運針関数と素数生成アルゴリズム：有限不定回数手順モデルに関するノート

仲　澤　幸　壽

1．はじめに

多くの期間分析の経済モデルでは，１つの期間内における経済主体のとる行動を一つに限定している。それは，均衡分析の伝統でもあり，技術的制約のためでもある。しかし，ゲーム論的設定のもとでの分析を進めていくと，１つの期間内に複数回の手順を繰り出す状況を記述することが求められる場合がある。

例えば，「ジャンケン」のような単純なゲームでも，決着がつくまでの経路をシミュレーションしようとすれば，「アイコ」によって不特定回数だけゲームが繰り返される状況を記述できるアルゴリズムが要求される。あるいは，仲澤［８，９］で分析したような経済の競争戦略でも，その分析を発展させようとすると，一期間内に不特定回数に分けて戦略を繰り出す状況の記述が必要になってくることがある。特に，取引タイミングの分布が不確実な状況では不可欠の要素になる。また，不確実な変数の態様を知るために調査的行動を繰り返す状況でも，事前には不特定の回数の調査が繰り返しなされる状態が想定可能である。その例としては，永野［10］が分析した，大学生の就職活動や企業の採用活動が挙げられるであろうし，Gabaix＝Laibson［３，４］の direct cognition という限定合理的意思決定理論の手順も挙げられるであろう。より一般化すれば，不確実性下のサーチ活動は，おおよそ同類の性質を持つといってよいであろう。

そこで，この研究ノートでは，運針関数（stitching function）と称するアルゴリズムを提示し，不定回数だけ行動が繰り返される状況を簡単に記述する方法を示す[1]。そして，その関数が素数生成アルゴリズムと類似のものであることが示される。特に，

行動が停止される条件を記述する関数は，素数カウント関数とよばれるものと同じ発想のものである。

次節では，まず運針関数の提示と，それを用いる例として「ジャンケン」ゲームの記述法が紹介される。３節では，運針関数を導入することによって逐次的に素数列を形成することが可能な定差方程式が存在することが示される。最後に，運針関数の可能性と素数生成式としての限界が議論されるであろう。

２．不定回数手順のモデル化

運針関数とは，次のようなものである。対象となるのは期間分析であり，手順とよばれる戦略または行動が各期において不特定回数とられるものとする。その手順は，運針関数の値が不変になった段階で終了する。すなわち，手順を繰り返しても関数値の変化がゼロであることが自明になると，その期の針の運行は終了するというアルゴリズムである。第 i 期における第 j 回目の手順を $s_{i,j}$ とし，一つ前までの手順

$$q_{i,j-1}=\{s_{i,1},s_{i,2},\cdots,s_{i,j-1}\} \tag{1}$$

を変数とするある関数

$$r_{i,j-1}=F(q_{i,j-1}) \tag{2}$$

があるものとする。ただし，

$$r_{i,j-1}=0 \text{ または } r_{i,j-1}=1 \tag{3}$$

である。すなわち，$q_{i,j-1}$がある条件を満たすかどうかで，関数 $F(\cdot)$ は０か１のいずれかの値しかとらない。なお，$r_{i,0}$の値は，第 $i-1$期の手順終了時の状態から与えられるか，あるいは第 i 期首における初期条件として天下り的に与えられるかのいずれかであるとする。この前提の下で，運針関数は次のように定義される。

定義：手順 $s_{i,j}$ の系列に関して，関数 $T_{i,j}$ が

$$T_{i,j}=r_{i,j-1}\left(s_{i,j}+r_{i,j}T_{i,j+1}\right) \tag{4}$$

と定式化されており，$r_{i,j-1}=1$ であれば手順 $s_{i,j}$ が実施されるが，$r_{i,j-1}=0$ であれば $s_{i,j}$ は実施されずにその期の手順が終了する（$T_{i,j}=0$）アルゴリズムであるとき，関数 $T_{i,j}$ は運針関数（stitching function）であるという。

この定義は，手順を続けるかどうかがそれまでの結果に依存する状況を記述したものである。形としては入れ子式であり，経済学ではいわゆるバロー流の家系的効用関数として知られるもののようにも見えるかもしれない。しかし，家系的効用関数が最大化の目的関数であるのに対して，運針関数は手順を続けるかどうかを記述するだけのものであるという違いがある。

運針関数と名づける理由は，一定の手順をある状態まで続けて休止し，次の段階でまた同じ手順を繰り返すという作業が，裁縫の運針に酷似しているためである。もし，手順が後ろ向きのものも含むのであれば，返し縫運針関数（backstitching function）も定式化可能である。

