小山工業高等専門学校研究紀要
第　37　号（2005）179 － 184

カオス時系列の短期予測における、線形予測関数を用いた予測精度について

渡辺　達男

About the Prediction Accuracy in the Time Series of Chaos using the Linear Prediction Function

Tatsuo Watanabe

1. 序論

カオスに関しては現在でも盛んに研究が行われている。カオスの研究は当初は数学的研究が行われていたが、その後は様々な分野に応用され、工学、医学、農学等にも研究範囲は及ぶ。内容も、システムのカオス的振る舞いから、カオス制御、そして最近ではカオス時系列の研究も行われている[1)]。

カオス時系列の問題の1つに、短期予測の問題がある。カオスの中でも決定論的カオスでは、時系列が決定しているのにも関わらず、統計的処理を用いた時系列の予測が難しい。そこで様々な方法で時系列予測の方法が提案されている[2)3)]

前回の論文では、Logistic 写像を例にとり、Lorenz の類推法[3)4)]を用いた時系列予測に関して、近傍点数がどのように予測精度に関係してくるかを調べた[4)]。近傍点数を増やした時、単純に予想精度が増加しないこと等、幾つかの性質が判明した。

今回は、前回の論文では触れていなかった予測関数の問題に関して少々考えてみることにする。

予測関数とは、近傍点に対して、新しい予測点をどのように計算するかを与える関数である。前回の論文では、単純に近傍点の算術平均から予測点を求めた。今回は幾つかの可能性がある関数を提案し、その関数を用いた時の予測精度を議論する。

さらには、埋め込み定理を用いる際の次元の問題も考える。前論文では3次元に限定したが、次元を変化させた時の予測精度の変化を若干計算したので報告する。

2章は予測関数の説明、3章は予測関数を用いた数値実験の結果及び次元を変化させた時の結果、4章は考察に、そして5章はまとめにあてられる。

2. 時系列予測関数

時系列の予測方法に関しては前論文[5)]もしくは合原他[1)3)]を参照してただきたい。ここでは、予測関数について述べる。

最初の可能性は内分点公式を用いることである。

$$x_i(t+1)=\frac{\sum_{j=1}^{l} x_{i,j}(t)m_{i,l-j+1}}{\sum_{j=1}^{l} m_{i,j}} \quad (1)$$

(1)式で、ベクトル $x_i(t+1)$ は予測点ベクトル。$x_{i,j}(t+1)$ は近傍点ベクトルの1ステップ先の j 番目のベクトル、$x_i(t)$ はデータの最後のベクトル。$m_{i,j}$ は、$x_i(t)$ と近傍点 $x_{i,j}(t)$ との距離を表すベクトル、ℓ は近傍点数である。$\ell=2$ ではこの式がよく知られているが、$\ell=3$ 以上では内分点といえない。しかし、簡単のために、単純に拡張した(1)式を用いて予測関数とした。内分点比は近傍点とデータの最後の点との間の距離の比に対応する。

次に(1)式に関して距離の n 乗した関数、

$$x_i(t+1)=\frac{\sum_{j=1}^{l} x_{i,j}(t)m^{n}{}_{i,l-j+1}}{\sum_{j=1}^{l} m^{n}{}_{i,j}} \quad (2)$$

を用いることにした(n＝1、2・・・)。(2)式はより近傍点との距離の影響を強くだす式として考えた。冪数 n を増やすことにより、距離の効果を変化させる。

今回は、上記の線形予測関数に関して予測精度の性質を議論する。

3. 数値実験

2章で説明した予測関数を用いて、実際に Logistic 写像、

$$x(t+1)=ax(t)(1-x(t)) \quad (t=0,1,2\cdots) \quad (3)$$

に関して数値実験で予測を行い、真値と予測値との差

の相対誤差を求めた。実験はPCで行った。

条件として、x(0)=0.5,a=3.8とした。

まず、単純算術平均による予測と、(1)式で近傍点を2点とした、すなわち内分点を用いた予測の比較を行った。

データ数を1000から5000まで変化させた時、算術平均と内分点で1ステップ先まで予測した時の真値との誤差のグラフを図1に示す。ここでデータ数とは、予測をするためのデータベースの数のことである。例えば、データ数1000とは、(3)式でt＝1000個まであらかじめ計算し、そのデータをもとに、t＝1001のxの値を予測する。t＝1000の場合、データ数は1000と表現する。1ステップ先の予測とはt＝1001の値の予測である。

