後藤　英一先生外伝

戸島　　淵
GEH01215@nifty.ne.jp

0．小　　序

「外伝」とは岩波国語辞典によれば「本伝からもれた伝記や逸話」のことで用例として「義士外伝」が上げられている。したがって本来は本伝がなければ外伝はあり得ないわけであるが本伝が将来然るべき人によって書かれることを期待しつつここでは「私の見た後藤先生」について語らせて頂く。

1．パラメトロン・コンピュータ

私が北海道大学経済学部から小樽商科大学に移ったのは1963（昭和38）年1月1日のことである。この年の三月に博士号を取得しているので博士学位論文

Concave Programming の Gradient Method

は前の年1962（昭和37）年に書かれたものと思われる。何でこんな話をするか

というとこの博士学位論文の付録にコンピュータプログラムを載せたのでこのことを頼りに往事を少し思い出してみようというわけである。当時私は gradient method なるものに凝っていた。今風に言えば gradient method にはまり込んでいたのである。これはシステムの動きをある微分方程式系で表しその解が目標としているシステムの状態に安定的に収束していくことをいくつかの条件のもとで証明しようというものでいわば分権的決定のモデルである。この分野では小樽商科大学教授（当時）古瀬　大六先生の優れた先駆的業績が既にありおまけに先生自身が開発されたアナログ・コンピュータ（以下アナコンと言う）による解の収束状況のブラウン管上における観察結果まで報告されていた。更に先生は解の収束性を確かなものにするために微分方程式にある項を付加することを提案されその収束性の証明は open problem とされたのである。落穂拾いで細々と生計を立てていた私はここから二つの落穂を拾った。一つはアナコンでやられたことをディジタル・コンピュータ（以下コンピュータと言う）でやってみること，もう一つは open problem に挑戦することである。幸いこれら二つの落穂は実を結んだが後年古瀬先生に open problem の付加項はどうして思い付かれたのですかとお尋ねしたところ先生は言下に「あれは物理的には摩擦なのですよ」と答えられた。このとき私は優れた研究者の頭脳構造を一瞬かいま見たような気がした。

さて最初の落穂はどうなったかと言うとこれがハッキリしている記憶とぼんやりしている記憶が入り交じってどうもあやふやである。確かなことは1962(昭和37) 年頃は日本のコンピュータ業界の黎明期であったことである。実際 IBM の Punch Card System （以下 PCS と言う）がまだ大手を振って使われていた。ひところ自分のコンピュータ歴を誇るのに「私は何しろ PCS 時代からの生き残りですから」という言い方が流行ったことがある。そう言えば私も IBM が開催した PCS の講習会に出席したことがある。戦時中軍人将校だったので自分は戦犯だという男が例のパネル配線によるプログラミングの話をしたのを覚えている。また雑談の時間に「計算機に知能を持たせるには乱数を使うそうですね」と何かで読んだことを受け売りすると男は「私は学者の言うこと

