，将来，モデルを発展させることが可能なように，多様な，時には互いに矛盾するようなモデルまでをも資源として保持することが可能となる。さらに，現実のエージェントがそうであるように，このアプローチにおいては帰納と演繹のメカニズムがうまく統合される。

本稿の構成は次のようになる。第2節で進化アルゴリズムを概説する。第3節で遺伝的プログラミングにもとづくエージェントのモデル化の基本的な考え方を提示し，さらに，Arifovic (1994) による遺伝的アルゴリズムにもとづく組織進化モデル，Chen and Yeh (1996) による遺伝的プログラミングにもとづく組織進化モデルを概説する。第4節で，進化アルゴリズムによる組織モデリングを用いてダイナミックに変化する環境情報を推定するシミュレーションを行ない，それがどううまく機能するかを調べる。

2．進化アルゴリズム

生物進化のメカニズムは遺伝子と呼ばれる記号列に埋め込まれている。そして進化は自然選択および遺伝子の自己複製や突然変異から生じる。進化アルゴリズムは，現実の生物に見られる進化のこの特性を数理的なシミュレーションモデルに取り入れるものである。ある問題解決の行動において，問題の解に対応する遺伝子の集団を構成し，それらに対して遺伝的操作を繰り返し適用することによって，よりうまく環境に適合するようにその集団を進化させるという手続きを持つ。

例えば，関数：$y=g(x)\,(-1\leq x\leq 1)$ を x に関して最大化する問題を考えてみよう。厳密な方法として解析的に解くものがある。これを実行する

には例えば関数の微分可能性といった条件が必要である。しかし現実の問題では，1）考えるべき変数の数が多い，2）関数が多峰性を示す，3）微分可能でない，といったイレギュラーな特性のために解を陽に求めることができないのが一般的である。しかし，進化アルゴリズムはこのようなイレギュラーな問題にも対処可能である。基本的な考え方は次のようになる。

とりあえずいくつか解の候補を定め，その集団を考える（最初，それらをランダムに発生させる)。当然，目的関数を最適化する解がこの初期集団に含まれることはまずないが，これらに自然選択と遺伝的操作を繰り返し適用し，数多くの世代にわたってコンピュータの中で進化させることにより，最終的には“かなり良い”解（かならずしも最適なものではない）を求めることができるというのが進化アルゴリズムの基本的考え方である。このとき“良さ”の測度として問題に対応した適合度という指標を考える。現実の生物の進化は通常途方もない時間を必要とするが，コンピュータの中の人工物の進化は“現実的な”時間内で終了する。

2．1　遺伝的アルゴリズム

遺伝的アルゴリズムは John　Holland により1960年代に考案された。Holland (1975) は様々な集団の各構成要素をある固定長の記号列で表現し（通常は 0 または 1 という記号による 2 進数表記を用いる)，それに対して生物における染色体の役割を与えた。各記号列（染色体）が持つ環境に対する適合度を計算し，その適合度に基づいて一種の自然選択を行う。そして選択された記号列に対し，交叉や突然変異等の遺伝的操作を加えて新しい記号列を生成する。このステップを繰り返すことにより記号列の集合を進化させる。

この手順をもう少し詳しく見てみよう。様々なバリエーションがあるが，基本的なプロセスは次のようになる（Mitchell（1996）参照)。

手順1．長さ l の記号列(染色体)x_iを n 個ランダムに発生させ，集合 $X=\{x_1, x_2\cdots, x_n\}$を構成する(普通，$n$ は50～2000個にとる)。各 x_i が問題の解の候補である。

手順2．集団内の各記号列の適合度 $f(x_i)$ を求める。

手順3．以下のサブステップを n 個の子孫の集団（次世代集団）が新しく生成されるまで繰り返す。

3-1　X から二つの記号列をランダムに選択する（これらの記号列を両親と呼ぼう)。選択される記号列は重複してよい。このとき，記号列の選択の確率 P_sは適合度に比例するように定める。

