金沢大学 計算機センター 広報

Vol. 4, No. 3, 1975

目　　次

DATA PROCESSING CENTER
KANAZAWA UNIVERSITY

——研　　究——

パターン認識と学習　(3)

工学部・基礎工　上　坂　吉　則

これまで2回にわたってつぎのような見出しで，パターン認識と学習についてお話ししてきた：

1．情報科学とパターン認識，
2．パターン認識の語義，
3．情報通信機械と認識機械，
4．人間にとって易しい認識・機械にとって易しい認識，
5．認識の原理は解明ずみ！？，
6．分類技術としての認識，

いましばらく，6番目の話題をつづけることにしよう。

6．分類技術としての認識（つづき）

前回ではコーネル大学の心理学者 Rosenblatt の提唱した〝パーセプトロン〟と呼ばれる図形認識機械を紹介し，そのエッセンスを数学的に簡単に表現したのであった。つまり，nケのメッシュからできた図形パターンを一般に

$$X=(x_1,\cdots,x_n)$$

のようにn次元ベクトルで表わす。このベクトルの各成分x_iは網膜上のi番目の視細胞に図形刺激があれば値1をとり，そうでなければ値0をとる。もっと平易にいえば，たとえば図形パターンの黒い部分を〝1〟，白い部分を〝0〟（あるいはその逆でもよい）で表わすということである。このとき，パーセプトロンというのは要するにつぎのような特別な形をした，パターンXの関数fのことをいうのであった：

$$f(X)=\begin{cases}1, & g(X)=\sum_{j=1}^{n} w_j x_j > \theta \text{のとき;}\\ 0, & g(X)=\sum_{j=1}^{n} w_j x_j \leqq \theta \text{のとき.}\end{cases}$$

この式で$w_1,\cdots,w_n$は重み係数，θはしきい値と呼ばれる実数の定数である。たとえばXが文字〝A〟を表わす図形パターンであることをこのパーセプトロンに認識させるには，Xが〝A〟を表わすパターンのときにのみ，$f(X)=1$となり，そうでないときには$f(X)=0$となるように，$w_1,\cdots,w_n$とθをうまく選んでおけばよいというわけである。

以前に，たとえば文字〝A〟を表わす図形パターンXの集合のことを〝パターン・クラス〟と呼んでいた。この用語を用いると，パーセプトロンfは$f(X)=1$となるXの集合であるパターン・クラスと$f(X)=0$となるXの集合であるパターン・クラスの2つを考え，任意に与えられたパターンXがいずれのクラスに属しているかを認識する機械ということになる。記憶の良い読者であれば，この認識の方法は前に自動診断の1例としてお話しした胃ガンの識別方法と似ていることに気付かれるであろう。[(5)]パーセプトロンfはそこで用いられた線形識別関数と全く同じ形をしている。したがって，Rosenblatt の考えたパーセプトロンというのは結局のところ線形識別関数の計算をする機械に他ならないということになるが，それが脳における神経回路のモデルから出発したものの一つの帰結であるわけだから，われわれの脳の中には無意識的に線形識別関数的思考をする部分があっても不思議でないことを示すものといえる。実際，日常経験の中である事柄を決めるのにそれにまつわるいくつかの要因に重みをつけて考えることがよくある。例えばある研究所で今年度海外留学させる所員を決定する状況を思い

浮かべて見よう。留学の各候補者について，語学力，研究能力，研究業績，これまでの海外研究者との交流の度合，年令，在職年数，現在の研究の進行状況等が調べ上げられる。決定委員会のメンバーはこれらの要因に無意識にあるいははっきりと重みw_jを定め，候補者に総合点の順位（$g(X)$の大きさの順位に相当する）をつけることによって，議論を進めていくということになるかも知れない。こうした場合，各委員の思考活動はおおむねパーセプトロン的といえる。実際，このような思考過程をより明確に定式化し，決定過程を合理化しようとする研究が統計学の立場からも行われている[6]。

ところで，上に示したパーセプトロンfでは，文字〝A〟を表わすものとそうでないものの2つのパターン・クラスに対する認識はできるが，たとえばアルファベット26文字のどれであるかというように，パターン・クラスの数が3以上になった場合には用いることができない。この場合には図13に示すような多クラス型のパーセプトロンが考えられている。つまり，これ

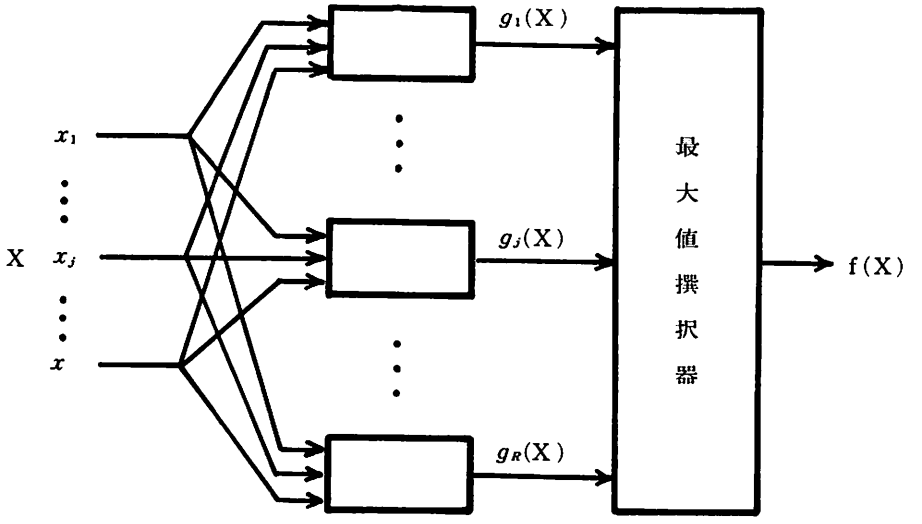

図13　多クラスを認識するパーセプトロン

までのパーセプトロンを構成するのに用いられた関数gをパターン・クラスの数Rと同じだけ，g_1，…，g_Rと用意する，そして入力パターンXに対するこれらの値$g_1(X)$，…$g_R(X)$を最大値撰択器に入れて，最大値をとっている関数の添字をf(X)の値として出力する。もしf(X)＝kならばパターンXはk番目のパターン・クラスに属することを意味する。識別関数fを正確にかけば，

$$f(X)=k,\ \text{ただし}\ j=1,\ \cdots,\ R\ \text{に対して}\quad g_j(X)\leqq g_k(X)\ \text{のとき}.$$

どうしてこのような構成を考えるのか？それは脳の生理学的知見によるというよりはむしろ統計学的根処に由る。というのは，パターンやその集合に適当に確率構造を入れて考えたとき，$g_j(X)$を〝パターンXが与えられたとき，そのXがj番目のパターン・クラスに属する〟条件付確率を表わすと解釈する．実際，ある仮定のもとでこの条件付確率はXの成分x_1，…，x_nの線形結合として$g_j(X)=w_{j1}x_1+\cdots+w_{jn}x_n$ と表わすことができるという数学的事実がよく知られている。この様な場合には$g_j(X)$が最大となっているようなjにパターンが属すると決定したとき，誤りの確率が最小になることが証明できるのである。いわゆるBayesの原理がこれである。このような統計学的背景を考えたとき，図13に示す構成は誤認識の少ない機械であることが十分期待できるわけである。

7．機械に学習させるには？

認識機械の一つの例としてパーセプトロンについてお話ししたが，その際，重み係数w_j やしきい値θ をどのように決めたらよいかという点には触れなかった。もしw_jやθ を勝手な値にとってパーセプトロンを働かせれば，まったくデタラメな認識をすることになろう。つまり，ちっとも認識能力がないわけである。これはちょうど生れたばかりの人間が全然文字を読めないのにある意味で似ている。人間の場合には，成長の過程で回りの人達から，いろいろな文字図形について教えられ，その結果文字を認識できる能力を身につけていく。このとき普通，文字を〝学習〟した，あるいは〝習得〟したといっている。この学習という所作は，文字だけに限らず，さまざまなものの認識能力の習得には重要な働きを持っていることは，心理学の本をひもとくまでもなく，日常体験から容易に認められるところである。

パーセプトロンにもこうした〝学習〟をさせることによって，当初はデタラメな認識をしていても次第に正しい認識をするようになるであろうか？もし，そうだとしたら，このパーセプトロンを〝学習する機械〟と呼んでもおかしくはないであろう。このようにして，さまざまな認識機械に学習できる能力をいかにして付与するかという研究，つまり学習機械の研究がパターン認識の分野でも活発に行なわれるようになり，認識と学習は密接な関係を持つようになったのである。（因みに，ある学会には「パターン認識と学習」研究会と称する専門研究会が設けられ，毎月の例会で活発な研究発表が行なわれている）。

ここでは，せっかくパーセプトロンを取り上げてきたのだから，それが実際に学習できる様子をごく簡単な例でご紹介しよう。いま，つぎのようなパーセプトロンを考える：

$$f(X)=\begin{cases}1, & g(X)=w_1x+w_2x_2+w_3x_3+w_4x_4\geqq 0\ \text{のとき};\\ 0, & g(X)=w_1x_1+w_2x_2+w_3x_3+w_4x_4<0\ \text{のとき}.\end{cases}$$

つまり，パターンとしては4つの枡目（x_1，x_2，x_3，x_4）を持ったものを考え，簡単のためしき値θ は零としておく。各x_jは0または1をとるから，パターンの総数は$2^4=16$個。この内の

	パターン X				希望の認識結果
X ＝(x_1	x_2	x_3	x_4)	f(X)
X_1＝(0	1	1	1)	0
X_2＝(0	0	1	1)	0
X_3＝(1	1	1	1)	0
X_4＝(0	1	0	1)	0
$\widetilde{X}_1$＝(0	0	0	1)	1
$\widetilde{X}_2$＝(1	0	0	1)	1
$\widetilde{X}_3$＝(1	0	1	1)	1
$\widetilde{X}_4$＝(1	1	0	1)	1

表2　パターンX_1～X_4に対しては認識結果は0，
パターンX_1～X_4に対しては認識結果は1
となるようにパーセプトロンを学習させたい。

8個のパターンに対して表2のように分類してほしいとしよう。残りのパターンに対してはどのような認識をしてもよいものとする。これらのパターンをパーセプトロンfに繰り返し〝見せて〟，いま見せたパターンは〝0〟のクラスであるか〝1〟のクラスであるかを教える。この学習の過程を通して，X_1～X_4，$\widetilde{X}_1$～$\widetilde{X}_4$のすべてのパターンが正しく認識されるように，パーセプトロン自身が重み係数w_jを変更していかなければならない。この変更の方法の一つはつ

ぎのようなものである:

（1°） パターンを見せてパーセプトロンがそれを正しく認識した場合には，重み係数は変更しない。

（2°） クラス〝0〞のパターンX＝（x_1，…，x_4）を誤ってクラス〝1〞のパターンと認識したときは，現在の重み係数w_1，…，w_4をw_1-x_1，…，w_4-x_4に変更する。