は信じない」と吐き捨てるように言った。いま考えてみるとPCSの講師にいささか「高級（？）」な議論をふっかけたようで慙愧に耐えない。乱数を使うというのはむろんこれによってランダムな要素を取り込もうとするわけでたとえば生物の進化過程における突然変異などのシミュレーションがこれによって可能になることはよく知られている。その後乱数と聞くとこの時の講師の態度を思い出す。いろいろ考えてみるとこれは1961（昭和36）年か1962（昭和37）年のことと推定される。そのうち北海道大学理学部にコンピュータが入ったという話が伝わってきた。これが何年か記憶が不確かだが前後の関係からいくと1962（昭和37）年と考えてほぼ間違いなさそうである。思い起こせば私には北大工学部でプログラミングの講習会を受けた記憶がある。記憶の断片によるとそこで変数の名前の付け方などを習っているのでFortranの講習会かなとも思うが1962（昭和37）年に既に日本でFortranが使えたと言うのは信じがたいことである。その上この講習会が理学部に入ったコンピュータを使うためのものなら工学部でその講習会をやるのは奇怪至極である。ともあれなにがしかの準備のあとにいよいよ理学部のコンピュータを使うことになった。今はもう無いが当時理学部（旧館）の前にコンクリートをうちっぱなしにしたような建物があって理学部計算センターと称していた。ここに2台のコンピュータが鎮座ましましていたのである。一つは日本電気のNEAC 2203G，もう一つは日立のHIPAC 103であった。私はNEAC 2203Gを使うことにした。その間の細かい経緯は省くけれどもこうして古瀬先生がアナコンでやられたことを私がNEAC 2203Gでやってみたときのプログラムを私の博士学位論文の付録にすることが出来たのである。私は機械語でプログラムを組んだ記憶があるが工学部の講習会が何語を対象としたものであったかを知る手がかりをうるためにも今回この記憶を確かめようとしたが引越しのどさくさに紛れて目下手元の博士学位論文が行方不明なので何語でプログラムを書いたかについてはこれ以上立ち入れないしまたそのことはここでの話の本筋とまったく関係が無いことである。大事なことはこの時NEAC 2203GとともにHIPAC 103を見ているということである。NEACについては2203GのGはGiantのGで2203の機能を拡張

したことを意味するという説明があった。またHIPACについてはPがこのコンピュータはパラメトロン（Parametron）という日本人が発明した素子を使用していることを表しておりその意味でHIPAC 103は純国産であると解説された。ちなみにNEACは素子としてトランジスタを使用していた。この時点では，まさか後年このパラメトロンの発明者の知遇を得ることが出来るなどとは神ならぬ身の私は当然知る由もなかった。この発明者こそ誰あろう元東京大学理学部教授，現神奈川大学理学部教授で，かって元東北大学学長の西沢　潤一氏が上げた日本に数少ない真に独創性を持つ学者のひとり後藤　英一先生であった[1)]。後藤先生とパラメトロンについてはこれまでしばしば語られてきたし本伝にきちんと載せられるべき事項でありその上私のよく為すをあたわざることでもあるのでここではこれ以上触れないが一つだけ付け加えておくと商用のパラメトロン・コンピュータは日立のほか富士通でも開発されていた。FACOM　201／202／212がそれらである[2)]。また富士通がIBMにパラメトロンを売り込みに行って体よく断わられたという話もあり当時の日本のコンピュータメーカのハッスルぶりが窺えて興味深い。ここで話は一気に10年以上飛ぶ。

2．Lisp

冒頭で述べたように私は1963（昭和38）年に小樽商科大学に移ったのであるが所属は管理科学科（現在は社会情報学科）であった。当時は今の情報処理センターとほぼ同じ位置に独立の平屋の建物がありそこに沖電気のOKITAC-5090Aというコンピュータが同年八月に入ってきた（これは翌年十一月に，より大型で高性能なOKITAC-5090Hに置き換えられた）。したがっ

1）西沢　潤一氏の独創性か何かを論じた文章にこの記述があった。「我が意を得たり」の指摘なので印象深く記憶しているが何という表題の文章だったか未だ特定できないでいる。したがって今は具体的に引用できないのが残念である。

2）㈱富士通金融システムズ取締役システム本部長　東口　豊氏提供の資料による。

てこの建物は計算センターと呼ばれることになった。この計算センターの海側の部屋は図書館分室として使用されており背表紙に貼られたラベルにｋマーク（kanri-kagaku の頭文字ｋ）の付いた図書をフリーアクセス出来るように開架展示をしていた。私はよくこの部屋に入ってｋマークのついた本をあれこれと読みあさったものである。そのうちに私は奇妙な本を見付けた。その本の題名は