3-2　ランダムに決定された位置で二つの記号列をある確率 P_cで交叉させ，二つの新しい記号列，つまり子孫を構成する。

3-3　子孫を構成する記号を小さな確率 P_mで変化させる。

手順4．新しく生成された集団で古い集団を置き換える。

手順5．手順2に戻る（最終世代になるまで，あるいはある基準を満たすまでこれを繰り返す)。

この手順を各世代の記号列の集団に繰り返し適用する中で，適合度が高いという性質が次世代の集団に受け継がれていくことになる。

手順3-1から3-3はそれぞれ，選択（selection)・交叉（crossover)・突然変異（mutation）という遺伝的操作に対応する。交叉はランダムに選択した二つの記号列の一部を交換する操作である。まず記号列の集団から選択確率 P_sにもとづいて二つの記号列をランダムに選択する。次に，$\{1, \ldots, l-1\}$からランダムに決定した数 k に対して，第 k 遺伝子座(記号列の左から k 番目のビットの位置を第 k 遺伝子座と呼ぶ) の右側の文字列を入れ替える。全部で $n/2$ 個（n は偶数とする）のペアを選択し，交叉確率 P_cで交叉させる。例えば次に示す二つの記号列

11111111　　00000000

が第4遺伝子座で交叉する場合，新しい二つの記号列

11110000　　00001111

を得ることになる。

突然変異は遺伝子座の値をランダムに変更する操作である。ある確率 P_m で各遺伝子座の値を突然変異させる(他の遺伝子座とは独立に行う)。突然変異が生じると，記号は1ならば0に，0ならば1に変化する。たとえば記号列：11110000の第7遺伝子座で突然変異が生じると，11110010となる。

2.2　遺伝的プログラミング

遺伝的プログラミングは遺伝的操作に関して遺伝的アルゴリズムと基本的に同じであるが，進化の対象が記号列の集団ではなく一種のプログラムから構成される集団であるという点で異なる。ここではプログラムが遺伝子の役割を果たす。Koza (1992) は“正しい”プログラムを自動的に生成する問題から出発して遺伝的プログラミングを発展させた。プログラムは本質的には関数と考えることができるので，遺伝的プログラミングでは進化する集団は関数の集まりと考えてよい。

コンピュータ言語の一つであるLISPではプログラムをS式と呼ばれる形で表現する。S式は一般には(関数　式$_1$…式$_n$)という形を持つ。S式を構成するのは関数記号(例えば，＋(加算)，－(マイナス)，＊(乗算)，／(除算)，*AND* や *OR* 等の論理演算，…)および終端記号(変数や定数等)である。関数記号と終端記号を総称してノードという。上記の式$_i$は終端記号でもS式でも良い。例えばC言語における $2*(a-b)$ は，関数記号 {*, －} と終端記号 {2，a，b} から構成されており，S式では

(*　2　(－　a　b))

と表現できる。S式と同値な表現としてグラフの木構造がある。上式に対応する木構造は図1のようになる。

関数記号と終端記号をうまく選択し，問題の解をS式や木構造で定義することにより，遺伝的プログラミングを用いて様々な問題にアプローチす

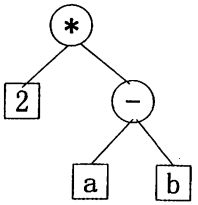

図1　木構造の例

ることができる。例えば，Koza (1992) は，シンボリック回帰 (Symbolic Regression)，最適制御問題，創発的集団行動等に関する遺伝的プログラミングの適用事例を報告している。

遺伝的プログラミングにおける交叉は木構造の枝部分の交換であり，突然変異はノードの変更である。交叉の例を図2に示す。

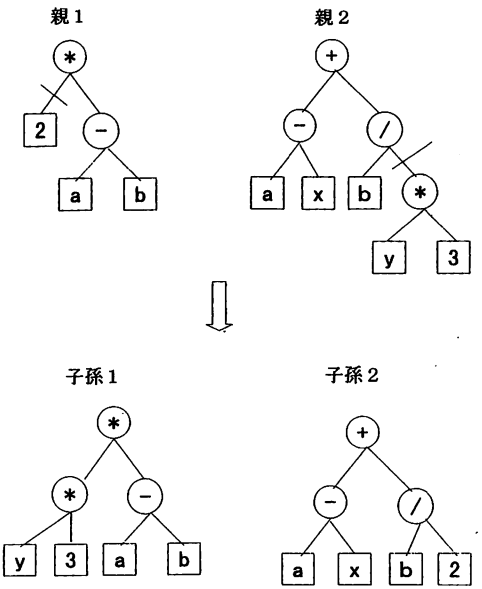

図2　交叉の例

3．組織進化モデル—進化アルゴリズムにもとづいて

第2節で見た進化アルゴリズムを組織進化のモデリングに利用するには，組織が学習すべき環境条件，組織が解決すべき問題の解を記号列や関数の形でうまく表現できればよい。特に本稿で中心的に扱う遺伝的プログラミングでは，環境および組織パラメータの選択，関数記号および終端記号の選択，適合度の評価を適切に行う必要がある。