（3°） クラス〝1〞のパターンX＝（x_1，…，x_4）を誤ってクラス〝0〞のパターンと認識したときは，現在の重み係数w_1，…，w_4をw_1+x_1，…，w_4+x_4に変更する。

この変更方法は人間の学習に例えれば，いわゆるドリル（問題集）を子供にやらせて，正しい解答をだしたら頭をなでてやり（あるいはアメを与え），間違った答をしたときには注意する（あるいはムチを与える）といった「アメとムチの教育」によく似ている。昨今流行のプログラム学習も（ムチなどという恐しいものは持ちださないが，その代り解けなかった問題よりもっと易しい問題を沢山解かされる）この種の学習方式の変形である。技能的なことがらの習得には大変効果が上るそうであるが，強い批判もないわけではない。これは心理学における対照的な2つの学派，連合主義と認知主義の思想的基盤の相違によるものであるがこれについては後にふれるとして，さきを急ぐこととしよう。

表3　パーセプトロンの学習の様子

学習回数	提示パターン X＝(x_1, x_2, x_3, x_4)	重み係数 w_1	w_2	w_3	w_4	$g(X)$	$f(X)$	重みの変更	新しい重み係数
1	X_1＝(0 1 1 1)	0	0	0	0	0	1	する	0 −1 −1 −1
2	X_2＝(0 0 1 1)	0	−1	−1	−1	−2	0	しない	
3	X_3＝(1 1 1 1)	0	−1	−1	−1	−3	0	しない	
4	X_4＝(0 1 0 1)	0	−1	−1	−1	−2	0	しない	
5	$\tilde{X}_1$＝(0 0 0 1)	0	−1	−1	−1	−1	0	する	0 −1 −1 0
6	$\tilde{X}_2$＝(1 0 0 1)	0	−1	−1	0	0	1	しない	
7	$\tilde{X}_3$＝(1 0 1 1)	0	−1	−1	0	−1	0	する	1 −1 0 1
8	$\tilde{X}_4$＝(1 1 0 1)	1	−1	0	1	1	1	しない	
9	X_1＝(0 1 1 1)	1	−1	0	1	0	1	する	1 −2 −1 0
10	X_2＝(0 0 1 1)	1	−2	−1	0	−1	0	しない	
11	X_3＝(1 1 1 1)	1	−2	−1	0	−2	0	しない	
12	X_4＝(0 1 0 1)	1	−2	−1	0	−2	0	しない	
13	$\tilde{X}_1$＝(0 0 0 1)	1	−2	−1	0	0	1	しない	
14	$\tilde{X}_2$＝(1 0 0 1)	1	−2	−1	0	1	1	しない	
15	$\tilde{X}_3$＝(1 0 1 1)	1	−2	−1	0	0	1	しない	
16	$\tilde{X}_4$＝(1 1 0 1)	1	−2	−1	0	−1	0	する	2 −1 −1 1
17	X_1＝(0 1 1 1)	2	−1	−1	1	−1	0	しない	
18	X_2＝(0 0 1 1)	2	−1	−1	1	0	1	する	2 −1 −2 0
19	X_3＝(1 1 1 1)	2	−1	−2	0	−1	0	しない	
20	X_4＝(0 1 0 1)	2	−1	−2	0	−1	0	しない	
21	$\tilde{X}_1$＝(0 0 0 1)	2	−1	−2	0	0	1	しない	
22	$\tilde{X}_2$＝(1 0 0 1)	2	−1	−2	0	2	1	しない	
23	$\tilde{X}_3$＝(1 0 1 1)	2	−1	−2	0	0	1	しない	
24	$\tilde{X}_4$＝(1 1 0 1)	2	−1	−2	0	1	1	しない	
25	X_1＝(0 1 1 1)	2	−1	−2	0	−3	0	しない	
26	X_2＝(0 0 1 1)	2	−1	−2	0	−2	0	しない	〈学習完了〉

さて上の手順によってパーセプトロンに実際に学習させて見る。最初は重み係数はすべて零としておく：$w_1=w_2=w_3=w_4=0$（実はw_iは最初は何であってもよい）。ここで$X_1=$（0，1，1，1）を提示すると，$g(X)$は

$$g(X)=w_1x_1+\cdots+w_4x_4$$

であったから，$g(X_1)=0$。したがって$f(X_1)=1$となる。X_1はクラス〝0〟のパターンであった（表2）から，変更規則（2°）によって重み係数を$w_1=0-0=0$，$w_2=0-1=-1$，$w_3=0-1=-1$，$w_4=0-1=-1$に変更する（表3参照）．つぎに$X_2=$（0，0，1，1）を提示すると$g(X_2)=0\times0+(-1)\times0+(-1)\times1+(-1)\times1=-2$．故に$f(X_2)=0$となり，$X_2$は正しく認識された。したがって変更規則（1°）によって重み係数は変えない。以下，パターンX_3，X_4，$\widetilde{X}_1$，…と繰り返し学習をすすめていくと，このパーセプトロンの重み係数は表3に示すように変化していく。この表で学習回数が19から26までの間にはX_1，…，X_4，$\widetilde{X}_1$…，$\widetilde{X}_4$のすべてのパターンが含まれており，これらのパターンに対して重みを変更していないということは，認識が正しかったことを示している。つまり，実質的には18回の学習によって希望のすべてのパターンを正しく認識できるような重み係数を習得したわけである。

実際のパターン認識ではパターンの成分x_iの個数は数百にもなる場合があるので，重み係数の修正を手計算で実行するのはもちろん大変であるが，電子計算機を用いれば何の造作もないことであろう。学習の問題で計算機によるシミュレーション実験がよく採用されるのはこのためである。

ところで，今の例では18回のパターン提示で学習が完了したが，いつもこのようにうまく学習が収集するのであろうか？実はRosenbattがパーセプトロンを提唱したとき，すでにこのことを問題にし，ある条件の下ではつねに有限回のパターン提示で学習が収束することを数学的に証明している。今日，パーセプトロンの収束定理としてこの分野では広く知れ渡っている重要な成果の一つである。

8．歴史を辿って——パターン認識の困難——

パターン認識——特に図形や音声の認識——について実に多くの研究がなされているにも拘わらず，人間の認識能力に匹敵するような機械の開発には遠く及ばないということを前にお話しした。それは一体何故なのだろうか？この疑問の一端を認識にまつわる思想にふれながら，歴史をややさかのぼる所から考え始めることにしよう。

近代哲学の始祖といわれるデカルトの著作の一つに「省察——神の存在，および人間の精神と身体との区別が証明されるところの第一哲学についての省察——」[7]がある。この中の一節につぎのようなくだりを見いだすことができる。

「物体ですら，本来は，感覚あるいは想像の能力によって把握されるのではなく，ただ“悟性”によってのみ把握されるのだということ，また，触れたり見たりすることによって把握されるのではなく，もっぱら“理解”することによって把握されるのだということが，いまや私に知られたのである…」

これは〝もの〟の認識についてデカルトがどのように考えていたかを示す一文として興味深い。それ以前（1619年11月10日）にいわゆる“コギト（Cogito ergo sum—— 私は考える，ゆえに私は存在する）”を発見し，それによって上に引用した結果を導いたといわれる。デカルトは精神と肉体を明確に区別して考え，肉体は機械的なものであって物理学（当時は自然学）の諸法則で説明できるが，精神は決して機械論的には把握されるようなものではなく，生得的に備わっている“観念ideas”によってはじめて理解できると考えた。この精神と肉体の二元性を導入したことが，デカルト以降のほとんどの学問，特に哲学やそれの分身としての心理学に強く影響して今日に至っている。具体的にはデカルトの精神と肉体のいずれに重点を置くかによって，哲学では

合理論　と　経験論

の対立がひき起され，また心理学では

認知主義　と　連合主義

の２大学派を対照的に区別することができる。

こうした思想の相違は工学の枠組の中での認識や学習の問題にも時には微妙に，また時には重大に関連してくる。この両者の立場の相違をやや詳しく見ることによってパターン認識の困難さの理由へと接近していくことにしよう。

前に「パターン認識の語義」の項で，広辞苑によれば，認識とは「知る作用および成果の両者をさす。知識と同義語」であることを述べた。そして学習というのは，パーセプトロンの例で見たように，重み係数についての〝知識〟を得ること，一般には認識のためのアルゴリズムの知識を得ることに他ならなかった。したがって，学習とは〝知識の習得〟であるといってよいであろう。実際，哲学の一つの重要部門ある認識論ではこの〝知識の習得〟に関する問題をめぐって，さきの合理論と経験論が火花を散らしている。つまり知識の習得がどのようになされるのかというテーマに対して，デカルトの二元論における肉体に関する部分は付随的なものと考えることによって〝合理論的アプローチ〟が生じ，また精神に関する部分を破棄することによって〝経験論的アプローチ〟が生れる。デカルトは知識の習得は肉体によってではなく，精神によってなされると考えたから，合理論的アプローチはデカルトの見解をほゞ受けついでいるといってよい。実際，後にふれるが，言語の学習に関して合理論的立場をとるチョムスキーは自からの言語理論を〝デカルト派言語学〟と呼んでいるくらいである。[8]

イギリスの経験哲学の大御所の一人ロックによればわれわれの高次の観念は感覚作用と内的感覚（自分自身の精神的操作の知覚，つまり反省のこと）の２つの源泉から生ずるという。われわれは観念によって思考するのであり，すべての観念は経験に由来するから，どのような知識も経験に先行することはあり得ないと考える。したがって知識の習得，つまり学習や認識の能力は本来生得的にわれわれに備っているのではなくて，外界からの刺激パターンによって〝条件づけ〟や〝連合・汎化・分化〟などの作用を通して作り上げられるものだということになる。

これに対して合理論的アプローチでは感覚器官などの抹しょう的な情報処理機構はもちろんのことさまざまな仕組み――とりわけ，哲学者が観念と呼ぶもの――が生得的に備っていて，外界からの刺激パターンはこれらによって高度に組織化されてしまうと考える。したがって，すべての真理は新しく見い出されるのではなく，すでに本性的にわれわれの内に存在しているのだという，デカルトやライプニッツの見解に従うことになる。たとえば言語の学習を例にとって見ると，チョムスキーはフンボルトの考えを辿りながらつぎのように述べている：

「言語というのは，本当は，教えることのできないものであり，ただ言語が，それなりに，精神の中で自発的に発展してゆくための条件を与えてやるということができるだけである。であるから，言語の特定の形式つまりその言語の文法のシューマは大部分与えられているものである。もっとも言語を形成していく過程が有効なものとなるような適切な経験がなければそれ（文法のシューマ）は利用できるようにならないであろう。ライプニッツと同様，彼（フンボルト）は，個体に関して言うなら，習得というのは，大部分，〝再生Wiederzeugung)〟，すなわち，精神の中に生得的にあるものを引きだすという仕事である，というプラトン主義的見解を再説している」[9]