LISP 1.5 Programmer's Manual

といった。著者としては五人の名前

John McCathy
Paul W. Abrahams
Daniel J. Edwards
Timothy P. Hart
Michael I. Levin

が上げられていた。出版社と出版年は

The M. I. T. Press, 1962

である。僅か100数ページしかないこの本のどこが奇妙か？　まず version no. が整数でなくいわゆる実数で表されているのが面白い。最近ではこういう表し方も珍しく無くなったが当時は斬新であった。次に内容であるがこれがなかなか理解できない。無論一つ一つの文章の意味は分かるのだが全体として何を対象として何を説明しようとしているかということが奇妙にもさっぱり見えてこないのである。しかし「数学的言語」として何か途方もなく重要なことが述べられているらしい気配は伝わってきたのでとにかく読み通してみた。今この manual を改めて読み直してみると Lisp として必要かつ十分な説明になっていることがよく分かる。しかしそれは“list”とはどういうデータ構造で“list 処理”とはどういうことか，Lisp でプログラムを組むとはどういうことかなど

などが理解できて初めて分かることでいきなりこのmanualを読んでもそのことに思い当たるものは希であろう。Lispの処理系も殆ど無かった当時はLispは私にとって念仏と変わらないものであった（後年8ビットのパソコンの初期に研究室に最初に入れたソフトウェアはCP／M上で走るmuLISPとmuMATHであった）。ともあれLispというものがこうして未消化のまま私の記憶の片隅に残ることになった。

1974（昭和49）年には共立出版から出ている雑誌bitの一月号より後藤先生の「LISP入門」の連載が始まった。その第一回目の題名は

マッカーシーの条件式とM式

であった。これを見て驚倒しない者がいたとすればそれはLispに縁無きものである。それまでのLisp入門の書き方はまずlistというデータ構造の説明があってそれに対する演算子として*car*や*cdr*などの基本関数が導入されるといった順序を踏むのが通例であった。しかし翻って少し考えてみれば明らかなことであるがLisp programmingの本質的特徴は*car*や*cdr*などを使用するところにあるわけではない。Lisp programmingの本質的特徴は条件式を用いて帰納（再帰）的に関数をうまく定義するところにあるのである。実際，1958（昭和33）年の夏をIBM Information Research departmentで過ごしていたマッカーシー（J. McCarthy）は「当時サンプル問題としていた代数式の微分」のためには「微分は明らかに帰納的に定義されるし条件式を用いれば複数の処理を一個の式に纏めることができる」ので「条件式を用いて帰納的に関数を定義すること」が必要であったと回顧している[3)]。条件式とはマッカーシーが考案した式でM式では一般的には次のように表記される。

$[p_1 \rightarrow e_1; \ldots ; p_n \rightarrow e_n]$

そのsemanticsは*if then else*が*n*重に入れ子になっているAlgol 60の構文：

if p_1 *then* e_1 *else* ... *if* p_n *then* e_n

3） J. McCarthy, "HISTORY OF LISP", *ACM SIGPLAN Notices*, Vol. 13, No. 8, August, 1978, p. 218.

とまったく同じである。このように条件式は評価方法が特殊なので通常の関数表記（たとえば $cond\,[p_1,\ldots,p_n\,;\,e_1,\ldots,e_n]$ など）には馴染まないないため特に考案された表記法でマッカーシーによるとこれは「ベクター記号と同じ種類の革新」で「数学の中で一般的に使用されるようになる」とされたものである[4)]。したがってLisp programmingの論理的説明順序として最初に条件式がくるのは極めて自然でありまた当然でもあった。それ故このことはまさにLispを熟知している者によるLisp programmingの本格的解説の開始を予感させるものであった。メタ言語であるM式を初回から使用するのもLispの構文を厳密に表記する必要性からこれまた当然のことである。連載第二回目の題名は果たせるかな

関数の帰納的定義

である（もっともこうなることは第一回目の末尾で予告されていた)。後藤先生のLisp解説に新機軸をもたらそうという意欲を目の当たりに見る思いである。ちなみに後年1982（昭和57）年1月に出版された後藤先生，石畑　清氏と私との共著