では，ある組織が，自らが置かれている環境に関するモデル・信念とも言うべきものを記号列や関数で表現することが可能であろうか。これは知能とはシンボルを操作する能力であるということを考えると可能である。たとえば，組織の内部構造や情報構造，組織が保持する資源は組織が積極的に変更・選択できるパラメータであり，記号による表現が可能である。それらのパラメータ値とともに，環境条件を計量変数・層別変数として特定しておき，そのときの組織行動の良し悪しを収益等の様々な結果変数の値として計測すれば，組織行動に対する環境の応答に対応して，記号列としての組織パラメータを進化アルゴリズムにより進化させることができる。組織は，組織パラメータを記号列またはS式で表現し，環境との相互作用の中で，適合度という環境からの応答を通じて環境情報を次世代の組織パラメータとなる遺伝子の中に取り込むわけである。

進化アルゴリズムを組織進化のモデリングに適用するとき，組織および組織を構成するエージェントにいわゆる限定された合理性という特性を持たすことができる。組織は遺伝子の長さ l および数 n という限られた情報を保持するのみである。環境の探索も各世代で n 回行うだけで，マクロな適合度情報を必要としない。有限個の遺伝子の適合度を情報として環境より得るのみであり，それを利用して学習する。よって進化アルゴリズムによって組織を表現すると，エージェントは，環境に関する完全な情報を持たず，環境との相互作用を通じてのみ環境情報を探索し，学習することに

なる。さらに，このモデリングでは，現実のエージェントにみられる次の諸特性を考慮することも可能である（Edmonds and Moss（1996））。

- 行動を決定するときに演繹のメカニズムよりも学習のメカニズムが優位となる。
- 学習は可能なモデルのうち最適なものをすべて求めようとはしない。それは漸進的であり，軌道依存的（path-dependent）あるいは探索的である。
- しばしば矛盾するモデルや信念を保持する。

進化アルゴリズムを社会科学の分野へ最初に適用したMiller（1986）以降，いくつかの研究が見られるが，本稿では特に，遺伝的アルゴリズムにもとづくもの（たとえばArifovic（1994）），遺伝的プログラミングにもとづくもの（たとえばEdmonds and Moss（1996），Chen and Yeh（1996））に注目する。ArifovicおよびChen and Yehはエージェントの学習・進化のシミュレーションにおいてくもの巣モデルの枠組みを採用した。次項では，くもの巣モデルを紹介し，両者が適用した進化アルゴリズムの手法を概説する。

3．1　くもの巣モデル[3)]

競争市場にいくつかの企業があり，同質財を同じ技術で生産しているとする（財の価格は同一）。生産には時間がかかるので，各企業は市場価格が決定される前に次期の生産量を決定する必要がある。生産量を決定するために，t期の期首に推定した価格をp_t^eとする。t期の期末に財は実際に決定された価格p_tで販売される。

適応的期待価格の決定法の一つに

3）Day（1994）による。Arifovic（1994）らはここで説明するモデルより限定的なものを考えたが，後の利用を考えて一般的な形で述べておく。

$$p_t^e = p_{t-1}^e + w(p_{t-1} - p_{t-1}^e)$$

とするものがある。これは，予測値と実際の値とのずれで，次回の予測値を調整するという適応的なものである。w が 1 のときには，前期の価格をそのまま次期の期待価格として投影する方式となる。ここでは簡単のために $w=1$ として話をすすめる。

S_iを企業 i の供給関数とすると，企業は期待価格 p_t^eと供給関数にもとづいて，

$$y_{i,t} = S_i(p_t^e)$$

としてその期の生産量 $y_{i,t}$を決定する。総供給量を Y_tであらわすと，適応的期待のもとでは

$$Y_t = S(p_t) := \sum_{i=1}^{n} y_{i,t} = \sum_{i=1}^{n} S_i(p_t)$$

となる。需要関数を $D(\cdot)$とし，その逆関数 $D^{-1}(\cdot)$が定義できるとする。供給量が与えられると，均衡価格は供給量で表現でき，

$$p_t = D^{-1}(Y_t)$$

となる。よって，価格に関する次の差分方程式を得る。

$$p_t = F(p_{t-1}) := D^{-1}[S(p_{t-1})]$$

3．2　Arifovic のモデル

Arifovic (1994) は，くもの巣モデルの枠組において，遺伝的アルゴリズムを用いて次期の生産と売り上げに関する企業の決定ルールを進化させるモデルを提案した。そして遺伝的アルゴリズムを学習スキームとして用いると，くもの巣モデルで従来よく用いられてきた他の学習アルゴリズム(合理的期待，過去の価格の標本平均，過去の価格の回帰等) と比べて，価格が合理的期待均衡にうまく収束することを示した。また，遺伝的アルゴリズムが人間の主観に関する実験的な行動が持ついくつかの特徴をうまく把握していることも示した。Arifovic (1994) のモデル (以下，Ar という) は次のようになる。