要するに経験論と合理論の相違は学習の本質が外界での〝経験〟にあると考えるか，それとも経験は不可欠ではあるがあくまで付随的なことがらであって大部分の知識はすでに〝生得〟されていると考えるかの違いである。この対照的な見解は，歴史的には当然ながら心理学へも受けつがれ，連合主義（経験論に由来する）と認知主義（合理論に由来する）の対照が見られる。

心理学では学習とは〝生活体に，練習または経験が与えられたとき，それによって生活体の行動に生起する比較的永続的な変化〟を指すのが普通である。したがって疲労や成熟による行

動の変化は学習とは見なされない。そこで経験論の立場からこの学習を考えると外界から生活体に入ってくる刺激パターンが学習つまり行動の変化をもたらす本質的な原因とするわけであるから，刺激と行動の間の内的なつながり，つまり刺激を行動に連合associateさせる仕組が学習の最も大切な面であるということになる。これが連合主義の基本的な見解である。連合主義の学習理論には歴史的にはパブロフの条件反射に基礎をおくものから，スキナーのプログラム学習の理論までさまざまなものがあるが，さきに取り上げた学習機械パーセプトロンも思想的にはこの連合主義の流れに沿うものとして位置づけることができる。

一方，合理論に従って学習の問題を把えると，〝経験〟としての刺激パターンが直接の原因となって学習結果としての行動の変容を規定すると考えるのではなくて，生得的なメカニズムの中で経験が〝組織化〟されることによって行動に変化が現われてくるということになる。生得構造による経験の組織化を〝認知〟と呼び，それは刺激パターンが形造る世界の構造を認知する作用に他ならないとするのが認知主義の学習観である。このことをもう少し具体的なイメージで把握するために，言語の学習は認知主義でなければ説明がつかないとするチョムスキーの見解を引用しておこう[10]：

「言語を習得する幼児は，文と文でないものを観察することによって（すなわち，自分をとりまく言語社会での修正によって）自力によって文法を作り上げていく。文を文でないものと区別し，文のあいまいさ ambiguities を検出するなどの，実際に観察された話し手の能力を研究して見ると，明らかにつぎの結論をとらざるを得ない。すなわち，話し手の持っている文法はとてつもなく複雑でしかも抽象的な性格のものであるということ，そして形式的観点からは注目すべきタイプの理論を構築するという所作を幼児が成切裡に成し遂げているということ。さらにこの所作はすべての幼児が同じように達成できるのである。どのような学習理論もこの事実をうまくとり扱わなければならない。…すべての正常な幼児が非常に複雑でしかも本質的には同じ文法をきわめて早期に習得しているという事実はつぎのことを示唆している。つまり人間には未知の性質や複雑なものについて〝仮説を定式化する〟能力が備わっいて，上述の所作を実行できるように，特にそのために設計されていると。」

認知主義の立場に立つ学習理論もいくつかあるが，その代表の一つにゲシュタルト学派がある。ゲシュタルト心理学が強い説得力を持つにいたった理由の一つは図形の知覚において生得的構造が著じるしく関与している現象を多く提供したことであろう。その一例を紹介して認知主義の理解の助けとしよう。

たとえば図14の図形を見れば，誰が見ても〝２つの円の重なった図形〟と認知するであろう。それ以外の見え方，たとえば図15のようにフランスパンのような閉曲線の中に凸レンズが入っ

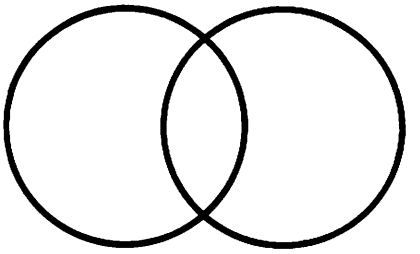

図14　この図形はどのように見えるだろうか？

ているような図形（図ａ）や２つの三ヶ月が左右に向き合っている図形（図のｂ）のように見ることは素直でないと思われるであろう。論理的にはその他無数の可能性があるはずである。にもかかわらず，図14を〝２つの円が重なったもの〟としか見ることができないのはどういう訳なのだろうか？ゲシュタルト学派にいわせれば，無限の可能性の中から特定のものを選択するこの仕組こそ生得的構造なのだというわけである。

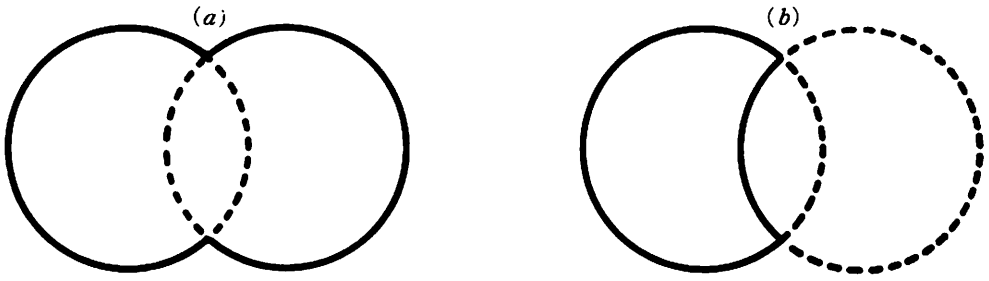

図15　2つの円の別の見え方

これに対して反論がでるかも知れない。たとえば，“2つの円が重ったもの”と見えるのは何も生得的構造によるのではなくて，生れて以来誰れでも円を三ヶ月や凸レンズの形よりも多く見て〝経験〟しているからに他ならないからだ——これが経験主義からの反論である。それではとゲシュタルト学派は再反論する：どんな図形の中にでもよく見なれた図形があればそれを〝見る〟ことが本当にできるのだろうか？たとえば，図16の図形から何を見ることができよう。実はこの図形の中によく見なれているはずの数字の〝4〟が埋没しているが，それを見てとる人はまずいないであろう。

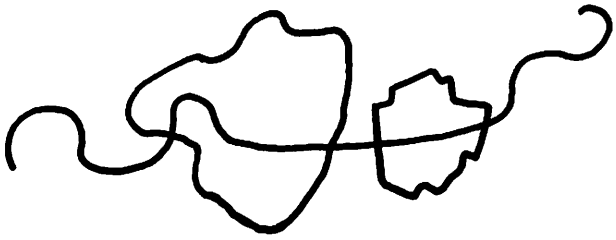

図16　この中に何を見ることができるか？

こうしたたぐいの議論が延々とつづき，合理論と経験論は今日に至っている。そのいずれが真理なのか？それはあるいは永遠に解決されることのない謎かも知れない。しかしゲシュタルト心理学が提供してくれた視覚現象の特異性はまぎれもない事実である（興味ある読者は文献（11），（12）に多くの例を見いだすことができよう）。人間が図14をいくら先入観を取り去り虚心に戻って眺めて見ても，2つの円の重なりという一種の〝偏見〟を持って見てしまう。コンピューターにこの図形を見せたらどうであろうか？この図形を構成している点のx, y一座標の数値の集合があるのみである。その集合には〝2つの円の重なり〟はおろか何の構造も入っていない。これがコンピューターの偏見，つまり〝偏見なしで見る〟という偏見である。

人間と機械のこの著るしい相違，ここにパターン認識の最大の困難があるわけである。もし，人間がこの〝偏見〟という生得的構造を持ち合わせていなかったら，世界とは人間にとって，ちょうど〝イチマツ模様〟を眺めたときのように，とりとめもなくギラギラと揺れ動く不安な様相を呈するに違いない。そこにはしかと確信を持って認めることのできる〝カタチ〟など見えよう筈もないであろう。生得的構造なぞ仕組まれてはいない今日のコンピューターは世界をまさにこうした様相のものとして受けとっているに違いない。とすれば，そのコンピューターにパターン（〝カタチ〟あるもの）を認めさせようとするのは所詮無理な話だということになる。

合理論者の主張する生得的構造が人間の中に実在するかどうかは別としてもそれが果している役割をコンピューターに付与することなしには人間に匹敵する能力を持った認識機械を創り得ないであろうとするのが，筆者の見解である。それ故に，ゲシュタルト心理学の提供してくれるさまざまな視覚現象あるいは合理論者がその存在を主張する生得的構造の正体を解明して

いくことが認識研究の最重要な課題となる。しかし，これはまた多くの哲学者や心理学者が永年にわたって追いつづけてきた永遠のテーマでもある。おいそれと片づく代物ではない。漸新な方法論を内包した情報科学がこの問題にどれだけ深くメスを入れ得るか，それをこれからの時代に期待したいものである。

9．あとがきにかえて

このお話は，１９７３年の秋に計算機センターと電気関係学会の共催で行なわれた講演会での「パターン認識と学習機械」を再現するようにという係からの指示で書きはじめたものもある。しかし，月日が経過していくうちに，忠実な再現が困難になり，講演会当時のものとはやや異なったものになった（であろう）ことをここでお詫びしておきたい。

パターン認識と学習という大きな題目に対して，ごくわずかの内容しか盛りこめなかったのが心残りであるが，この分野の雰囲気や問題の困難さ，あるいは心理・生理・数学・哲学等との学際的な協力研究の心要性などが多少なりともご理解いただけるならば筆者の望外の幸せである。

終りにこの分野を一瞥するのに手頃な展望論文を紹介して筆をおくことにする。

（A） G.Nagy : State of the art in pattern recognition, Proc. of IEEE, **56**, 5, 836－862 (1968).

（B） M.D.Levine : Feature extraction (A Suvuey), Proc. of IEEE, **57**, 8, 1391－1407 (1969).

（C） Yu-Chi Ho and A, K, Agrawala: On pattern classification algorithms (Introduction and Survey), Proc. of IEEE, **56**, 12, 2101－2114 (1968).

また，最近の新しい手法を見るには

（D） K.S.Fu : Syntactic methods in pattern recognition, Academic Press (1974).

（E） R. O. Duda and P. E. Hart : Pattern classification and scene analysis, Wiley (1973).

（完）

参　考　文　献

(5) 上坂吉則：パターン認識と学習(２)，金沢大学計算機センター広報，**4**，１，２－８ (1974).

(6) 林知己夫：数量化の方法，東洋経済新報 (1974).

(7) 野田又夫編：世界の名著22，デカルト，中央公論社 (1972) のpp. 223－307.

(8) N. Chomslcy : Cartesian linguistics : A chapter in the history of rationalist thought, Harper & Row (1966) (川本訳，テック社，1970).

(9) N. Chomsky : Aspects of the theory of syntax, the MIT Press (1965). (安井訳，研究社，1970).

(10) N. Chomsky : Review of Skinner's "Verbal behavior", in J. A. Forder and J. J. Katz(eds): The structure of language, Prentice-Hall(1964).

(11) メッガー：視覚の法則，岩波書店 (1968).

(12) グリゴリー：インテリジュントアイ，みすず書房 (1972).