この関数が有用性を持つことを説明するためには，(2)式と(3)式で示された関数の具体化が必要であろう。そのためには，具体例を用いる方がよい。そこで，次に「ジャンケン」ゲームを例として，運針関数の使用方法を説明することにする。

周知のように，「ジャンケン」ゲームは，３つの戦略（グー，チョキ，パー）を３分の１ずつの割合でだすという混合戦略が均衡解になる例とされる。しかし，現実に「ジャンケン」が行われて，何回か目に決着がつくまでのプロセスの記述がなされているゲーム理論のテキスト等はないようである。その理由は，特殊なアルゴリズムが工夫されない限り，意外に困難だからであろうと推察される。

では，まず最も簡単な２人でジャンケンをする場合から始めよう。運針関数を用いてジャンケンを記述する場合，ゲームに決着がつく状況とアイコで継続する場合との

1) このノートでは，運針「関数」といういい方と「アルゴリズム」という表現とが，混同されているかのような印象を与えるかもしれない。正しくは運針アルゴリズムに統一すべきなのかもしれない。しかし，以下の説明で数式表現そのものを指すときと実際の操作とのときとが区別されることもあるので，あえて不統一にしている。それは，素数生成関数と素数生成アルゴリズムという二つの用語が併用されているのと同じと考えてもらいたい。

区別が必要である[2]。そこで，二人のプレーヤーA，Bのうち，一方が勝ち他方が負けるという決着がついたとき，勝った方の利得を1，負けた方の利得を－1とする。逆にアイコのときには双方とも利得は0であるとする。つまり，各プレーヤーの利得の絶対値が1であればゲームは終了し，0であれば継続するのである。そこで，第i回目のジャンケンゲームの第j手順$s_{i,j}$から得られる利得を$h_{i,j}$とするとき，

$$r_{i,j}=F(h_{i,j})=1-\left|h_{i,j}\right|,\qquad h_{i,0}\equiv 0 \tag{5}$$

とすれば，運針関数はジャンケンの各段階を記述できるアルゴリズムになる。手順$s_{i,j}$は（グー，チョキ，パー）からランダムに選択される手順とみなせるので，運針関数

$$T_{i,j}=\left(1-\left|h_{i,j-1}\right|\right)\left(s_{i,j}+\left(1-\left|h_{i,j}\right|\right)T_{i,j+1}\right) \tag{6}$$

の$T_{i,j}$が，j番目の手順をだす操作か手順をださずに止める操作かのいずれかを意味しているものと解釈できるからである。

これに対して，プレーヤー3人の間の序列を決めるジャンケンゲームになると事態はかなり複雑化する[3]。3人のうち1人だけが負け抜けになって残りの2人が決勝戦を続ける場合等が発生するからである。そこで，2人ジャンケンのケースと同じように，利得によって場合分けを考えよう。プレーヤーA，B，Cの3人が，それぞれ1人の相手に負けたら－1，勝ったら1だけの利得を得るものとし，同じ手どうしのときには双方とも利得は0とする。すると，例えばAがグーでBとCがチョキなら，Aが2の利得を得て終了となるが，BとCは－1ずつの利得で次の順位を決めるジャンケンを続けることになる。BとCの二人のケースは上記の場合と同じで，双方の利得の絶対値が1になったときに終了する。逆に，Aがパーで BとCがチョキなら，Aは－2の利得になって負け抜けとなるが，BとCはさらに続けることになる。3人とも同じ手であれば利得は全員が0で，3人ともゲームを継続する。また，3人の手がすべて異なるときには，おのおのが±1の利得で0になるため，やはり3人とも継続することになる。

2) 決着のつく状況だけがわかればよいのでは，という疑問もあるかもしれない。しかし，以下の3人ジャンケンのときに明らかになるように，アイコの記述は不可欠である。

3) ここでのジャンケンは，3人のなかから1人を選択する場合ではない。3人のプレーヤーにジャンケンの勝ち負けで序列が確定するまでなされる場合を想定している。前者の場合は1人だけの勝ちまたは負けで，他の2人がアイコでもゲームが終了してしまう。それに対して後者では，アイコの2人は決着するまで2人でジャンケンを続ける，という違いがある。