一つの企業があり，それが価格を決定するために n 人のエージェントを擁するとする。エージェント $i(i=1, \cdots, n)$ は独自の価格決定ルール $A_{i,t}$ を持っており，それを用いて生産量を決定する。このモデルではその決定ルールの集合 $A_t=\{A_{i,t}: i=1, \cdots, n\}$ を遺伝的アルゴリズムによって学習・進化させる[4]。

$A_{i,t}$ はアルファベット {0, 1} を用いた2値文字列であり，これが生産量に対応すると考える。実際には，文字列を数値（整数）化し，それを調整した値を生産量とする。記号列は0と1から構成されているので，それを2進数と考えることにより整数への変換は簡単にできる。長さ l の文字列（決定ルール）i：

$$\{a_{i,t}^1, a_{i,t}^2, \cdots, a_{i,t}^l\}$$

を生産量に変換するのは次式により行う。

$$q_{i,t}=\sum_{k=1}^{l} a_{i,t}^k 2^{k-1}/\overline{K}$$

ここで，$a_{i,t}^k$ は第 t 期における記号列の第 k 座の記号（0または1）であり，$\overline{K}$ は正規化するための係数である[5]。$q_{i,t}$ を決定ルール i が示唆する t 期における生産量と考える。記号列 i の適合度 $\mu_{i,t}$ は t 期に実際に得た利潤：$\mu_{i,t}=p_t q_{i,t}-C_{i,t}$ によって定める（$C_{i,t}$ は生産に要するコスト）。

2.1項で述べた遺伝的アルゴリズムの手順に従って，記号列の集合は選択・交叉・突然変異により更新される。選択は，高い適合度を持つ記号列は高い確率で子孫を作り出すことができるように適合度をもとに行なわれる。記号列 $A_{i,t}$ が選択される確率は，たとえば

$$P_s(A_{i,t})=\mu_{i,t}/\sum_{i=1}^{n}\mu_{i,t},\quad i=1, \cdots, n$$

4）ここではArifovic(1994)の“集団が一つの場合の遺伝的アルゴリズム”を述べる。Arifovic(1994)は“多集団遺伝的アルゴリズム”の場合も扱っている。

5）ここでいう正規化とは，ある一定範囲の実数値，すなわち現実的な生産量に変換することを意味する。よって係数の取り方は問題によって異なる。

とすればよい[6]。

3．3　Chen and Yeh のモデル

Chen and Yeh (1996) は，遺伝的プログラミングによる学習モデルを提示した。彼らによるとこのモデル (以下，CY という) は遺伝的アルゴリズムにもとづく Ar に対して次の諸点で優れていると考えられる。

1．Ar では学習するエージェントは価格の上限を知っていると仮定しているが，CY ではその仮定を必要としない。

2．Ar では突然変異を外生的にシャットアウトする操作が導入されているが，CY ではこの操作を採用していない。人間の主観に関する実験において，十分に均衡点に近づいた後でも突然変異と考えられる行動が観測されることを考えるとこれはより現実的な設定である。遺伝的プログラミングにもとづくエージェントから構成される経済は自己を安定化させる特性を持つので，その仮定を必要としないともいえる。

3．CY では Ar モデルに比較してより不安定な場合にも対処可能であると考えられる。

GP_tを第 t 期に企業が保持する決定ルールである価格予測関数 $gp_{i,t}$の集合とする。

$$GP_t=\{gp_{i,t},\ i=1,\ \cdots,\ n\}$$

エージェント i はルール $gp_{i,t}$を用いて生産量を決定する。$t-1$ 期までの情報を Ω_{t-1}であらわすと，これにもとづいて t 期の予想価格である

$$p^e_{i,t}(\Omega_{t-1})$$

を求めることになる。CY では，よりよい予測関数を遺伝的プログラミング

6）再生のこの方法をルーレット方式という。他にトーナメント方式やエリート戦略等様々な方法が提案されている。それは計算の効率を考えるとともに，適合度の高い記号列の再生の強さと集団における多様性の保持との間にバランスをとることを工夫するためである (Mitchell (1996) 参照)。