測定値から未知パラメーターを決定する方法(III)

——多変数直接探索法——

理学部・化学　須原正彦・村田重男

これまでに，我々が採用している非線形最適化法の概要とその中の一変数探索法のプログラムを紹介した[1,2]が，今回は多変数の場合の直接探索法（３種類）について具体的に記す。

(1) Hooke-Jeenesの方法

この方法は，パターン決定とパターン移動から成るくりかえしのパターン探索である。アルゴリズムは図１に示す。具体的なプログラムをシート１に示す。但し関数サブプログラムは全て略した。

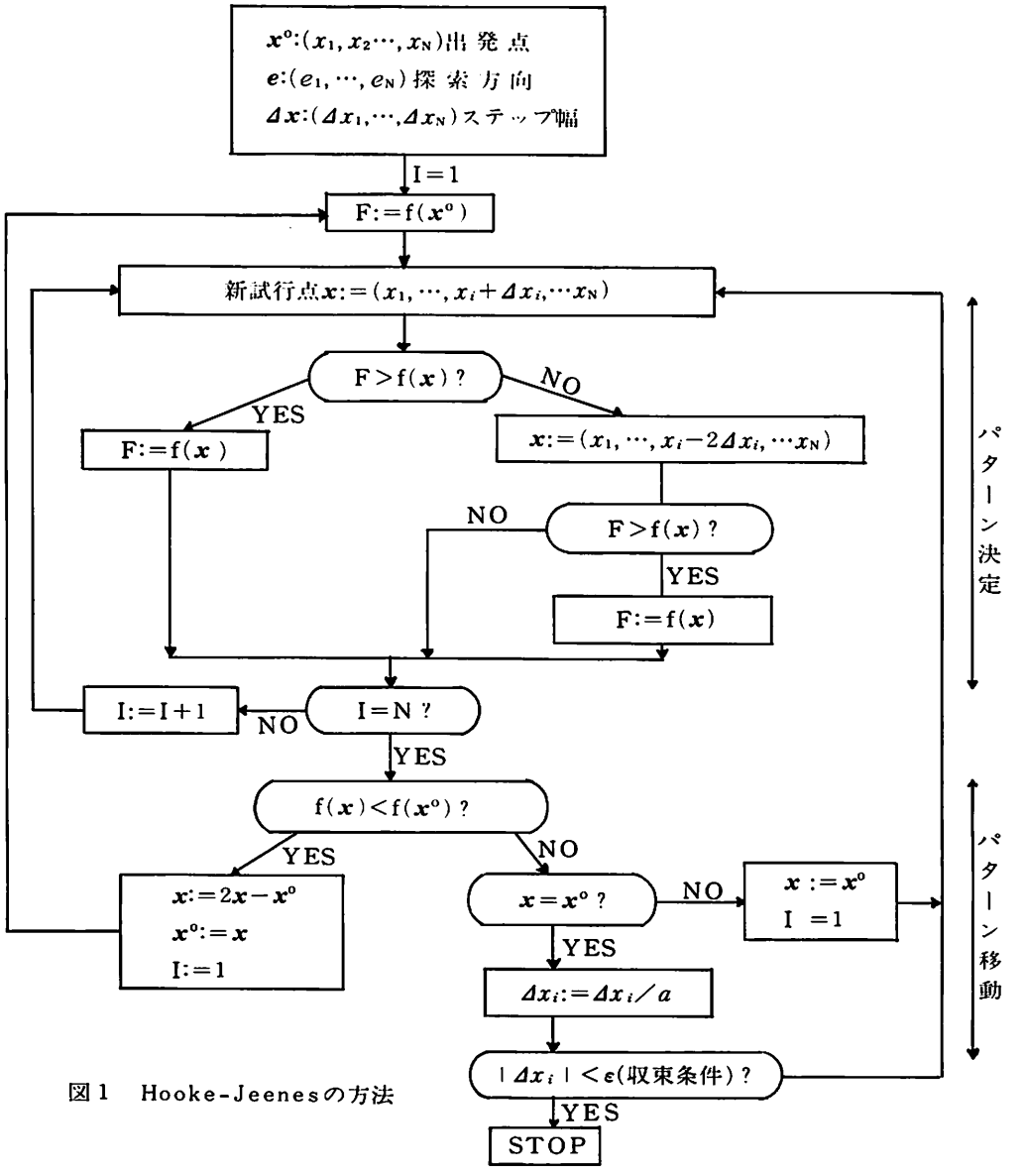

図１　Hooke-Jeenesの方法

〔プログラム名〕

ＨＪＤＳＭ：Hooke-Jeenes 法の主プログラム．初期値の読み込みと印字．結果の印字．

ＨＪＭＡＩＮ：ＨＪＳＵＢ１，ＨＪＳＵＢ２，ＨＪＳＵＢ３を用いた探索の制御．

ＨＪＳＵＢ１：基点の回りでのパターン決定

ＨＪＳＵＢ２：パターン移動

ＨＪＳＵＢ３：ステップ幅の縮小

〔入力データ〕

ＭＡ：　Ｉ３　パラメーターの群の数（普通は１）

ＭＢ：　Ｉ３　くりかえしの数（普通は１）

ＭＣ：　Ｉ３　全パラメーターの数

ＭＸ(Ｉ)，Ｉ＝１，ＭＡ：10Ｉ５　i 番目のパラメーター群中のパラメーターの数（即ち実際に動かすパラメーターの数）

(ＸＰＡＲＡ(Ｊ，Ｉ)，Ｊ＝１，ＭＣ)，Ｉ＝１，ＭＡ：　６Ｆ13.6　i 番目の群中の j 番目のパラメーターの初期値

(ＳＤ(Ｊ，Ｉ)，Ｊ＝１，ＭＣ)，Ｉ＝１，ＭＡ：　６Ｆ13.6　i 番目の群の j 番目のパラメーターについてのステップ幅の初期値

〔出　　力〕

途中経過の様子を知るため，ＨＪＳＵＢ１，ＨＪＳＵＢ２，ＨＪＳＵＢ３を通過するごとにＭＸ個のパラメーターの値を印字する。ＨＪＤＳＭでは最終的に求まった値を印字する。

(2)　Rosenbrockの方法

これは，Hooke-Jeenes の方法におけるパターン決定にあたる探索の各サイクル毎に，互に直交する方向の組を用いる点が特徴である。但し全ての方向の内，１つでも試行が成功するかぎり探索をくりかえす。アルゴリズムは図２に示す。プログラムは省略する。

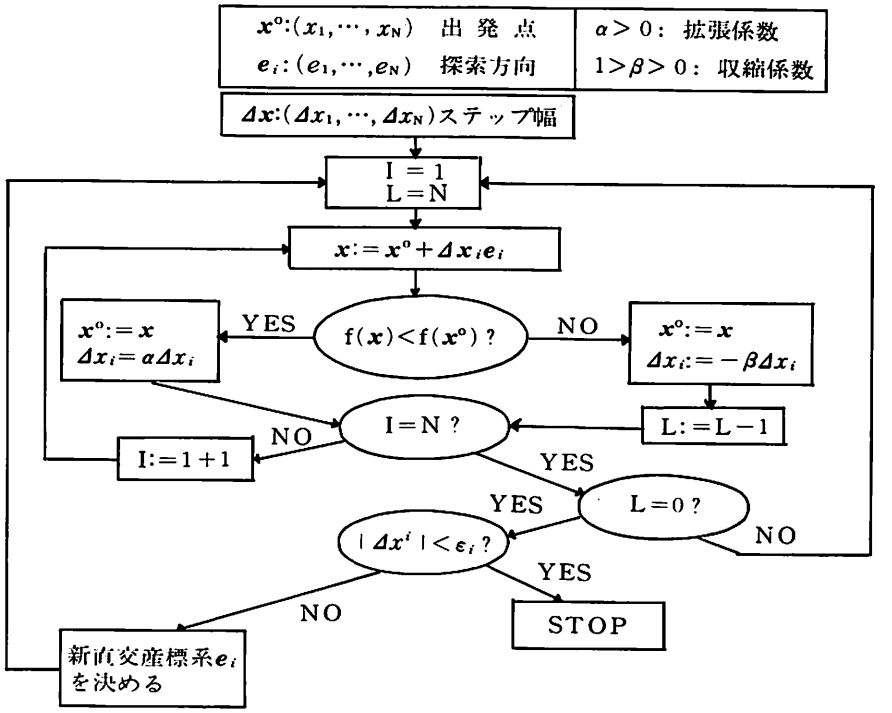

図２　Rosenbrockの方法[3)]

(3) シンプレックス法

この方法は，最小点をかこむ空間を縮めることにより最小点を探索するものである。従って，一方向探索を用いていない点，および関数が２次近似で良い近似であるという事を仮定していない点から，不連続な導関数をもつ関数や，最小点において２次でないような関数にも適用できる。収束が速く，浅い局所的極小点や関数の誤差に対して鈍い。アルゴリズムは図３に示す。[3] プログラムはシート２に示す。但し関数サブプログラムは全て略した。

〔プログラム名〕

ＭＮＳＬＸ：　主プログラム．パラメーターの初期値．コントロール用定数の読み込みと印字．

ＳＬＸＳＵＢ：シンプレックスの各頂点の初期値の読み込み．シンプレックス法による探索．結果の印字．

〔入力データ〕　ＭＮＳＬＸ用

ＡＸ(Ｉ)，Ｉ＝１，10：６Ｅ13.6　パラメーターの初期値

Ｍ　Ｘ　Ｘ：Ｉ５　パラメーターの数

ＡＬＰＨＡ：Ｆ15.5　鏡像係数＞０

ＡＢＥＴＡ：Ｆ15.5　１＞収縮係数＞０

ＡＧＡＭＭＡ：Ｆ15.5　拡張係数＜１

ＡＥＲＲ：Ｆ15.5　停止条件パラメーター

ＳＬＸＳＵＢ用

(Ｘ(Ｉ，Ｊ)，Ｊ＝１，ＭＸＸ)，Ｉ＝１，ＭＸＸＸ：６Ｅ13.6　シンプレックスの各頂点の初期値。但し　ＭＸＸＸ＝ＭＸＸ＋１

〔出　力〕

ＳＬＸＳＵＢにおいて　Ｘ(Ｉ，Ｊ)：シンプレックスの各頂点の初期値の印字，ＺＦ(Ｉ)：各頂点での関数値，ＩＰＡ：プロセスの番号，Ｘ(Ｉ，Ｊ)：新らしいシンプレックスの各頂点の値　を印字する。収束すればＡＡＢ：平均二乗誤差を印字する。

以上多変数直接探索法について述べたが，Kowalik と Osborneの成書[3] に引きつづき，最近「非線形最適化」に関するすぐれた解説書が次々と出版され，今後，この種の問題について，プログラミングをなさる方に非常な助けとなると思われますので，参照文献4)，5)をあげておきました。

参　照　文　献

1）須原正彦，村田重男，金沢大学計算機センター　広報 3（１），10（1973）.

2）須原正彦，村田重男，金沢大学計算機センター　広報 3（２），16（1973）.

3）J. Kowalik and M. R. Osborne,「非線形最適化問題」（1968）山本善之，小山健夫共訳（培風館）.

4）M. J. Box, D. Davies and W. H. Swann,「非線形最適化の技法」（1969）黒田充訳（培風館）.