記号処理の基礎と応用　情報処理学会

のLisp解説の部分もこの論理的順序を踏襲している。このようにして始まった連載は翌1975（昭和50）年2月号まで14回続いた。いたるところにある示唆に富んだアイデアの提示と問題点の指摘は計算機科学の共有財産として今もって一読の価値がある。特にこの連載の後半では当時東京大学理学部情報科学科の後藤研究室で開発が進められていた高速Lispインタプリタ HLISP（Hash coded LISP；hash codeというデータ構造を持ち込むことによって処理系の高速化を図ろうというLISP）についてその特有な機能や内部構造が詳細に解説

4）J. McCarthy, "A Basis for a Mathematical Theory of Computation", *Computer Programming and Formal Systems*, North-Holland, 1963, p. 41.

されている。HLISPはソースプログラムがFortranで書かれているのでポータビリティが極めて高いことが期待された。私もこの連載から多大な啓示を得ることが出来た。纏めて一冊の単行本として刊行されていないのが惜しまれる。

3．後藤　英一先生

1974（昭和49）年頃，私は多忙で出張する機会の多い古瀬　大六教授の代理で北海道大学大型計算機センターの運営委員会に出席していた。大型計算機センターは全国共同利用施設であるから運営委員の中には道外大学の委員も混じっていて，東京大学の後藤先生はその一人であった（もう一人は京都大学からの委員であったように記憶している）。後藤先生は勤勉な方で殆ど会議を欠席されることはなかった。私は〈朝早く羽田をたってくるのは大変だろうな〉と思っていたが後に伺ったところでは冬など実際には前日に着かれて翌朝スキーを楽しまれたこともあったそうである。してみると後藤先生は東京・札幌間往復の小旅行をあまり負担と思わず案外楽しんでおられたのかも知れない。さて，1974（昭和49）年6月24日の運営委員会で私はたまたま後藤先生の隣席に座る巡り合わせになった。運営委員会の議事が進んでいく中でふと後藤先生の手元を見るでも無く見ると資料として配られた紙の余白にしきりに何か式らしきものを書いておられる様子が窺えた。当日6月24日にはすでにbit 7月号は発売されていたし間もなく8月号が出ようという時であるから私は直感的に9月号の原稿の準備ではないかと思った。（ちなみに連載9回目の題名は「HLISP」であった）。そこで休憩時間に思い切って後藤先生に声を掛けてみた。まず簡単に自己紹介をした後，先ほどメモしていたのはLispではないのか，bitの「LISP入門」を読んでいると言ったところ話題はたちまちHLISPのことになったので私はHLISPを北大大型計算機センターに移植させて貰えないかと申し出た。その時の細かいやり取りはもう記憶に無いが一応OKということになった。ただしこの時点では私のHLISPの開発状況の認識が少し間違っていたので後藤先生にご迷惑をお掛することになってしまった。私はHLISPがも

う出来上がって完全に動作しているものと思い込んでいたのでそのソースプログラムをMTにすぐ書き込んで貰えればよいと考えて7月19日にはイニシャライズ済みのMTを後藤先生宛に送った。後年この時のことを「MT1巻がドーンと送られて来てね」と言われ私の不躾なやり方に驚かれた様子であった。ここで改めてプレッシャーをおかけするような具合いになったことに対してお詫び申し上げる次第である。実際にはHLISPはデバッグをまだ完全には終えておらずソースを外へ出すにはもう少し時間が必要だったらしい。結局MTが返送されて来たのは秋になってからで9月9日であった。この日以降北大大型計算機センターでHLISPとREDUCE（後述）を動かそうといういわば疾風怒涛の時代がやって来るのだがそのことについては他所で述べた[5]のでここでは繰り返さない。こうして北海道にもLisp文化が花開き先端技術と言ってもよい数式処理がREDUCEを通じて身近なものになったのはひとえに後藤先生の並々ならぬご厚意と強力なサポートの賜であることを銘記しなければならない。

ところで本稿は外伝であるから本伝に載るべき事柄は避けなければならないがせめて後藤先生の経歴だけでも簡単に述べさせて頂きたい。次に掲げるのはInternet上で公開されているものである。

後藤英一，理学博士，　Eiichi GOTO, D. Sc

経　　歴

1931年　東京に生まれる.