いま述べたことをプレーヤーAを例にして(5)式と同様の継続終了判定関数に定式化するならば，次のようになるであろう。すなわち，第j回目に参加している人数をmとし，Bを相手とする利得をh_{ij}，Cを相手とする利得を$k_{i,j}$とし，［　］をガウス記号として[4]，

$$r_{i,j}=F(h_{i,j})=1-\left[\left|\frac{h_{i,j}+k_{i,j}}{m-1}\right|\right],\qquad h_{i,0},k_{i,0}\equiv 0 \tag{7}$$

である。ただし，該当する相手が既に終了しているときには，その相手に関する利得は変数としては消滅する，すなわち0とする。(7)式が意味するのは，3人の場合では利得の絶対値が2にならなければ終了しないし，2人の場合はそれが1で終了するということである。このケースの運針関数としての表現は，(6)式と同様なので割愛する。

この定式化は，基本的に4人以上のゲームにも拡張可能である。例えば，4人のケースでは，一人が残り3人に対して勝ち抜けになる場合は3の利得，負け抜けになる場合は－3の利得であり，2人が2の利得を獲得し他の2人が－2のときには，2人ずつに分かれて決着をつけることになる。このようにみていけば，(7)式の定式化が4人以上でも有効であることは明らかであろう。

いまのジャンケンのケースをアルゴリズムとしてプログラム化すれば，コンピュータ上で複数のプレーヤーにジャンケンゲームをさせて，実際に決着がつくまでのプロセスを記述させることが可能になる。同様の記述は，一定の状況が成立するまで不特定回数のゲームが繰り返されるような状況下での，経済主体の行動についても可能である。

いま例示したジャンケンゲームでも明らかなように，運針関数で中心的テーマになるのは運針を続けるか終了するかを決める判定関数の定式化である。それは，ゲームの性質から具体化され，継続する際には1，終了する際には0をとるように設定されるものである。これは，数論の世界で素数判定関数あるいは素数カウント関数と呼ばれるものと基本的に同じ構造のものである。そこで，節を改めて，運針関数のアルゴ

4) つまり，［a］は実数aを越えない最大の整数値を意味する。

リズムを用いて素数の数列の生成が可能かどうか検討してみることにしよう。もちろん，素数生成は経済学とは無関係の課題である。しかし，最も法則性を捉えることが困難とされる数列である素数列の記述を研究することは，期間分析の手法の研究としては有益性があるはずである。

3．素数生成アルゴリズム

素数の定義は実に単純である。最も狭義の素数は，自身と 1 意外に約数を持たない自然数のことである。その概念は数千年前から存在するとされるが，いまだに素数公式は発見されていないといわれている[5]。だが，コンピュータの発展もあって，実際には，素数は極めて大きな値まで求められており，実用性のある素数生成のアルゴリズムは相当数存在する。

ここで，リーベンボイム［12］の第 3 章に倣い，素数生成関数は次の 3 条件のなかの 1 つを満たすものとする。すなわち，素数生成関数は，m，n を自然数とし，小さい順に並べた第 n 番目の素数を p_n とするとき，

(a) $f(n)=pn$,

(b) $f(n)$は常に素数であり，$m\neq n$ ならば$f(m)\neq f(n)$,

(c) すべての素数の集合は関数の正の値の集合に等しい,

のいずれかを満たさなければならない，とするのである。

条件(a)が 3 つのなかで最も厳しいものであるのは，明らかであろう。この条件は，例えば，$f(6)=13$，$f(104)=569$というように，第 n 番目の素数が公式に n を代入するだけで得られることを要求する。だが，$f(104)$をいきなり求めることができないにしても，$f(1)=2$ から始まる素数の数列を順次形成するアルゴリズムが定式化されれば，それは(a)の条件に準じるものとみなせるであろう。運針関数を用いることによって，それが可能になるのである。

ここで提示する素数生成式は，次のようなものである。

5) 例えば，芹沢［7］をみよ。また，数論の基礎と素数については，多くの優れた解説書やテキストがあるが，例えば Ribenboim［6］，山本［11］が参考になるであろう。素数そのものに焦点をあてたものとしてはリーベンボイム［12］および Caldwel［2］および後者の原典となっているホームページが参考になるであろう。

運針関数による素数生成アルゴリズム

関数 $F(x)$は x が合成数のとき 1，素数のとき 0 となる素数判定関数として，

$$\begin{aligned} p_n &= 1+p_{n-1} \\ &\quad +F(q_{n,2})\Big(p_0+F(q_{n,3})\Big(p_1+F(q_{n,4})T_{n,2}\Big)\Big) \end{aligned} \tag{8}$$