5）L. C. W. Dixon,「非線形最適化計算法」（1972）村原正一訳（培風館）.

プログラムシート　1

```
      SUBROUTINE HJDSM(F)
      IMPLICIT DOUBLE PRECISION(F,E,X,S,C,A)
      INTEGER TIMECH,TIMEIN
      DIMENSION MX(8),XPARA(5,10),SD(5,10),XS(20),XX(5)
      COMMON /BIPA/MAX,JO, MXXX,IM,IL
      COMMON /BCONST/CQ,CO,CN,CL
      COMMON /BXD/XPARA,SD,MX,AAAA(716)
      COMMON /BFUNC/MMOD,MVAR,AC(20),AGAMMA
      EXTERNAL EB,EA,EC
  500 FORMAT(3I3)
  502 FORMAT(10I5)
  504 FORMAT(6F13.6)
  600 FORMAT(1H0////' HOOKE-JEEVES METHOD STARTING DATA'////)
  602 FORMAT(1H0,T3,'MOTOMERU PARAMETER NO KUMI NO KAZU'/1H0,I12/1H0,T10
     1,'KURIKAESHI NO KAZU'/1H0,T10,I13/1H0,T20,'KUMI NO PARAMETER NO KA
     2ZU'/1H0,T20,10I10)
  604 FORMAT('0INITIAL VALUE OF PARAMETER'//(1H0,5X,6E20.6))
  606 FORMAT('0INITIAL VALUE OF SPACE'//(1H0,5X,6E20.6))
  608 FORMAT(1H0,50X,'MOTOMETA PARAMETER NO ATAI --',I3,'/',I3//(1H0,6E13.
     16))
  610 FORMAT(1H0,T50,'** HJDSM WA ERR NI YORI END **')
  612 FORMAT(1H0,'MMOD= ',I3,5X,'MVAR = ',I3)
  111 FORMAT(' TIME=',I10,'(SEC)')
      WRITE(6,600,ERR=800,END=800)
  800 READ(5,500,ERR=700,END=810)MA,MB,MC
  810 READ(5,502,ERR=700,END=820)(MX(I),I=1,MA)
  820 WRITE(6,602,ERR=700,END=830)MA,MB,(MX(I),I=1,MA)
  830 READ(5,504,ERR=700,END=840)((XPARA(J,I),J=1,MC),I=1,MA)
  840 WRITE(6,604,ERR=700,END=850)((XPARA(J,I),J=1,MC),I=1,MA)
  850 READ(5,504,ERR=700,END=860)((SD(J,I),J=1,MC),I=1,MA)
  860 WRITE(6,606,ERR=700,END=870)((SD(J,I),J=1,MC),I=1,MA)
  870 READ(5,500,ERR=700,END=880)MMOD,MVAR,MMMM
  880 WRITE(6,612,ERR=700,END=890)MMOD,MVAR
  890 A=FUNC(XS,XX)
      DO 1 IO=1,MB
      DO 1 JO=1,MA
      MXX=MX(JO)
      CALL HJMAIN(EC)
      WRITE(6,608,ERR=880,END=1)IO,JO,((XPARA(J,I),J=1,MC),I=1,MA)
    1 CONTINUE
      GO TO 910
  700 WRITE(6,610,ERR=900,END=900)
  900 CQ=0.0
  910 RETURN
      END
```

```
      SUBROUTINE HJMAIN(E)
      IMPLICIT DOUBLE PRECISION(X,S,E,A)
      DIMENSION XPARA(5,10),SD(5,10),XZ(5),SZ(5),EE(10)
      COMMON /BXD/XPARA,SD,MX(8),AAA(716)
      COMMON /BXDS/XZ,SZ,XXZ(5),XXPZ(5)
      COMMON /BIPA/MAX,JO,MXXX,IM,IL
      COMMON /BRPA/LPA
      EXTERNAL EB,EA,EC
      DATA EE(1),EE(2)/0.1D2,0.5D-2/
      KDEBUG=1
      TRACE(KDEBUG)LPA,SZ
      SUBTRACE(KDEBUG)HJSUB1,HJSUB2,HJSUB3
```

```
      DO 1 I=1,MXX
      XZ(I)=XPARA(I,JO)
    1 SZ(I)=SD(I,JO)
      ALPHA=1.5D0
      AB=E(XZ)
1000  CALL HJSUB1(EC)
      AC=E(XXZ)
      IF(AB-AC)100,100,200
 200  AB=E(XXZ)
      DO 3 I=1,MXX
    3 IF(XZ(I).NE.XXZ(I)) SZ(I)=SZ(I)*ALPHA
      GO TO 1000
 100  LPA=1
 700  LP=MOD(LPA,3)+1
      GO TO(730,710,720),LP
 710  CALL HJSUB1(EC)
      GO TO 700
 720  CALL HJSUB2(EC)
      GO TO 700
 730  CALL HJSUB3
      IF(SZ(1)-0.1D0)740,740,700
 740  DO 2 I=1,MXX
    2 XPARA(I,JO)=XZ(I)
      A=E(XZ)
      RETURN
      END
```

```
      SUBROUTINE HJSUB3
      IMPLICIT DOUBLE PRECISION(X,S)
      DIMENSION XZ(5),SZ(5)
      COMMON /BIPA/MAX,JO,MXXX,IM,IL
      COMMON /BXDS/XZ,SZ,XXZ(5),XXPZ(5)
      COMMON /BRPA/LPA
      DO 1 I=1,MXX
    1 SZ(I)=SZ(I)*0.1
  700 LPA=LPA+1
      WRITE(6,600)(SZ(I),I=1,MXX)
  600 FORMAT(1H0,4HSUB3,5E15.7)
      RETURN
      END
```

```
      SUBROUTINE HJSUB1(E)
      IMPLICIT DOUBLE PRECISION(E,X,S,A)
      DIMENSION XZ(5),SZ(5),XXZ(5),YZ(5)
      COMMON /BXDS/XZ,SZ,XXZ,XXPZ(5)
      COMMON /BRPA/LPA
      COMMON /BIPA/MAX,JO,MXXX,IM,IL
      KDEBUG=1
      TRACE(KDEBUG)A'XXZ
      A=E(XZ)
      AB=A
      DO 1 I=1,MXX
      YZ(I)=XZ(I)
    1 XXZ(I)=XZ(I)
      DO 2 I=1,MXX
      XXZ(I)=XXZ(I)+SZ(I)
      AA=E(XXZ)
      IF(AA-A)700,710,710
  700 A=AA
      GO TO 3
  710 XXZ(I)=XXZ(I)-4.0*SZ(I)
      AA=E(XXZ)
      IF(AA-A)700,730,730
  730 XXZ(I)=YZ(I)
    3 CONTINUE
    2 CONTINUE
```

```
      WRITE(6,600)(XXZ(I),I=1,MXX)
  600 FORMAT(1H0,4HSUB1,5E15.7)
      AC=E(XXZ)
  740 IF(AB-AC)760,760,750
  760 LPA=LPA+1
  750 LPA=LPA+1
      RETURN
      END
      SUBROUTINE HJSUB2(E)
      IMPLICIT DOUBLE PRECISION(E,X,S)
      DIMENSION XZ(5),XXZ(5),SZ(5),XPZ(5),XXPZ(5)
      COMMON /BXDS/XZ,SZ,XXZ,XXPZ
      COMMON /BIPA/MXX,JO,MXXX,IM,IL
      COMMON /BRPA/LPA
      EXTERNAL EB,EA,EC
      DO 1 I=1,MXX
      XPZ(I)=XZ(I)
      XXPZ(I)=XXZ(I)
    1 XZ(I)=2.0*XXZ(I)-XZ(I)
      CALL HJSUB1(EC)
      LPA=2
      IF(E(XXZ)-E(XXPZ))700,710,710
  700 DO 2 I=1,MXX
    2 XZ(I)=XXZ(I)
      GO TO 100
  710 DO 3 I=1,MXX
    3 XZ(I)=XXPZ(I)
  100 LPA=LPA+2
      WRITE(6,600)(XZ(I),I=1,MXX)
  600 FORMAT(1H0,4HSUB2,5E15.7)
      RETURN
      END
```