1953年　東京大学理学部 物理学科卒

1954年　パラメトロンを発明

電気通信学会論文賞

朝日賞

5）戸島 熙，「北海道大学大型計算機センターへのHLLISP, REDUCEの移植について」，文部省科学研究費による特定研究「広域大量情報の高次処理」総合報告第IV分冊，第5部情報構造研究グループ報告，昭和51年3月，pp. 915－921.

IRE 論文賞等

東京大学大学院，助手，助教授.

1961－62年　MIT 客員準教授.

磁気モノポール探索実験を提案，実施

1968－91年　理化学研究所主任研究員(非常勤)

1970－91年　東京大学理学部教授

1971年　超高精度陰極線管を発明

テレビジョン学会論文賞

市村賞等

1978年　可変面積型電子ビーム露出法を発明

大河内記念賞

科学技術庁長官発明賞

紫綬褒章等

1979年　並列ハッシュ算法

情報処理学会論文賞

1986－91年　新技術事業団創造科学推進事業磁束量子情報プロジェクト

リーダー（兼務）

理化学研究所後藤特別研究室主任研究員（非常勤）

1995年　初等関数の高速計算法の改良と新提案

情報処理学会論文賞

情報処理学会名誉会員

現　　在　無省安長（無公害，省エネルギー，安価，長寿命）熱機械

量子限界感度磁束計

完全磁気遮蔽

超高速論理回路

信頼できるモンテカルロ算法

無発熱計算等の研究に従事

なんと嚇嚇たる成果ではないか！　これらに加えてはhashingの技法を使って実際に高速なLISP，すなわちHLISPを作ってみせたことも明示的な業績であると私は考える。実際，HLISPの与えた衝撃は大きかった。Lisp処理系の速度が問題とされ情報処理学会記号処理研究会（後述するがこの研究会も後藤先生の肝入りで出来たと聞いている）でLisp処理系コンテストが行われたのも明らかにHLISPを意識している。なおHLISPの移植は北大大型計算機センターだけで行われたのでなくより大きいプロジェクトの一環として行われたものである。その結果HLISPは次の機種の上で稼働した。

HITAC 8800/8700/5020
FACOM 230-75/60/45S/48
NEAC 2200-700/500
IBM 370-195
DIPS

懐かしい機種名が並んでいるがこれだけを見てもHLISPのソースプログラムをFortranで記述したのは異機種間のコンパチビリティにとって正しい基本方針であったことが分かる。ともかくHLISPは後藤先生の精力的活動とあいまって日本のLisp communityに衝撃を与えるという大役を立派に果たしたのである。これを業績と言わずして何を業績と言わんや。

以上のことと関連する後藤先生の功績としてあげられるのは情報処理学会の研究会として前述の記号処理研究会を発足させるのに尽力されたことである。記号処理とは言うものの主役は何と言ってもLisp，即ちリスト処理であった。Lispに何らかの意味で関係を持つ日本の主だった研究者がメンバーであったこの研究会は運営も非常にスムーズにいき，あまりにスムーズにいくものだから外部から「仲良しクラブだ」という悪口が聞こえてきたほどである。この研究会が日本のLisp文化に果たした役割は大きい。一つにはそれまであまり成果の発表の機会に恵まれなかった研究者に発表の場を与えたこと（実際私は後藤先生がこのことに言及するのを聞いている），二つには研究者個人なり研究