によって進化させる。たとえば連続した過去 h 期の価格データ $\{p_{t-1}, \cdots, p_{t-h}\}$ を用いて p_t を予測する関数 f：

$$p_t^e = f(p_{t-1}, \cdots, p_{t-h})$$

を求めることができる。ここで h はエージェントまたは組織の能力に相当するパラメータの一つと考えることができる。

GP_t に対して選択・交叉・突然変異という遺伝的操作を適用することによりこれは次世代の GP_{t+1} へと進化する。企業 i の適合度 $\mu_{i,t}$ は予測にもとづいて生産した場合の利益をもとに評価する。

4．組織パフォーマンスのシミュレーション

これまで述べてきたように，本稿で提案する進化モデリングでは組織をエージェントの集合体と考え，組織目標を達成するために各エージェントは決定ルールを持って自律的に行動しているとする。組織全体のパフォーマンスは各エージェントの決定能力に依存する。絶え間なく変動する環境の中では，エージェントは環境からの情報に従って決定ルールを進化させる必要がある。本節ではその進化のメカニズムに遺伝的プログラミングを用い，そのパフォーマンスをシミュレートする。

Arifovic (1994) および Chen and Yeh (1996) は，彼らの進化アルゴリズムモデルが均衡値に収束するかどうかに重点をおいてシミュレーションを行ったが，本稿ではモデルの構造そのものを推定することを考える。すなわち遺伝的プログラミングによりダイナミクスを生み出す関数形そのものを推測することを試みる。Arifovic らのくもの巣モデルではこの関数形は多項式という簡単なものであったが，ここでは推定により困難な関数形を考える。また，ダイナミクスもカオスを発生させる複雑な場合を考える。

4．1　カオス的選好モデル

Benhabib and Day (1981) は，合理的な選択の結果としてもカオス的な振る舞いがあらわれることを例証するために次のようなモデルを提示した。

二財モデルを考え，各財の消費量を x と y とする。コブ＝ダグラス型効用関数を仮定すると各個人が得る効用は

$$U(x,\ y)=x^a y^{1-a}$$

となる。ここで a は効用の重みパラメータで，過去の消費に依存して決定されるとする。Benhabib and Day (1981) は，a が 1 期前の各財の消費量に

$$a_{t+1}=g(x_t,\ y_t\ ;\ \alpha)$$

という形で依存すると仮定した。

m を各期における個人の支出額とする。上記の関数 g が

$$g(x_t,\ y_t\ ;\ \alpha)=\alpha x_t y_t$$

のとき x 財に関する差分方程式

$$x_{t+1}=F_1(x_t\ ;\ \alpha,\ m):=\alpha m x_t(m-x_t) \tag{1}$$

が得られる。αm^2が 3 を越えて大きくなるとこのダイナミクスではカオスが発生することがわかっている。

また，

$$g(x_t,\ y_t\ ;\ \alpha)=x_t e^{\alpha(1-x_t)}$$

とするとき，差分方程式

$$x_{t+1}=F_2(x_t\ ;\ \alpha,\ m):=\alpha m x_t e^{\alpha(1-x_t)} \tag{2}$$

が得られる。$m=1$ に対して，$\alpha\sim 2.6924$のときカオスが発生する。各方程式のダイナミクスの例を図 3 および 4 に示す[7]。

7）式(1)および(2)は May and Oster (1976) が扱った生態学の分野における古典的なモデルと同じである。

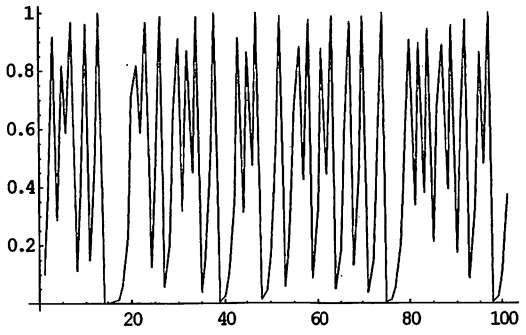

図3　F_1のダイナミクス
($a=4$，$m=1$，$x_0=0.1$，$t\leq 100$の場合)