プログラムシート　2

```
      SUBROUTINE MNSLX
      IMPLICIT DOUBLE PRECISION(F,E,C,A,V)
      DIMENSION AX(20),AS(5)
      COMMON /BFUNC/MMOD,MVAR,V(20)
      COMMON /BCONST/CW,CO,CN,CL
      COMMON /BIPA/MXX,JO,MAXY,IM,IL
      COMMON /BRPA/MXXA,ALPHA,ABETA,AGAMMA,AERR
      EXTERNAL FS,ES
 1000 FORMAT(6E13.6)
 2000 FORMAT(1H0,10E13.5)
  502 FORMAT(2I3)
  504 FORMAT(1H0,'MMOD=',I3,'MVAR=',I3)
  500 FORMAT(I5,4F10.5)
  600 FORMAT(1H0////T5,'SIMPLEX METHOD PARAMETER'////'DIMENSU NO KAZU =',I
     15//'ALPHA =',E13.6,T30,'BETA =',T36,E13.6,T60,'GAMMA =',T67,E13.6//'
     20ETSMI JOKEN PARAMETER EPSILON =',E15.6)
  602 FORMAT(1H0,T30,'** SIMPLEX METHOD WA ERR NI YORI END **')
      READ(5,1000)(AX(I),I=1,10)
      WRITE(6,2000)(AX(I),I=1,10)
      DO 1 I=1,10
    1 V(I)=AX(I)
      AAA=FUNC(AX,AS)
      READ(5,500,ERR=700,END=800)MXX,ALPHA,ABETA,AGAMMA,AERR
  800 WRITE(6,600,ERR=700,END=810)MXX,ALPHA,ABETA,AGAMMA,AERR
  810 MXXX=MXX+1
      READ(5,502,ERR=700,END=830)MMOD,MVAR
  830 WRITE(6,604,ERR=840,END=840)MMOD,MVAR
  840 CONTINUE
      CALL SLXSUB(FS,ES)
      IF(CW.EQ.0.0)GO TO 700
      GO TO 820
  700 WRITE(6,602,ERR=820,END=820)
  820 RETURN
      END
```

```
      SUBROUTINE SLXSUB(F,E)
      IMPLICIT DOUBLE PRECISION(F,E,C,A,X,Z)
      INTEGER TIMECH,TIMEIN
      DIMENSION X(21,20),ZF(21),XO(20),XR(20),XE(20),XC(20)
      COMMON /BIPA/MXX,JO,MXXY,IM,IE
      COMMON /BCONST/CW,CO,CN,CL
      COMMON /BRPA/MXXA,ALPHA,AHETA,AGAMMA,AERR
      COMMON /BXD/A,AXCA(400)
  500 FORMAT(6E13.6)
  600 FORMAT('0SIMPLEX INITIAL VALUE'//(1H ,6E20.6))
  602 FORMAT('0SIMPLEX METHOD NO KEKKA (IPA='I3,')'//(1H ,6E20.6))
  604 FORMAT('0SIMPLEX METHOD END --DS@RT(B)='E20.6)
  111 FORMAT(' TIME='I10,'(SEC)')
      KOEBUG=1
      READ(5,500,ERR=700,END=800)((A(I,J),J=1,MXX),I=1,MXXA)
  800 WRITE(6,600,ERR=700,END=810)((X(I,J),J=1,MXX),I=1,MXXA)
  810 CALL CLOCKM(TIMEIN)
      DO 1001 IPA=1,200
      WRITE(6,610)(ZF(I),I=1,MXXA)
  610 FORMAT(10E13.6)
      CALL CLOCKM(TIMECH)
      TIMECH=TIMECH-TIMEIN
      WRITE(6,111)TIMECH
      DO 1 I=1,MXXX
    1 ZF(I)=F(I)
      DO 2 I=1,MXX
      II=I+1
      DO 2 J=II,MXXA
      IF(ZF(I)-ZF(J))200,200,900
  900 AZ=ZF(I)
      ZF(I)=ZF(J)
      ZF(J)=AZ
      DO 3 K=1,MXX
      AA=X(I,K)
      X(I,K)=X(J,K)
    3 X(J,K)=AA
  200 CONTINUE
    2 CONTINUE
      AAXX=MXXX
      AAX=MAX
      DO 4 I=1,MXX
      XO(I)=0.0
      DO 4 J=1,MXX
    4 XO(I)=XO(I)+X(J,I)/AAX
      AA=0.0
      DO 5 I=1,MXXX
    5 AA=AA+ZF(I)/AAXX
      AB=0.0
      DO 6 I=1,MXXX
    6 AB=AB+(ZF(I)-AA)**2/AAXX
      WRITE(6,602,ERR=700,END=820)IPA,((X(I,J),J=1,MXX),I=1,MXXA)
  820 IF(DSQRT(AB)-AERR)2000,910,910
```

```
      940 DO 7 I=1,MXX
        7 XR(I)=(1.0+ALPHA)*XO(I)-ALPHA*X(MXXX,I)
          AA=E(XR)
          IF(AA-ZF(MXX))920,920,930
      920 IF(AA-ZF(1))990,990,940
      990 DO 8 I=1,MXX
        8 XE(I)=AGAMMA*XR(I)+(1.0-AGAMMA)*XO(I)
          AB=E(XE)
          IF(AB-AA)950,950,940
      940 DO 9 I=1,MXX
        9 X(MXXX,I)=XR(I)
          ZF(MXXX)=AA
          GO TO 1000
      950 DO 10 I=1,MXX
       10 X(MXXX,I)=XE(I)
          ZF(MXXX)=AB
          GO TO 1000
      930 IF(AA-ZF(MXXX))940,940,960
      960 DO 11 I=1,MXX
       11 XC(I)=ABETA*X(MXXX,I)+(1.0-ABETA)*XO(I)
          AA=E(XC)
          IF(AA-ZF(MXXX))970,980,980
      970 DO 12 I=1,MXX
       12 X(MXXX,I)=XC(I)
          ZF(MXXX)=AA
          GO TO 1000
      980 DO 13 I=2,MXXX
          DO 14 J=1,MXX
       14 X(I,J)=(X(I,J)+X(1,J))/2.0
       13 ZF(I)=F(I)
     1000 CONTINUE
     1001 CONTINUE
     2000 AAB=DSQRT(AB)
          WRITE(6,604,ERR=630,END=630)AAB
          GO TO 630
      700 CW=0.0
      630 RETURN
          END
```

——解　　説——

FŌRTRANによるエラーについて

計算機センター　　車　古　正　樹

昨年の11月より，FŌRTRANの異常終了（文法エラー・結合エラー・実行エラー・時間オーバ・用紙オーバ）をパックに収集するようシステムを開発し，12月よりデータを収集始める。このデータはファイル容量の関係上3万個までとし，これを越えた時，最初から始める為データはすべてのエラーの種類を含み3万個まで記憶している。また数に制限があるため1件のJŌBについてエラーの数は5個までとして，それを越えたものは集収していない。これらのデータを整理して3月に中級講習会で発表したが5月，9月，12月に整理したデータを表1～表3に掲載する。

表1は4月から11月までの処理件数に対する時間，ページ数，異常終了等を掲げてある。この表のその他というのは，P－ERRŌRが主であり，他の異常終了の種類中のCŌMPILEというのは文法的エラーであり，LIEDというのは結合時のエラーであるがこれは¥LIED文を含んでいる時の文法的エラーを持ったものを含むので実際の数より多くでている。また，RUNは実行時にエラーがでた場合である。

表2，3は文法時のエラーと実行時のエラーを多い順に並べたものであり，エラー番号で表わされているのでFŌRTRAN文法編を参照し，プログラム作成にあたっての注意点を見出しエラーがなくなる様，期待したい。

プログラムの熟練者あるいは指導者に対して，この表が初心者指導に対する役割あるいわプログラム作成に対する注意を促すものとして利用されれば幸いである。ここで初心者に対して簡単ではあるがプログラムの作成に対する注意事項をあげてみよう。

プログラム作成時の注意

プログラム作成した後のチェック・ポイントを文法的なものについて10個かかげた。

(1)　配列の書き忘れがないか
(2)　配列名と変数名が重なっていないか
(3)　DŌの組み合せが正しいか
(4)　文番号が重複していないか
(5)　文番号が抜けていないか
(6)　H型の記述が正しいか
(7)　文の順番が正しいか
(8)　（　）の組み合せが正しいか
(9)　文の使い方が正しいか
(10)　型が一致しているか

1．配列に関するエラー

配列に関するエラーが非常に多く発生しているが，これは定義することを忘れるということが非常に多い為，プログラム作成中非常に注意すべきことである。この配列の未定義はその配列名が使われている文が，すべてエラーとなるので非常に多くエラーが現わる。また配列名を定義してあるが途中の区切り記号に誤りがあった為，それ以後の定義が無効になりエラーが続出する結果となる。特に配列が多くある時，継続行の所で誤りが多いのでこれを減少するには，継続行は多く使用せず幾つかのDIMENSIŌNの定義に分けた方がよい。

2．名前のつけ方

プログラムに際して変数名と配列名に同じ名前をつけてエラーとなることがあるが，これ

は自分なりに名称のつけ方をきめる方がよい。例えば，配列名に関して１ないし２文字の名前・変数に対しては３ないし４文字，又は英文字のあとに数字をつける。そしてサブルーチンや関数名に対しては５ないし６文字の名前をつければ間違いが少なくなるであろう。

3．ＤＯ文の組み合せについて

ＤＯ文に対するエラーも非常に多く表われている。これは特に端末文のつけ忘れが多い。ＤＯ文を作る時の注意として特に気をつけなければいけないのは，制御変数名のつけ方である。端末文のつけ忘れは注意することによってなくすことが出来るが，コンパイルのエラーとして表われない。ＤＯの中のＤＯ文に同じ制御変数をつける事があるが，これはフローチャートを正しく書けば防げるし又，例の様にプログラム上でＤＯ文と端末文を線で結んでみるのもよい。

例

```
┌──── ＤＯ　１０　Ｉ＝……
│     ……
│  ┌─ ＤＯ　２０　Ｌ＝……
│  ├─ ＤＯ　２０　Ｊ＝……
│  │  ……
│  │  ＷＲＩＴＥ（ｂ，……）（Ａ（Ｉ），（Ｉ＝……））
│  │  ……
│  └─ ２０　ＣＯＮＴＩＮＵＥ
└──── １０　ＣＯＮＴＩＮＵＥ
```

しかしながら例のＷＲＩＴＥ文にあるようなＤＯ相当文の制御変数名をＤＯ文の制御変数名と同一にする間違いが特に多いので，線で結ぶのみならずＲＥＡＤ文，ＷＲＩＴＥ文の制御変数名を特に注意する必要がある。これも名前のつけ方を自分で決めておく方がよい。例えばＲＥＡＤ，ＷＲＩＴＥの制御変数名に数値を用いる様にし（例えば，Ｉ１，Ｊ２），その他を用いないという様に区別する。

4．文番号の重複

これは長いプログラムにありがちだが，それよりもプログラムの修正の時に新しく文番号を使用しなければいけない時によく起きるエラーである。このエラーをなくする為に文番号のつけ方に気をつければよい。例えば，ＲＥＡＤ・ＷＲＩＴＥのＦＯＲＭＡＴ文に5000あるいは6000代の文番号をつけ，それ以外はその文番号を避ける様にする。そして最初にプログラムを作る時は，文番号の最後が０で終る様にし，修正の時は最後に０でないものを置く様にすれば避けられる。

5．文番号が抜けていないか

文番号未定義が非常に多くこれの主なるものはプログラムの訂正時において，文番号のついた文を抜いた時に発生しやすい。これはＤＯ文の端末文番号であれば気づくのであるが，ＩＦ文とかＧＯ　ＴＯ文の文番号が抜けやすいので特に注意する必要がある。ここで，これらの文番号の終りが５で終る様にするのも１つの方法であろう。

6．ＦＯＲＭＡＴのエラー

ＦＯＲＭＡＴのエラーにも各種あるが，特に多いのはＨ型記述子である。これはＨに続く文字の数がＨの前の数と一致しない為に起こるものであるが一般にはＨ型が継続行に続く場合，７２カラム目までのブランクもすべて含まれていなければならない。が，しかし多くの場合，継続行で間違えているのでＨ型記述子を書く時に２行に渡りそうな場合は，あらかじめ前の行を避け継続行の最初にもって来た方が良い。

7．文の順番

文の順番を時々間違えている人がありますが，これは正しく認識していなければならない。

順序を次に掲げる。

(1) { ＥＬＥＭＥＮＴ文
　　ＵＳＥＲ文

(2) { ＳＵＢＲＯＵＴＩＮＥ文
　　ＦＵＮＣＴＩＯＮ文
　　ＢＬＯＣＫ　ＤＡＴＡ文

(3) 宣言文

(4) ＤＡＴＡ文

(5) 文関数定義文

(6) { 実行文
　　ＦＯＲＭＡＴ文

8．かっこの組合わせ

このエラーも意外と多く数式中のかっこが多い場合や，あるいはＦＯＲＭＡＴ中のかっこの数が重なる時よく左右のかっこの数があわない時がありますので，注意をして下さい。

9．文の使い方が正しいか

定義文を良く見て正しいかどうかをチェックして下さい。特に区切り記号等に注意すべきです。ＤＩＭＥＮＳＩＯＮ，ＥＱＵＩＶＡＬＥＮＣＥ，ＳＵＢＲＯＵＴＩＮＥ，ＤＯＵＢＬＥ ＰＲＥＣＩＳＩＯＮのつづりをよく間違えていますから注意して下さい。またＧＯ　ＴＯ文をＧＯＴＯまたはＧＯ　ＴＯＸＸのようにブランクをあけることを忘れるのも多いようです。他に関数名でＦＬＯＡＴをＦＬＯＲＴと書き間違える人が多いので定義文のみならず関数名等のつづりにも気をつけて下さい。