グループなりがいまどんな研究をしているかという情報がすばやく流れたこと，三つにはそうしたことを通じてLisp理解が格段に深まったことなどを上げることが出来る。特に私のような地方からの参加者にとってはこのうえない知的刺激の場であった。なおこの研究会は今はもうなくなりプログラミング研究会の中に吸収されている。「昔の友は今いずこ」で1980（昭和55）年代を振り返るとこの期間に雑文・駄文の類を書き散らしたり講習会をやったりして次節で述べるREDUCEの認知と普及のために微力を費やしついにマニュアルの翻訳までやってしまった日々が懐かしく思い出されていささかの感慨無きにしもあらずといったところである。

4．数式処理システムREDUCE

後藤先生は東京大学理学部物理学科の高名な高橋　秀俊教授の研究室出身である。その高橋教授の1955（昭和30）年の「電子計算機の現状と将来」という標題の文章の中に次の一節がある[6)]。

> 現在の計算機で普通やるのは，数値を式に入れて計算する算術ばかりであるが，文字の入った式を計算させる代数計算のプログラムも考えられる。これができれば，電子計算機はさらに便利になるであろう。

これはまさに今日の数式処理システムを予見するものである。巷間には高橋教授が数式処理に否定的な見解を漏らしていたと称して頭から数式処理を毛嫌いするものがいるようであるが，もしそれが本当として，「否定」がどういう文脈で行われたかが問題である。その時点でまだまだ真に実用の域に達していないという意味での「否定」もあるわけで上掲の高橋教授の文章から判断すると

6）高橋　秀俊，「電子計算機」，岩波講座，現代物理学，岩波書店，1955年8月，p. 51.

数式処理の「全否定」とは考えられない。そこで実用の域に達しているかどうかの判断の目安にするために数式処理の実例を一つ上げよう。

私はかって数式処理システム REDUCE の数理経済学における利用を論じた後読者に次の問題を出して open problem とした[7)]。

問　題

$$w = y - k\frac{dy}{dk} \tag{1}$$

$$y = aw^b$$

の二式から導かれる微分方程式

$$y - a\left(y - k\frac{dy}{dk}\right)^b = 0$$

の解が

$$Y = \gamma\left(\delta K^{-\rho} + (1-\delta) L^{-\rho}\right)^{-\frac{1}{\rho}} \tag{2}$$

となることを REDUCE で確かめよ。ここで

$$y = \frac{Y}{L},\quad k = \frac{\mathrm{K}}{L}$$

である。

この問題には経済的な意味があるが今はそれは関係がないので述べない。結果は一人の読者の反応のみでしかもいくらやっても出来ないというものであった。以下に私の解答を掲げる。

7）戸島　淵，「Reduce 活用のために」，archive　No. 12，CQ 出版社，1990年 8 月，p. 124.

まず(1)によりwを定義する。

```
w:=y-k*dy!/dk;
```

```
W  :=   -  DY/DK*K  +  Y
```

するとde=0が微分方程式となるようなdeは

```
de:=y-a*w**b;
```

```
                                        B
DE  :=   -  (  -  DY/DK*K  +  Y)  *A  +  Y
```

で与えられる。変数dy!/dkを陽に指定するため微分方程式を変数dy!/dkついて解くと

```
solve(de,dy!/dk);
```

```
                              (1/B)
SOLN(1,1)  :=  (  -  Y       *COS(ARBREAL(1))

    (1/B)          (1/B)
+  A       *Y  -  Y       *SIN(ARBREAL(1)

                              (1/B)
                  )*I)/(A       *K)
```

ここでは多重解になっているので主値に限定するため

```
off allbranch;
```

と指定しもう一度同じことを試みる。

```
solve(de,dy!/dk);
```

```
                    (1/B)        (1/B)     (1/B)
SOLN(1,1) := (A        *Y  -  Y        )/(A
*K)
```

1

ここで便宜上定数変換を施す。

```
a:=alpha**(-b);
```

```
                 B
A  :=  1/ALPHA
```

```
b:=1/(1+ro);
```

```
B  :=  1/(RO  +  1)
```