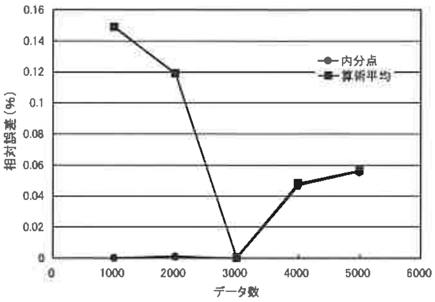

図1　算術平均と内分点(近傍点2点)

図を見ると、データ数が1000、2000までは算術平均に比べて内分点による予測精度が上回っている。しかし、3000以上になると、予測精度は一致している。このことから、データ数が少ない時には、内分点による予測関数は効果があることがわかる。しかし、データ数が3000以上になると予測関数による精度変化がなくなる。

次に、近傍点3点としたときの、内分点の式(1)を用いた予測と算術平均の予測の比較結果を図2に示す。データ数は1000から5000である。

図を見ると、図1と同様にデータ数が1000、2000までは算術平均に比べて内分点による予測精度が上回っている。しかし、近傍点2点による予測よりもその差は小さい。データ数3000以上ではどちらの予測精度にもあまり差がない。このことより、3点以上でもデータ数が少ない時の予測に、内分点の式(1)は有効であることがわかる。しかし近傍点3点の方が2点より誤差は多くなっているようである。

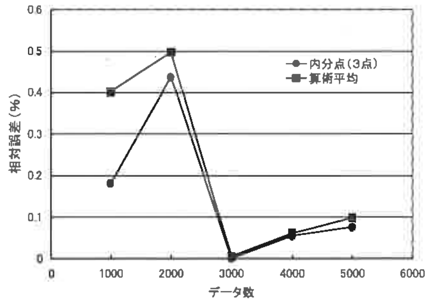

図2　算術平均と内分点(近傍点3点)

そこで、内分点の関数での予測で近傍点2点と3点ではどの程度精度が異なるのか調べてみた。

図3は近傍点2点と3点の内分点を用いた、データ数1000から5000までの1ステップ先の予測の相対誤差の図である。

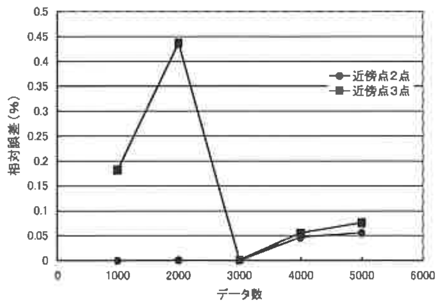

図3　近傍点2点と3点

図を見ると、データ数1000、2000では近傍点2点のほうが精度がよい。これは、データ数が少ないので、近傍点といっても遠くの点を取ってしまうことによるという理由と、3点の内分点の式自体の正当性の問題の2つの理由が重なり合っている可能性がある。いずれにしても、データ数が少ない場合、近傍点2点の方が予測精度がよいように思われる。

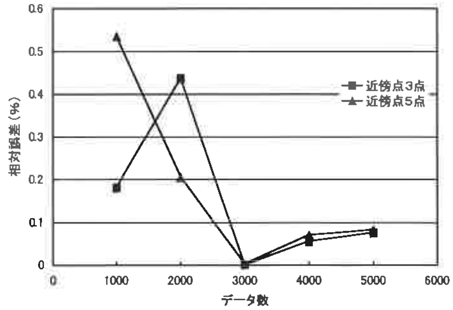

図4　近傍点3点と5点

近傍点をもっと増やしたらどうなるであろうか？

近傍点3点と5点の図を図4に示す。

図を見ると、データ数3000以上では精度は同じようだが、1000, 2000ではどちらがよいか、よくわからない。特に、近傍点5点では誤差がかなり大きく、近傍点を増やすとデータ数が少ないと誤差は増加することが考えられる。