10．型の一致

実数型，整数型の区別なく数式中に書く人がよく見うけられますが注意して下さい。数式中ではよく実数型変数名に整定数を乗除したい場合に，小数点を忘れがちですから注意して下さい。又，論理ＩＦ文の場合でも実数型と０を比較する場合，０の側を０．０としなければならないのを，小数点を忘れがちですから特に気をつけて下さい。

実行時のエラーのチェックポイントについて

(1) 未定義の変数名を使っていないか
(2) 割り算で分母に零がこないか
(3) ＡＬＯＧ・ＳＱＲＴの使い方が正しいか
(4) ＥＸＰの使い方が正しいか
(5) 入出力データに誤りがないか
(6) 配列のわくを越えていないか

実行時のエラーが特に多いのが指数アンダーフロー，零ディバイド，オーバーフローである。この原因の最も多いものとして変数の値の未定義によるものが，あげられる。これは変数名を特徴づけた名前を使うことによって避けられる。次に注意すべきことはプログラム・コーディングに際してＩ，／，１あるいはＯと０，２とＺ，ＵとＶ等の区別をはっきり書かない為の，パンチミスである。しかし次の様に区別すると意外と便利であり見やすい。

Ｉ（アイ）――――i　　２（ニ）――――2
／（スラッシュ）――／　　Ｚ（ゼッド）――――Ƶ
１（イチ）――――1　　Ｕ（ユー）――――u
Ｏ（オー）――――$\bar{O}$　　Ｖ（ブィ）――――∇
０（ゼロ）――――0　　Ｄ（ディ）――――Đ

次にベキ乗で低が負か零の場合のエラーが多いようであるが，これはベキ数が整数の場合でも実数型定数あるいは変数を書くためであり，できる限り整数のものは＊＊整数とす

べきである。次にＥＸＰで変数の値が大きすぎるというエラーも非常に多いが，この場合も負の絶対値が大きすぎるものが大部分である。したがって指数関数を使うにあたってＡ＝ＥＸＰ（－Ｘ＊＊２）……と書かず

```
B＝－X＊＊2
C＝0・0
ＩＦ（Ｂ・ＧＴ・－１００・０）　Ｃ＝ＥＸＰ（Ｂ）
……
```

とすべきである。

特にこれはエラーが出る前にプログラムを作る時の習慣として身につけてほしいものである。これと同じ様に零ディバイトの場合も，分母を判定してから計算するよう心掛けた方が良い。また関数を使う時その使用制限をＦＯＲＴＲＡＮ文法編（付録２）を参照した方が良いであろう。また，エラー表に現われていないが非常に多くあるのはＰ－ＥＲＲＯＲであり，これは配列のわくを越えた場合ＣＯＭＭＯＮ文の対応が正しくない場合，SUBROUTINEの引数の対応が正しくない場合，あるいはＥＸＴＥＲＮＡＬ宣言を忘れた場合等である。

文法，実行時のエラーの注意点を初心者向きに書いたが，これらはプログラムが慣れてから注意すれば良い事であり初心者はエラーをおそれず，計算機に出来るだけ多く接することであり，またエラーの数と共に進歩していくであろうから初心者の留意点としておこう。

またプログラムのより一層の上達は他の多くのプログラムに接することであり，自分の殻に閉じこもった場合，それ以上の進歩は望めないであろう。

表１）

		４　月	５　月	６　月	７　月	８　月	９　月	１０　月	１１　月	計
処理件数		５３７	９５９	１２７７	１４５１	１１７６	１５４８	１６５６	２１４７	１０７５１
平均USE時間		5′１１″	5′１２″	5′０４″	5′１３″	5′３２″	5′４２″	5′４０″	5′１８″	5′２３″
平均CPU時間		3′０８″	2′５８″	2′５３″	2′４１″	2′４０″	3′１３″	3′２１″	2′５７″	2′５９″
CPO/USE(%)		６０．４	５７．２	５７．２	５１．５	４８．４	５６．５	５９．２	５５．７	５５．４
平均ページ数		１４	１４	１２	１３	１４	１３	１３	１２	１３
異常件数		２３６	４６８	６０５	６７１	６２７	７６９	７６６	１００４	５１４６
異常件数処理件数(%)		４４．０	４８．８	４７．４	４６．２	５３．３	４９．７	４６．３	４６．８	４７．９
異常終了の種類	COMPILZ時のエラー	５３ (22.5)	９３ (19.9)	１４９ (24.6)	１３８ (20.6)	１３２ (21.1)	１４５ (18.9)	１６８ (21.9)	２２３ (22.2)	１１０１ (21.4)
	LIED時のエラー	１０ (4.2)	２１ (4.5)	３７ (6.1)	４０ (6.0)	３２ (5.1)	３９ (5.1)	４８ (6.3)	５１ (5.1)	２７８ (5.4)
	RUN時のエラー	８３ (35.2)	１８６ (39.7)	２２３ (36.9)	２４７ (36.8)	２４３ (38.8)	２９５ (38.4)	２８６ (37.3)	３４９ (34.8)	１９１２ (37.2)
	TIME OVER	５６ (23.7)	９９ (21.2)	１２５ (20.1)	１４９ (22.2)	１４０ (22.3)	１８０ (23.4)	１６４ (21.4)	２５９ (25.8)	１１７２ (22.8)
	PAGE OVER	１１ (4.7)	１５ (3.2)	１２ (2.0)	１８ (2.7)	１２ (1.9)	２０ (2.6)	２６ (3.4)	３８ (3.8)	１５２ (3.0)
	その他	２３ (9.7)	４９ (10.5)	５８ (9.6)	７５ (11.2)	６４ (10.2)	８９ (11.6)	７１ (9.3)	８２ (8.2)	５１１ (9.9)

（　　）内は異常終了の百分率です

表2）　コンパイル・エラー集計

エラー番号	計
FT　048Y	875
014Y	873
114Y	770
ステートメントナンバーの未定義	742
FT　047Y	352
404Y	327
068Y	311
096Y	266
200Y	234
004Y	208
005Y	161
033Y	159
008Y	144
111Y	136
056Y	117
076Y	95
007Y	90
060Y	81
019Y	76
061Y	74
402Y	73
003Y	70
403Y	67
105Y	66
044Y	65
009Y	61
100Y	54
098Y	52

エラー番号	計
FT　022Y	51
072Y	50
067Y	49
055Y	48
059Y	48
410Y	47
021Y	42
110Y	40
006Y	39
081Y	35
085Y	32
039Y	32
053Y	28
117Y	27
089Y	26
602Y	24
078Y	19
075Y	19
121Y	19
097Y	17
036Y	16
002Y	15
071Y	15
054Y	14
035Y	14
017Y	14
040Y	13
119Y	12

表３）　実行時のエラー集計

エラー番号	計
5054	3088
5055	3069
5056	2799
4100	666
4060	554
4010	512
5050	413
4020	273
3200	212
5404	209
3113	184
4070	158
3214	145
3400	84
3213	70
4040	65
4210	64
5060	48
5051	44
4260	41
5062	40
3211	37
3215	34
4220	32
3112	28
3050	28
3000	28
3212	26
3100	25

エラー番号	計
5414	25
3818	24
3600	23
4300	21
3210	21
3060	16
4270	12
5061	10
5410	10
3111	9
3012	7
3500	7
5066	5
3001	4
3114	2
3705	1
5407	1
5402	1
3052	1

プログラム開発 (1)

計算機センター

従来のＳＳＬでは，目的とした解が得られない事が今までにしばしばありました。そこでセンターでは独自に補間法と多項式のサブルーチンを開発しましたので御利用下さい。

a．関数近似――ニュートン法，ラグランジュ法，連分数近似法の三種類で使い方は同じです。

ニュートン法 (DNWTNS / DNWTND)，ラグランジュ法 (DDLGLS / DDLGLD)，

連分数近似法 (DREANS / DREAND)

単精度	CALL DNWTNS (X, Y, N, XX, YY, SX, SY, EPS, I LL) CALL DDLGLS (X, Y, N, XX, YY, SX, SY, EPS, I LL) CALL DREANS (X, Y, N, XX, YY, SX, SY, EPS, I LL)
倍精度	CALL DNWTND (X, Y, N, XX, YY, SX, SY, EPS, I LL) CALL DDLGLD (X, Y, N, XX, YY, SX, SY, EPS, I LL) CALL DREAND (X, Y, N, XX, YY, SX, SY, EPS, I LL)

Ｎ個の点 (X_1, Y_1)，(X_2, Y_2) …… (X_N, Y_N) が与えられている時，ＳＸなる点の関数値ＳＹを各補間法により求める。

パラメーターの内容

(1) 入力パラメーター

Ｘ―――Ｘ(Ｎ)なる配列で独立変数の値を小さい順に与える。実数型配列名

Ｙ―――Ｙ(Ｎ)なる配列でＸ(Ｎ)に対応する関数値を順に与える。実数型配列名

Ｎ―――ＤＡＴＡの組数。整数型変数名又は整定数。Ｎ≧２

ＳＸ――求めたい補間点のＸ座標を与える。実数型変数名又は実定数。ＳＸは独立変数Ｘの範囲内になければならない。Ｘ(１)≦ＳＸ≦Ｘ(Ｎ)

ＥＰＳ―収束判定値を与える。ＥＰＳ＞０・０

実数型変数名又は実定数

単精度で　10^{-5}程度

倍精度で　10^{-7}程度

(2) 出力パラメーター

ＳＹ――ＳＸに対応する結果の関数値がセットされる。

実数型変数名

ＩＬＬ―サブルーチンからもどった時の状態がセットされる。

整数型変数名

０の時――計算が正常に行なわれた時セットされる

３００００の時―入力パラメーターＮ，ＳＸにエラーがあった時やＸ(１)～Ｘ(Ｎ)が小さい順に与えられていなかった時にセットされる。

なお，ＸＸ，ＹＹは作業用１次元の実数型配列名：ＸＸ(Ｎ)，ＹＹ(Ｎ)倍精度サ

ブルーチンを呼び出す時にはパラメーターX，Y，XX，YY，SX，SY，EPSは倍精度指定の宣言を必要とする。

b．代数方程式——ニュートン法，ベアストウ法で使い方は同じです。

ニュートン法 (NWTNHS / NWTNHD)，ベアストウ法 (BARSTS / BARSTD)	
単精度	CALL {NWTNHS (A, BB, BC, N, EPS, I LL) / BARSTS (A, BB, BC, N, EPS, I LL)}
倍精度	CALL {NWTNHD (A, BB, BC, N, EPS, I LL) / BARSTD (A, BB, BC, N, EPS, I LL)}