ただし，ここで初期値は

$$p_0\equiv 1 \tag{9}$$

であり，(8)式右辺の運針関数 $T_{n,j}$ は

$$T_{n,j}=F(q_{n,j+1})\Big(p_1+F(q_{n,j+2})T_{n,j+1}\Big),\qquad j=1,2,3,\cdots \tag{10}$$

であり，$q_{n,j}$ は(8)式右辺の第 1 項から j 項までの和である。

この定式化で，素数判定関数 $F(x)$が機能すれば，

$$p_1=1+p_0+F(1+p_0)\Big(p_0+F(q_{1,3})T_{1,1}\Big)=2$$

$$p_2=1+2+F(3)\Big(p_0+F(q_{2,3})T_{2,1}\Big)=3$$

$$p_3=1+3+F(4)\Big(1+F(5)T_{3,1}\Big)=5$$

$$\vdots$$

$$p_{104}=1+563+F(104)\Big(1+F(565)\Big(2+F(567)\Big(2+F(569)T_{104,5}\Big)\Big)\Big)=569$$

$$\vdots$$

というように，素数の数列が順次形成されていく。(8)式の定式化では，素数の数列が 2，3 と始まるということも前提にされていないことにも留意すべきである。だが，それでも n 段階の手順は n 番目の素数に到達するまで和をとり，到達した時点で終

了するというアルゴリズムになっている。よって，素数の数列が順次形成されていくことが自明なので，⑻式が素数生成関数になっていることの証明は不要であろう。

このアルゴリズムが実用的かどうかは，素数判定関数の数値計算が容易であるかどうか，コンピュータで計算する際にオーバーフローを生じさせないかどうかに依存する。なぜなら，素数判定関数として利用されるものは，Wilson の定理を用いるために計算の桁数が極めて急速に増大するからである。Wilson の定理とは，

$$p \text{ が素数} \quad \leftrightarrow \quad (p-1)!\equiv -1 \pmod{p}$$

というものである。つまり，ある自然数が素数であれば，それから 1 を引いた和の階乗に 1 を加えた数は元の素数で必ず割り切れるし，そうなるものは素数しかないという内容である。この定理は古典的なもので，Fermat の小定理から比較的容易に導くことができる[6]。そこで，

$$F(x)=1-\left[\cos^2\left(\frac{(x-1)!+1}{x}\pi\right)\right] \qquad ⑾$$

とおけば，$F(x)$ は x が素数のとき 0，合成数のとき 1 となる。なぜなら，ラジアン表示の角度が π の整数倍ならコサイン関数の絶対値は 1 だが，整数ではない有理数倍ならばそれが 1 未満だからである。よって，⑻式の運針関数に必要な素数判定式の条件を満たす。しかし，容易に想像できるように，コサイン関数内の階乗の計算量は，x の増大とともに急激に増大し，パソコン上で数学ソフトを利用して計算しようとしても，いずれはオーバーフローが発生してしまう[7]。

これに対して，ごく最近，理論上はオーバーフローが生じ難い AKS 素数判定法と呼ばれるアルゴリズムが Agrawal＝Kawal＝Saxena［1］によって提示された。しかし，そのアルゴリズムは多段階の手続きからなる複雑なものであり，実際に用いるまでには相当に改善される必要があるようである。

そこで，⑾式と同程度に単純な構造で，なおかつ計算量が急激には増大しない素数判定法が要求されることになる。それがなければ，⑻式の運針関数による素数生成関数は意味のないものとなってしまう。しかし，現在までのところ，決定的な素数判定方法は見出されていないといわざるをえない。よって，このノートでは，暫定的な方法を提案するにとどめざるを得ない。

まず，素数を $p_4=7$ までの段階では，コンピュータや電卓を利用しなくても⑾式は十分に筆算で求められることを確認しておこう。すなわち，

$$\frac{(7-1)!+1}{7}=\frac{6\cdot5\cdot4\cdot3\cdot2\cdot1+1}{7}=\frac{721}{7}=103$$