次は内分比の冪乗の関数を考えることにする。まず、(2)式でn=2とした時を考える。

近傍点2点の内分点(n=1)と距離を2乗した内分点(n=2)での精度の比較をする。1ステップ先の相対誤差の結果の図を図5に示す。

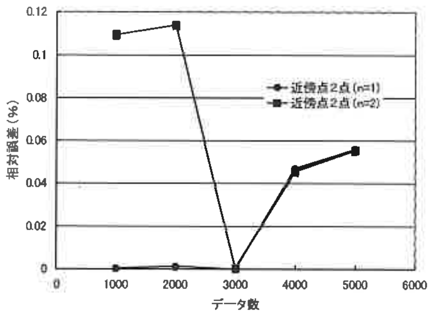

図5 n=1とn=2

図を見ると、データ数が1000, 2000では n=2 の精度が悪い。内分点比を2乗すると精度が落ちる。しかし、データ数が多いと精度は変わらない。

これだけではよくわからないので、もっとnを変化させた時、精度がどうなるか調べてみた。

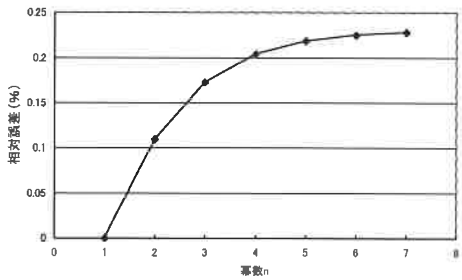

図6 冪数を変化させた時の誤差(2点)

図6はデータ数1000における、近傍点2点での冪数を変化させた時の誤差である。図を見ると、冪数nを増加させると、単純に誤差は増加し、一定値に達するように見える。誤差は冪数 n=1、すなわち内分点の時に一番小さくなる。このことから単純に冪数を増加させても精度はあがらず、逆に悪くなることがわかる。

ところが、近傍点5点にすると、図は変化する。次に示す図7は図6と同じようにデータ数1000における、近傍点5点の冪数を変化させた時の図である。同時に近傍点2点の図を描いてある。

図をみると近傍点5点の場合、誤差が冪数を増加させると減少する傾向がある。

さらには、近傍点2点と5点は冪数が増加すると一定値に収束するように見える。他の近傍点でも同じような傾向が見られた。この原因が数値計算の誤差の可能性かそれとも、他の影響もあるのか、今のところ不明である。しかし一定値に収束するのは興味深い現象である。

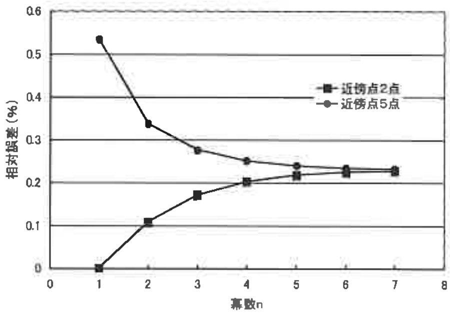

図7 冪数変化(2点と5点)

現象としては興味深いが、実際の予測に対しては誤差が少ない方がよいので、式(2)を用いた場合、近傍点2点で、冪数 n=1、すなわち通常の内分点が一番精度がよいことがわかる。

冪数を一定にしたときの、近傍点数の精度に対する関係を見てみる。図8はデータ数1000の時の、冪 n=1、n=5における、近傍点を変えた時の相対誤差の図である。

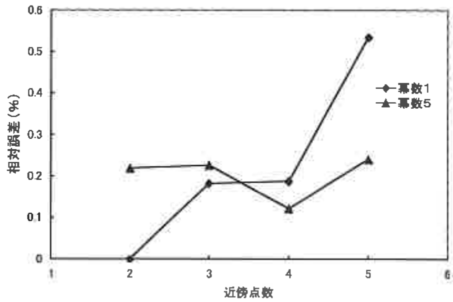

図8 近傍点数と誤差(冪数一定)

図を見ると、冪数 n が1の場合、近傍点数が増加するに従い、誤差も単調に増加している。しかし、冪数 n=5の場合、誤差が一番少ないのは近傍点数4の場合である。冪数 n=2、4の場合でも、近傍点数4の時が誤差が少ない。ただこれはデータ数1000の場合で、データ数が3000以上になると、近傍点2点が誤差が少なくなり、それ以上は誤差は一定の傾向がある。