$a_1X^N+a_2X^{N-1}+\cdots\cdots\cdots\cdots+a_{N+1}=0$ なる実係数の代数方程式の根を求める。

パラメーターの内容

(1) 入力パラメーター

A————代数方程式の係数を高次より入れる。

A(N＋1)なる1次元の実数型配列名。

A(1)=a_1，A(2)=a_2，……A(N)=a_n，

A(N＋1)=a_{n+1}の様に与える。

EPS—収束判定値を与える。EPS＞0・0

実数型変数名又は実定数。

単精度で 10^{-5}程度

倍精度で 10^{-7}程度

(2) 出力パラメーター

BB，BC——それぞれBB(N)，BC(N)なる根の実数部，虚数部がセットされる。実数型配列名

ILL——サブルーチンからもどった時の状態がセットされる。

0の時————計算が正常に行なわれた時にセットされる

30000の時——A(1)＝0・0の時にセットされる。

その他————収束条件が満足されず解が求まらなかった時にくりかえした回数がセットされる。

倍精度サブルーチンを呼び出す時には，パラメーターA，BB，BC，EPSを倍精指定の宣言を必要とする。

使用法その他くわしい事は，センターまでお問い合せ下さい。

——会議報告——

運営委員会

昭和49年10月29日(火)

1．事務長より医学部運営委員が８月１日付で　大村　裕氏　にかわって　本陣　良平氏　に交代した旨，報告があった。
2．センター増築工事について10月24日に起工式があり，工事完了が３月25日になる予定の旨報告があった。
3．京都地区協議会規程について説明があった。
また，地区協より京大大型センターのマニュアルが無料配布される旨，報告があり，他の東大・名大・大阪大についても無料配布を要望した旨報告があった。
4．49年度使用料金について報告があった。
5．元如　春江　が９月１日より終日分室勤務になった旨，報告があった。
6．11月１日からの時間外使用について討議した結果，使用案通り決定した。
7．増築に際しセンターの工事で計算機に支障があるとき，計算業務を停止することについて了承を得た。
8．予算の振替分について，当初予算より元如さんの理学部からの賃金分を差引いた残りの141万円を理学部に振替えた旨，報告があった。
9．科研費での計算機使用について意見の交換があった。
10．Ｖ・Ｔ・Ｒの導入の件について，現在までの状況について報告があった。

連絡委員・プログラム相談員合同委員会

昭和49年12月３日(火)

1．10月29日の運営委員会会議の報告があった。
2．年末・年始のセンター業務予定について検討した。
3．増築工事に伴いカード保管だなをプログラム室に移動するので各自のカード箱をなるべく早く移してほしいこと。又，カードせん孔機が移動する場合がありうるので，了承願いたい旨報告があった。
4．ジョブの処理時間・優先・打切り等について検討し，了承を得た。
5．大型センターの各マニュアルについて京大・東大について一部寄贈があった。しかし，まだ不足の分や他大学大型センターのマニュアルについては購入するかどうか検討してみることになった。
6．12月27日(金)午後大掃除の為，パンチ室を一時閉鎖することで了承を得た。
7．ＢＯＳ－２のマニュアルを分室にそろえてほしいと要望があった。
答．以前，そろえてあったはずだが，一応調べて不足の分は補充することになった。
8．計算機利用者以外で夜間に出入りする人があるようだから利用者は，お互いに注意してほしい旨報告があった。

速　報・再　録

1．11月１日(金)より計算受付時間が下記の様になりますので御利用下さい。

	時間外受付	終　了	オープン使用
月 ～ 金	17:00～22:00	23：00	23：00～
土	12:30～17:30	18：30	18：30～

2．11月１日(金)より後期料金になります。
料金算定法は下記のとおりです。

使用料金 ＝ A × N ＋ C × M

A：後期料金　(65円／分)
C：ページ料金（５円／ページ）
N：計算時間　(分／件)
M：ページ枚数（枚数／件）

3．年末・年始のセンター業務予定
(1) 時間外受付について
１．12月25日(水)より，１月５日(日)まで休みます。
２．12月より２月までの受付時間を22時までとする。（土曜日は17時30分まで）
(2) 一般受付業務について
１．ＩＢＭせん孔機の休暇中（12月28日の13時より１月５日まで）の使用申込みは，12月28日(土)午前中まで提出。
２．計算受付は12月26日(木)17時まで，返却は12月27日(金)17時までとする。
ただし，出来ない分については連絡する。
３．来年の受付業務は１月６日(月)より。

4．ジョブの処理時間・優先・打切り等について
(1) （平日）

9：00～11：00	BŌS－2(ただしBŌS－2が終り次第Ａジョブをかけます)
11：00～13：30	１－Ｂジョブ，２－Ａジョブ
13：30～18：00	１－Ａジョブ，２－Ｂジョブ
18：00～	１－Ｃジョブ，２－Ａ，Ｂジョブ

（土曜日）

9：00～13：00	１－Ａジョブ，２－Ｂジョブ
13：00～	１－Ｃジョブ，２－Ａ，Ｂジョブ

（保守日）

9：00～９：30	Ａジョブ
13：00～18：30	１－Ａジョブ，２－Ｂジョブ
18：00～	１－Ｃジョブ，２－Ａ，Ｂジョブ

注．Ａジョブは　DEBUG, TŌKKYU, KYUKŌ
Ｂジョブは　KAISŌKU, FUTUU
Ｃジョブは　CYŌKYŌRI, YAKŌU

ただし，工学部以外のジョブは時間帯において優先いたします。

5．打切りについて

平日	１人・１日・10件・120分まで
時間内	１人・１日・５件・40分まで

ただし，保守日と土曜日は時間内
１人・１日・３件・20分とします。

6．増築工事に伴ない下記の事項に注意下さい。

(1) カード保管だなをプログラム室に移動しますので各自のカード箱も10日以降，なるべく早く移して下さい。

(2) やむを得ず計算機を止めざるを得ぬ時がありうるので，その時は速報等でお知らせしますので御了承下さい。

(3) カードせん孔機が移動する場合がありうるので御了承下さい。

7．計算機利用者以外で夜間に出入りする人があるようだから利用者はお互いに注意して下さい。

I／Oチャネル

THE CNIND METHŌD

この研究は，半年余りの間でＣＮＩＮＤ法（分子軌道計算プログラム）を，抗生物質に応用することにより抗生物質の反応性を実験の方面からと，計算の方面から検討していこうというのである。紙面の都合上，実験方法を記す前にまず結果を述べておくと「プログラム未完成の為に，実験保留の状態。」である。以下実験方法というより，実験の為の予備実験について記すことにする。ＣＮＩＮＤは，大枚２０００枚以上のパンチカードを必要とする物である。当金沢センターのビデオテープにより学んだ事を足場に，まずとにかく浅い知識なりにも当センターに計算を依頼した。当然といっていいのか分からぬが，エラーである。主量子数３のｄ軌道の計算を組み込む必要のあるこの多原子分子にとって，当センターの容量が足りないのである。これ以上減量が無理であるので，通例のごとく京都センターに依頼した。京都センターでは，自分でカードリーダーを操作する。これは一見便利なようだが，プログラム完成者が連続的に何回も計算できるのに比べ，未完成者は，プログラム完成まで幾分長い待ち時間を強いられる。未完成者のひがみであればいいのだが。ここでは，細かいエラーも解消できた。しかしよくよく見ると，今までセンターのお知らせと思っていた場所に，花文字に囲まれて「LŌAD－RYŌŌIKI GA KADAI……」と印刷されていた。このミスは，今でも，京都センターの見落としを生じやすい印刷方法の責任だと強く信じている。東京センターである。ＨＩＴＡＣの機種を使用している為，ＨＩＴＡＣの使用書が必要となる。今まではＦＡＣŌＭであった。使用の面でＨＩＴＡＣと大した差はない。再びカードリーダーに執着してみると，ここではひどい失敗をしたのである。こうである。カードリーダーにパンチカードを入れ，シャーシャーという機械の音と共に読み込ましていると，突然，ガッガッガッ，カタン，ピー，と音が鳴り，赤いランプが点燈した。後ろでは多くの使用者が並んでいる。その上どう処理していいか分からないとくると，背筋がぞっとして冷汗が出てくる。使用した者でない限りこの緊迫状態は分からないであろう。とどのつまりは，あらぬ所のボタンを盲滅法に押し，リードを途中で中止してしまうような事になってしまったのである。東京センターの計算では，一応結果が出たものの，アンダーフローの為，ばかに０，０が多くなってしまった。当然の報である。そして現在に至っている。卒業研究の期間も追し迫ってきている上にプログラムが通らなければ，論文の結果，考察の所に，次のごとく書かざるを得ないであろう。「ＣＮＩＮＤとは，まことしやかなプログラムである。」と。

CHŌCK

（薬学・学4. N. H）

広報　Vol. 4, No. 2　正誤表

ページ	行	箇所	誤	正
2	中程	結合編集	$\bar{O}$BJ／MT	¥STŌRE␣ŌBJ／MT
2	下から11	¥VŌLINT	通番〔所有	通番〔, 所有
3	上から19	¥LIED	Bm_1〕, 〔, NEG	Bm_1〕〔, NEG
3	下から11	¥LIBEDT	Bm_1, 〔／MT	Bm_1〔／MT
3	〃	〃	Bm_2〕SIN	Bm_2〕, SIN
11	上から14	¥␣EX	F, FTMAIN	F・FTMAIN

原稿募集要項

1．この広報を有用なものにするため，つぎのような原稿を募っておりますので，積極的なご協力をお願いします。

a　計算機に直接・間接に関係する随想・論説

b　計算機を利用した研究・開発の紹介とプログラミング技法

c　**I／Oチャネル**に載せる情報，利用者・非利用者の声

2．原稿用紙は規定のものを用意してありますので，広報委員かセンターに申し込んでください。

3．原稿は，各キャンパスの広報委員にご提出ください。計算機センターを通していただいても結構です。投稿について不明の点がありましたら，広報委員にお問いあわせください。

編　集　後　記

広報№3の発刊作業が終り，来年度は自由の身に……と思ったのも夢。利用者の少ないキャンパスに居ることがわざわいしてか広報編集にたずさわること3年。その間，原稿募集をしてUser からの投稿が非常に少ないのは残念です。本号ではエラーについて解説してもらいました。どの様なエラーが多いか，自分の経験と照しあわせて非常に興味を感じます。

（A・T）

プログラム相談委員名		
所　　属	氏　　名	TEL
理学部・地　学	河野芳輝	62−4281　(568)
理学部・化　学	佐道　昭	62−4281　(548)
理学部・地　学	松本崧生	62−4281　(568)
医療短大・放射線	小島一彦	62−8151　(494)
工学部・電　気	武部　幹	61−2101　(331)
・機　械	佐藤秀紀	62−2101　(258)
センター	車古正樹	61−2101(291,292)
センター	沼田道代	61−2101(291,292)

センター　電話(0762)61−2101(内線)291.292番
センター(時間外)　電話(0762)61−2108
理学部分室　電話(0762)62−4281(内線)536番

昭和50年2月25日　発行

編集者　金沢大学計算機センター
広報委員会

発行者　金沢市小立野2丁目40番20号（〒920）
金沢大学計算機センター

印刷所　田中昭文堂印刷株式会社