変数dy!/dkを陽に表すと

```
dy!/dk:=soln(1,1);
```

```
                              RO
DY/DK  :=  (Y*(  -  Y    *ALPHA  +  1))/K
```

となる。これを積分が出来る形にするために次のように変形する。

```
dk!/k:=1/num(dy!/dk);
```

```
                                  RO
DK/K  :=  (  -  1)/(Y*(Y    *ALPHA  -  1))
```

両辺を積分すると

```
log!  k:=int(dk!/k,y);
```

```
                              RO
LOG  K  :=  (  -  LOG(Y    *ALPHA  -  1)  +  LOG(Y)
```

```
*RO)/RO
```

両辺にρを掛けさらに積分定数log βを考えlog U=log Vという形に整理して左辺lhをU、右辺rhをVとすると

```
lh:=k**ro;
```

```
          RO
LH  :=  K
```

```
rh:=y**ro*beta/(1-alpha*y**ro);
```

```
                RO              RO
RH  :=  (  -  Y    *BETA)/(Y    *ALPHA  -  1)
```

が得られる。ここで改めて方程式U-V=0をyについて解くと

```
solve(lh-rh,y);
```

```
                          RO                        (1/RO)
SOLN(1,1)  :=  K/(K    *ALPHA  +  BETA)
```

1

となる。再び次のような定数変換をする。

```
let alpha+beta=gamma**(-ro);
```

```
let beta*gamma**ro=delta;
```

yの値は配列solnの(1,1)要素に入っているから

```
y:=soln(1,1);

Y := (GAMMA*K)/
                RO                RO              (1/RO)
      ( - K    *DELTA + K     + DELTA)
```

となる。以下、大文字Kをkk,大文字Lをll,大文字Yをyyで表すことにすると

```
let k=kk/ll;
```

であるからYは次のように求められる。

```
yy:=ll*y;

YY := (GAMMA*KK*LL)/
                  RO                RO       RO
       ( - KK    *DELTA + KK     + LL    *DELTA

 (1/RO)
)
```

最後に(2)式を入力してYと比較する。

```
yyy:=gamma*(delta*kk**(-ro)+(1-delta)*ll
**(-ro))**(-1/ro);
```

```
YYY  := (GAMMA*KK*LL)/

                RO                RO       RO
      ( - KK   *DELTA + KK    + LL   *DELTA

 (1/RO)
)


yy-yyy;


0
```

これでめでたく(2)式が微分方程式の解であることが示された。

ここではREDUCEの古い版であるREDUCE3.2をあえて使っている（1998年10月現在REDUCE3.7になろうとしている)。それにはちょっとした訳がある。REDUCE3.2がリリースされたのは1985（昭和60）年4月のことであった。日本でREDUCEのサードパーティとして一番早く名乗りを上げたのは当時ベンチャー企業として売り出し中であったBUGで，すでに1986（昭和61）年にはパソコンの上で動作するBUG版のREDUCE3.2が出荷されていた。このシステムは実行時にはFD 1枚の身軽さで処理速度も早く十分実用に耐えた（その意味でこれはパソコン用の傑作ソフトの一つであろう)。そのためこのシステムの普及という形で静かなブームを呼びREDUCEユーザが広がっていったと思われる。1987（昭和62）年7月にはREDUCE3.3がリリースされるとBUGも直ちにパソコン版を出して対応したがこれは軽快なREDUCE3.2に比べて16ビットパソコンには少し重たい感じであった。1988（昭和63）年に私のREDUCE3.3英文マニュアルの翻訳が