である。言い換えれば，この段階までは素数判定関数は⑾式のままでよいということになる。では，それより大きな数の場合はどうすべきだろうか。ここでは，素数のそもそもの定義に遡って，初歩的な判定方法を定式化することにしよう。初歩的な判定方法とは，素因数分解ができないということをもって素数と判定することである。つまり，判定する対象の数より小さな素数で割り切れるものがあれば判定関数は 1 となり，割り切れなければ素数となって判定関数が 0 をとるという定式化である。しかし，ここで注意しなければならないのは，判定対象の数より小さい素数のすべてを試さなくてもよいということである。なぜなら，$p_1=2$，$p_2=3$ より 7 を超える素数は偶数ではなく 3 の倍数でもないからである。そうであれば，対象となる数のせいぜい 3 分の 1 未満までの素数を試せば十分ということになる。以上から，$i\geq 5$ に関して，

$$F(q_{i,j})=\prod_{k=2}^{m}\left(1-\left[\cos^2\left(\frac{q_{i,j}}{p_k}\pi\right)\right]\right), \qquad p_m=\max\left\{p_t \mid p_t\leq\left[\frac{q_{i,j}}{3}\right]\right\} \qquad ⑿$$

という素数判定関数が設定可能であることになる。⑿式は，⑻式で p_0 または p_1 を加えて和をつくるたびに素因数分解を試みて，1 つでも素因数があれば判定関数が 0 になり，素因数が 1 個もなければ判定関数が 1 となって次の手順，すなわち次の和をとる作業に進むというアルゴリズムを提供する。これは，最も初歩的な素数判定方法を数式として記述したにすぎないものともいえる。

⑿の素数判定関数では，計算そのものは多段階だが，計算する桁数は判定対象の数を超えることがない。これは，オーバーフローをできるだけ生じさせないという意味では優れている。しかし，判定対象の数が比較的小さなときには，⑾式の方がコンピュータの計算速度は早いようである。

6) 証明は，例えばリーベンボイム［12］の pp.16 - 17をみよ。
7) ⑾式の形の素数判定関数を用いた素数生成式には Willans の定式化があるが，その定式化では素数判定式以外にも計算量が膨大になる部分があり，通常のパソコンでは20番目の素数を算出するだけでも膨大な時間量を要してしまい，実用性はないと評価されている。やはり，リーベンボイム［12］（3章）をみよ。

以上で，運針関数が素数生成関数を構成可能であることが明らかになったことになる。そこで不定回数手順のコントロールに中心的役割を果たす素数判定関数は，リーベンボイム［12］の挙げた３つの条件のうち，条件(c)とも関係がある。

リーベンボイム[12]によれば，条件(c)は見かけ上の表現よりも限定的な意味を持つものと解釈されている。それは，素数の集合がディオファンタス的集合(diophantine set）と呼ばれる集合に属することから，素数を生成する特殊な多項式を導出する試みを指している。その多項式のとる正の値は素数の集合を構成するが，同じ値を繰り返しとるという性質もあり，また次数が極めて高いという難点を持っている。しかし，数論学者の一部は，この手法に多くの期待を寄せているようである。

だが，素数判定関数との関係で条件(c)に言及するのは，そのような高度な次元の話ではない。例えば，自然数 n の関数

$$y = nF(n), \quad n = 1,2,3,\cdots \tag{13}$$

を考えてみよう。この関数が正の値をとるのは，明らかに n が素数のときだけである。それ以外の合成数のときは，y は０である。確かに，条件(c)の要請に合致している。しかし，素数を生み出す効率は悪い。なぜなら，すべての自然数列のなかから素数以外を０にしたものなので，０の方が素数よりもはるかに多いのである。ちなみに，100項までのケースを掲げてみよう。

0	2	3	0	5	0	7	0	0	0
11	0	13	0	0	0	17	0	19	0
0	0	23	0	0	0	0	0	29	0
31	0	0	0	0	0	37	0	0	0
41	0	43	0	0	0	47	0	0	0
0	0	53	0	0	0	0	0	59	0
61	0	0	0	0	0	67	0	0	0
71	0	73	0	0	0	0	0	79	0
0	0	83	0	0	0	0	0	89	0
0	0	0	0	0	0	97	0	0	0

素数の割合は25％である。周知のように，値が大きくなるほど素数の占める割合は低下するので，素数はさらにまばらになっていく。

この点は，比較的簡単な工夫によって少しは改善することができる。前にも触れたが，３より大きな素数は３の倍数ではない。すなわち，５以上の素数は

$$y' = kF(k), \quad k = 3n \pm 1 \tag{14}$$

という数列に含まれる。これは，自然数の数列より項がおおよそ３分の１少ないので，素数の割合はそれだけ増加する。だが，もう少し改善することも可能である。それは，５以上の素数が奇数であることを利用する。もし(14)式の n が奇数なら，k は偶数になってしまう。よって，n が偶数のケースに限定できることになる。つまり，５以上の素数は，

$$y'' = k'F(k'), \quad k' = 6n \pm 1 \tag{15}$$

に含まれることになる[8]。もちろん，$6n \pm 1$ の双方が素数のときは，いわゆる双子素数を形成する。このケースで100項までを掲げると以下のようになる。