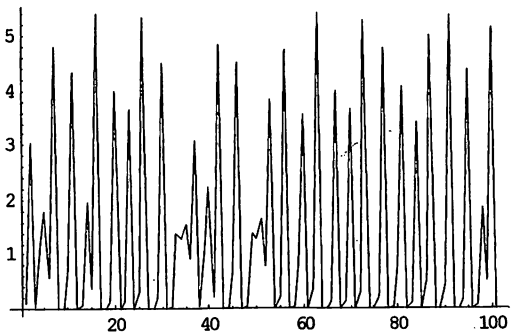

図4　F_2のダイナミクス
($a=2.6924$，$m=1$，$x_0=0.1$，$t\leq 100$の場合)

4．2　カオス的くもの巣モデル

Hommes (1994) は，カオスが発生するくもの巣モデルとして次のような設定を考えた。1）需要関数は線形とし[8]，2）供給関数はS字曲線とする。2）を考える根拠は，初期コストと固定費のため価格が低いとき供給

8）$D(x)=a-bx$，$b>0$

はゆっくりと増加し，供給とキャパシティの制約のため価格が高くなりすぎても供給はゆっくりと増加することになる，という考察にもとづく。そして，これらの仮定に対応する差分方程式として，

$$x_{t+1}=F_3(x_1)$$

ただし，

$$F_3(x\,;\lambda,\ a,\ b,\ w)=-\arctan(\lambda x)/b+(1-w)x+aw/b \quad (3)$$

を考えた。このとき，たとえば，$x_{t+1}=F_3(x_t\,;4.8,\ 0.75,\ 0.25,\ 0.3)$のダイナミクスは図5のようになる。

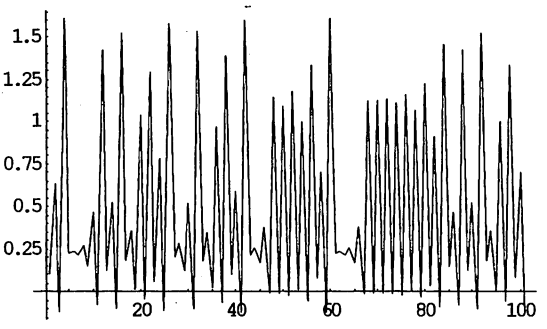

図5　F_3のダイナミクス
($a=0.8,\ b=0.25,\ \lambda=4,\ w=0.34,\ x_0=0.1,\ t\leq 100$の場合)

4.3　シミュレーション結果[9)]

4.1，4.2節で述べたモデルに対して遺伝的プログラミングにもとづくシミュレーションを行う。遺伝的プログラミングにもとづいたエージェントがカオス的に変化する情報を生み出すメカニズムをどのようにうまく推定するかを見る。その概略は次のようになる。

差分方程式の初期値x_0を0.1とする。組織を構成するエージェントは，各ダイナミクスからの$n=40$組のサンプルデータ$\{(x_i,\ x_{i+1}),\ i=1,\ \cdots,\ 40\}$

9）シミュレーションにはLISPインタープリタの上でKoza(1992)のプログラムを利用した。

にもとづいて，差分方程式を生み出している関数の関数形自体を遺伝的プログラミングを用いて推定するというタスクを実行する。関数記号の集合は非常に簡単な {+, −, *, %} [10]とし，終端記号は実数および変数 X，選択確率は0.1，交叉確率は0.2とする。また，進化の期間は50期とし，集団の大きさは100と定める[11]。

関数記号の多様さ，サンプルデータの大きさ，集団の大きさが組織の能力に対応する組織パラメータである。よって，ここではかなり限定された能力の組織・エージェントを扱っていると考えることができる。結果は次のようになった。

(1)式の推定結果

シミュレーションの結果，カオス選好モデルの第一の式(1)の場合，そのようなダイナミクスを生み出すメカニズムであるモデル式 F の全世代にわたる最適な推定式は次であった[12]。

```
(* X (+ (− (− (% X (− (* X (+ X X))
(* −2.81945351875362 −4.75385408837062)))
(+ −3.95973398767399 X)) X) (* X −1.86724236554803)))
```

この関数のグラフを図6に示す。図6—(iii)より，遺伝的プログラミングを用いて得た推定式がモデル式をほぼ完全に再現していることがわかる。

(2)式の推定結果

シミュレーションの結果，選好サイクルモデルの第二の式（2）の場合，

---

10) %は次の除算をあらわす。

(% a b) = a/b if b ≠ 0, = 1 otherwise

11) この期間の数および集団の大きさはKoza(1992)らが提唱する数よりかなり少ないものであるが，ここではシミュレーションの目的がモデルの有効性を実証することにあるため，数の大きさは重要でない。