最後に、埋め込み次元を変化させた時、予測精度が変化するかどうかを調べてみた。

図9は埋め込み次元を変化させた時、データ数1000、3000とした時の予測精度の変化を示す。予測関数は関数の結果への依存性を除外するために、算術平均を用いた。

図を見ると、データ数1000では次元数を増やすと著しく誤差が増加するのがわかる。次元数が9の時は10%程度の誤差がある。

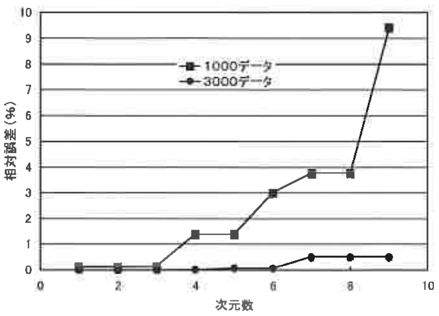

図9　次元数と予測精度

データ数を3000にした時には誤差は少なくなるが、次元数を増やすとやはり誤差は増加する傾向にある。データ数をさらに増やしてもこの傾向は同じようである。データ数が5000では、データ数3000よりも誤差が増えてしまう。このことから、単に次元数を増やしても精度が上がらないことがわかる。注意することは、次元数が1から3までである。この間は精度はあまり変化がない。安定して予測している。データ数を増やしてもこの傾向があまり変わらない。このことから次元数は1から3までがよい。

次元数を増やすとデータ数に対して埋め込み空間内の点の数が少なくなる。従ってより遠くの近傍点を参照して予測しなくてはならないことになる。これにより予測精度が下がると思われる。ただ、十分に多くのデータを用いればこれは改善される。しかし実際の予測において、データ数は少ないので、むやみに次元数を上げる必要は、この結果を見ただけでは必要ないことが結論される。

4. 考察

3章では、数値実験を用い、Logistic 写像の短期予測に関する予測関数の幾つかの例をあげ、実際に予測精度と各パラメータの依存性を調べた。ここでは若干の考察をする。

さまざまなパラメータ変化を行ったが、結局一番成績がよかったのは近傍点2点とした時の、単純な内分点を用いた予測関数によるものであった。

内分点の距離に重みを付けて、冪乗にすると、精度は全般に下がる。データ数が少ない時でもよい成績をおさめたのは近傍点2点の内分点を用いたものである。この理由はよくわからないが、考えられるのは、データ数が少ない時、近傍点数を多く取ると、より遠くの近傍点を取ることになり、遠くの近傍点を用いれば予測精度は明らかに下がる。従って近傍点数2点を用いるとよい。

ただ、近傍点を増やした場合、冪乗にすることで、内分点比（距離の比）を変化させ、少々精度を上げることができる。しかし、その関係は複雑である。

次元を増やした場合、予測精度は下がった。理由は既に述べたが、実際に予測に用いる場合、データ数3000以下では3次元で最も精度がよくなるので、3次元程度で予測するのがよいだろう。

5. まとめ

Logistic 写像を例にとり、Lorentz の類推法を用いた短期予測に関して、予測関数に線形関数の幾つかの例を掲げ、それらの性質に関して若干の数値実験及び考察を行ってきた。

それによれば、最も予測精度がよいのは内分点を用いた予測関数を使うことであった、

また、内分比に冪乗した予測関数を用いると、近傍点数により精度が変化し、ある近傍点で最も精度がよいことがわかった。その近傍点数、冪数はデータ数等に依存することがわかった。

埋め込み次元数を変えた場合の予測精度は、単純に次元数が増加するに従い、単純に悪くなった。ただ、次元が3次元までは比較的一定の低い誤差を示した。

これらをまとめると、次元数は3次元以下で、近傍点数は2点。予測関数は内分点が最も良いことがわかった。

結論はとても簡単になった。パラメータを変化させると、予測精度に変化があるが、どのパラメータも予測精度を良くする方に働いていない。まだ若干のパラメータの考察が残っているが、予測精度の増加の可能性は残されてるだろうか？

この方法ではデータ数が多い場合、パラメータを変化させた時、予測精度が上がる可能性もあるが、実際の自然界の現象の予測を想定した場合、我々が得ることのできる情報はそれほど多く無いので、予測方法自体、他の方法も考える必要があると考える。

多くのデータがある場合、統計的処理との隙間を埋める方法としてこの方法が適用されるかもしれない。

いずれにしても、他の方法と併用するなり、この方法だけに頼ることだけでは精度の向上は難しいかもしれない。

今後は、他の方法と、例えばニューロネットワーク等との併用により予測をする方法を考えたい。

参考文献

1）合原、五百旗：カオス応用システム、朝倉書店（19

95)。
2)松本他:カオスと時系列、培風館(2002)。
3)合原編、池口他:カオス時系列解析の基礎と応用、
産業図書(2000)。
4)E.N.Lorenz:J.Atmos.Sci.26(1969)636.
5)渡辺:小山高専紀要、36号、137(2003)。

「受理年月日　2004 年9月30日」