5	7	11	13	17	19	23	0	29	31
0	37	41	43	47	0	53	0	59	61
0	67	71	73	0	79	83	0	89	0
0	97	101	103	107	109	113	0	0	0
0	127	131	0	137	139	0	0	149	151
0	157	0	163	167	0	173	0	179	181
0	0	191	193	197	199	0	0	0	211
0	0	0	223	227	229	0	0	239	241
0	0	251	0	257	0	263	0	269	271
0	277	281	283	0	0	293	0	0	0

初項が５から始まるため，前のケースとは単純には比較できないが，今回は素数の比

率が59％まで上昇している。脚注８でも述べているように，特定の n を排除すれば，さらに素数比率を高めることができる。だが，それにも限度があり，素数のみの集合に絞り込むことは困難である。結局，このような方法としては，条件(c)の要請する，正の値がすべて素数という条件を満たすというだけで満足しなければならないようである。

４．運針関数の可能性と限界

この研究ノートは，１期間の手順のとられる回数が事前には不確定な状況を記述するための運針関数を提示し，その可能性を探ることを目的としていた。そのために，ジャンケンゲームと素数生成関数のアルゴリズムを検討した。しかし，ジャンケンゲームの場合はある程度の成果を得たといえるであろうが，素数生成関数に関しては，大きな課題が残されているといえよう。なぜなら，ここで提示したアルゴリズムが素数生成の手続きを記述しているにしても，数論の専門家が素数公式と認めることは期待できないからである。その理由は，アルゴリズムが簡単でオーバーフローも生じさせ難いものであっても，素数の性質そのものを明らかにする情報の提供をあまり期待できないからである。素数に関して，よく知られている未解決問題である双子素数の無限性やゴールドバッハの予想に関しての貢献すら，期待は薄いであろう。ましてや，リーマン予測に関しての貢献の可能性は極めて小さいといわざるをえない。つまり，ほとんどの問題は課題として残されてしまうのである。その点を改善するためには，おそらく素数の性質そのものに関して，未知の部分が明らかにされるというような画期的発展があって，素数判定式が大幅に改善される必要があるように思われる。それは，もちろん経済学とは無縁の分野での発展かもしれない。しかし，そのことで運針関数のアルゴリズムが改善されれば，不定回数手順ゲームの分析力も大きく進歩するのではないだろうか。

8) ⑮式のとき，素数判定関数を用いずにある程度素数を絞り込むことができる。⑮式で n の1の位が4か9だと，$6n+1$ は5の倍数になってしまう。また，n の1の位が1か6のときには，$6n-1$ は，やはり5の倍数になる。よって，5以外のケースは排除可能である。しかし，これだけでは，例えば77等の合成数を排除できない。やはり，判定関数が必要なのである。

参　考　文　献

[1] Agrawal, M., N. Kayal and N. Saxena, (2002) ‘PRIMES in in P,’ mimeo, http : //www.cse.iitk.ac.in./news/primality_v3.pdf.
[2] Caldwell, C. K., (2004) *The Prime Pages － Prime Number Research, Records, and Resources*, (http : //primes.utm.edu/), (SOJIN 編訳『素数大百科』共立出版。)
[3] Gabaix, X. and D. Laibson, (2004) ‘Bounded Rationality and Direct Cognition,’ mimeo.
[4] Gabaix, X. and D. Laibson, (2004) ‘Information Acquisition : Experimental Analysis of a Boundedly Rational Model,’ mimeo.
[5] Ribenboim, P., (1995) *The New Book of Prime Number Records*, New York, Springer-Verlag.
[6] Ribenboim, P., (2000) *My numbers, My Friends － Popular Lectures on Number Theory*, New York, Springer-Verlag。
[7] 芹沢正三（2002）『素数入門』講談社。
[8] 仲澤幸壽（2004）「経営者心理と販売戦略：過剰需要期待分析序論」西南学院大学経済学論集，39-1，145-192。
[9] 仲澤幸壽（2005）「経営上の意思決定における心理と景気変動」西南学院大学経済学論集，39-3，179-232。
[10] 永野　仁編著（2004）『大学生の就職と採用』中央経済社。
[11] 山本芳彦（2003）『数論入門』岩波書店。
[12] リーベンボイム（吾郷孝視訳）（2001）『素数の世界－その探検と発見　第2版』共立出版。