12) 出力結果をそのままS式で表示する(以下同様)。

ダイナミクスを生み出すモデル式 F の全世代にわたる最適な推定式は次であった。

(% (− X (+ X (+ (+ X X) X)))
(− (* X (* X (* (+ X(− −4.40460171057126
−4.93944156446468)) −0.801892325189845)))
(% (* −1.02335632593527 (% (% X X) −1.55246664609409))
(+ (% X X) 4.70111397546768))))

この関数のグラフを図 7 に示す。図 7 −(iii) より，2 次式の場合と比べ

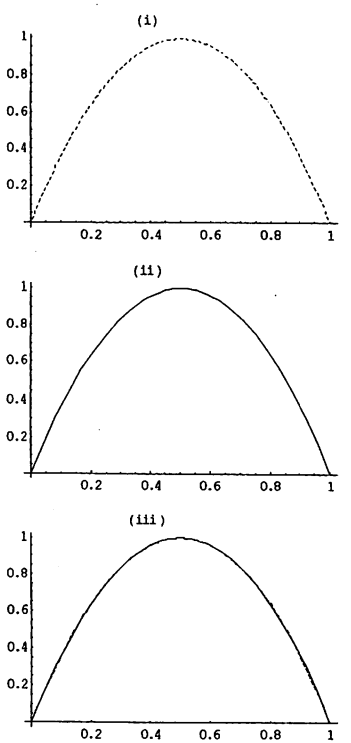

図 6　$F_1(x\,;\,4,\ 1)$ の推定

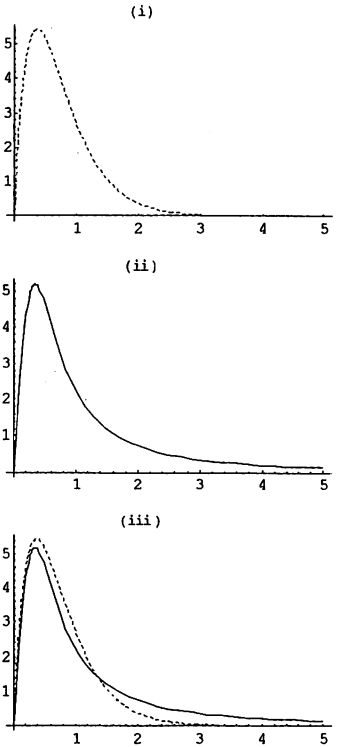

図 7　$F_2(x\,;\,2.6924,\ 1)$ の推定

(i)モデル式　(ii)推定式　(iii)モデル式と推定式の合成図

て差分方程式の関数形が複雑になった結果，推定式の説明力がある程度落ちていることがわかる。

(3)式の推定結果

Hommes (1994) の考えたくもの巣モデルのダイナミクスにおいて，モデル式 F の全世代にわたる最適な推定式は次のようになった。

```
(% (- (% X (* X 0.694564034088778))
        (* (% X(+ (% X X)(* (% (+ X (* (+ X X)(+ X X)))
        (- X(+ X -3.35543372126084))) X))) 4.7528074028682))
(- (+ (- (* (% X(+ (- -1.19046856006164
        -2.93008200262211)
    X))(- X (+ (-2.07187115078414 -0.786335070518001) X)))
      1.16313206784573)(* 4.79851496862178 X))
        -3.31998713701963))
```

この関数のグラフを図8に示す。図8より，現在のシミュレーション条件のもとではモデルをほとんど説明できていないことがわかる。そこで，関数記号を {+, −, ＊, %, SIN, COS} に増やし，他の条件は同一にしてもう一度シミュレーションを行うと次の結果を得た。

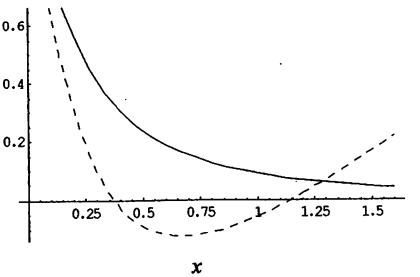

図8　$F_3(x;\ 4,\ 0.8,\ 0.25,\ 0.34)$ の推定(1)
モデル式（波線）と推定式（実線）

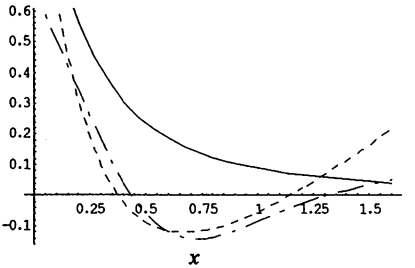

図 9　$F_3(x;\ 4,\ 0.8,\ 0.25,\ 0.34)$ の推定(2)
モデル式（波線）と新しい推定式（一点鎖線）

(COS (+ (SIN (− X (SIN 3.32574851267307)))
(COS (SIN (− (− (+ 2.44754044220203 X)
(+ X 1.45948707426874)) (+ (SIN X) X))))))

このグラフを図 8 に加えたものを図 9 に示す。図より，関数記号を増やしたことによりかなり適合度が増加していることがわかる。

以上は，各ダイナミクスに対して遺伝的プログラミングによるモデリングが与える最終結果を示したものである。本来は，進化プロセスの各世代におけるエージェントの振る舞いを分析する必要があり，それは可能であ

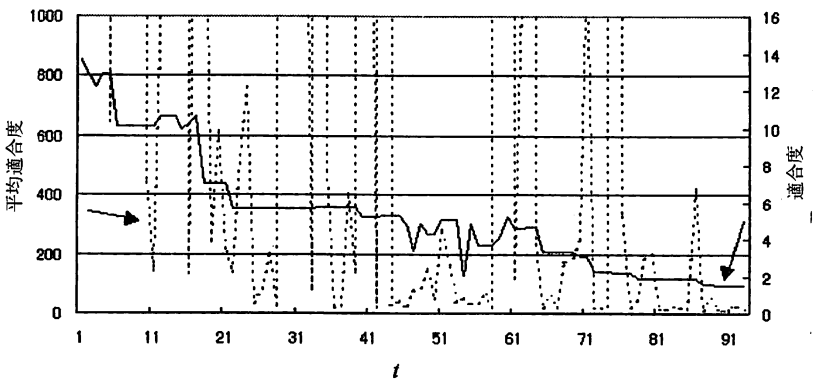

図10　適合度の推移
波線：各世代の平均適合度（Y軸）
実線：各世代の最適適合度（第2Y軸）

るがスペースの都合により，ここでは世代の平均的な動きおよび，最適なものの動きのみを与えておく。

図10は，各世代にわたっての集団の動きをみるために，(1)式の場合のシミュレーションにおける各世代の集団の平均適合度と最適適合度を示したものである。なお，適合度は非負であり，値は小さい方がよい。図の各世代の最適適合度値をあらわす実線の動きを見ると（縦軸は右側の第２Ｙ軸），最適な決定ルールは着実に進化してきていることがわかる。それに対して波線があらわす集団の平均適合度の動きも全体としてはほぼ進化していることがわかる（極端に悪い決定ルールが含まれると，平均値が異常に増加するため顕著には傾向が見られない）。

参考文献

Arifovic, J. (1994). Genetic algorithm learning and the cobweb model. Journal of Economic Dynamics and Control 18, 3-28.

Chen, Shu-Heng and Chia-Hsuan Yeh (1996). Genetic programming learning and the cobweb model. in : P. J. Angeline and K. E. Kinnear, Jr. eds. Advances in Genetic Programming Volume 2, The MIT Press.

Day, R. H. (1994). Complex Economic Dynamics Volume I. The MIT Press.

Edmonds, B. and S. Moss (1996). Modeling bounded rationality using evolutionary techniques. CPM Report No. 96-10 http://www.cpm.mmu.ac.uk/

Hommes, C. H. (1994). Dynamics of the cobweb model with adaptive expectations and nonlinear supply and demand. Journal of Economic Behavior and Organization 24, 315-335.

伊庭斉志（1996），『遺伝的プログラミング』，東京電機大学出版局。

Koza, J. R. (1992). Genetic Programming. On the Programming of Computers by Means of Natural Selection. MIT Press.

May, R. M. and G. F. Oster (1976). Bifurcations and dynamic complexity in simple ecological models, The American Naturalist, 110, 573-594.

Miller, J. H. (1986). A genetic model of adaptive economic behavior. Working paper (University of Michigan).

Mitchell, M. (1996). An introduction to genetic algorithms. MIT Press.（伊庭斉志監訳（1997），『遺伝的アルゴリズムの方法』，東京電機大学出版局。）

Simon, H. A. (1996). The Science of the Artificial. Third Edition. MIT Press.