A. C. ハーン，REDUCE ユーザーズマニュアル　マグロウヒル（絶版）

として出版されるとこのマニュアルは神田神保町で一時コンピュータ関連書の売行き順位のかなり上位を占めたこともあったようである。ともあれ1988（昭和63）年当時 REDUCE3.2，REDUCE3.3を保有している人がユーザーズマニュアルを買い求めたと考えられるのである。したがっていまだ REDUCE3.2を保有しているものが大勢いると推測される。これが REDUCE3.2をあえて使用した理由なのである。思わず筆があらぬ方向に走ってしまったが上述した問題の解法に興味を持たれた方は是非とも REDUCE3.2で同じものを入力して REDUCE の威力を目の当たりに体験して数式処理が完全に実用の域に達していることを確認してほしい。それでもまだ疑っている人は「縁無き衆生は救い難し」の類であるから無駄な説法は止めることにする。

さて，このような数式処理文化はどのようにして日本に根づいたのであろうか。それも後藤先生の大きな功績の一つである。もともと後藤先生が Lisp に興味を持たれて HLISP を開発されたのは論理的な興味もさることながらその上で数式処理システムを走らせようという目的があってのことであった。北大大型計算機センターでお合いする度に「マキシマを動かしたいですな」と言っておられた。最初に聞いたときは一瞬何のことか分からなかったがしばらくして MIT に MACSYMA という名前の有名な数式処理システムがあることを思い出し納得了解した次第である。HLISP ではさしあたって REDUCE（当時は REDUCE 2 であった。REDUCE 3 は1983年から）を動かすことが目的となっていたが後藤先生の視野には当然 MACSYMA も入っていたに相違ない。私が北大大型計算機センターに移植したのも前の方で述べた通り HLISP-REDUCE 組である。このようにして HLISP が移植されたところでは皆 REDUCE が動いたはずである。この間の事情を後藤先生は次のように述べている[8)]。

8）後藤　英一，「数式処理の現状と展望」，北海道大学大型計算機センターニュース，第8巻，第1号，1976年2月，p. 24.

数式処理用言語には……いろいろなものがありますが，その処理システムの中身には非常に複雑なアルゴリズムがたくさん含まれていますので，そのプログラミングには高級言語をホスト言語として使用する例が多いわけです。……割合よく使われているのがLispであり，数式処理以外の人工知能のプログラムにもLispでかかれているものがたくさんあります。そこで大学間でこういうプログラムを移すにはLispさえうまく働けば非常に簡単にできるという事情があります。……このようなことを考えましたのでまずLispを作ったわけです。……ところがこれだけではまだ困ります。というのは日本の中にはいろいろな機械がありまして……都合によっていろいろな機械を使わざるをえません。一番安上がりの機械は当然手もとにある小型計算機です。それで間に合わなければ東大大型センター……を使うとか，北大大型センター……を使うとか，少なくとも3種類程度の機械を使わないと仕事は進まなくなります。そこでHLISPシステムは多少遅くなることにはかまわずに互換性のためにFORTRANでかきました。ですからわれわれのシステムでREDUCEを使いますと言語が4重ということになります。機械語はFORTRANコンパイラが作るわけです。そのFORTRANをホストとしてLispがあってまたLispをホストとしてREDUCEなどがあるという4段がまえの言語構成になっています。

こうして各地の大型計算機センターなどでHLISP-REDUCEによって数式処理を体験した人は決して多いとは言えなかったにしろ着実に増えていった。今様に言えばいよいよ数式処理ビッグバンが後藤先生のイニシエイションによって始まったのである。その後の数式処理システムと数式処理アルゴリズムの研究開発には紆余曲折があるがそれはまた別の物語に属することなのでここではこれ以上は述べない。いま私が個人的に80年代を回顧するとハイライトは1984（昭和59）年8月21日（月），22日（火）に理化学研究所（埼玉県和光市；以下理研という）で開催された

The Second RIKEN International Symposium on Symbolic and Algebraic Computation by Computers

であったように思われる。後藤先生は理研の情報科学研究室の主任研究員を勤めておられた。上の国際シンポジウムはその研究室でFLATSという数式処理マシンの開発が完成したのを機縁として開催されたものである。海外からの参加者がすごい。

ハーン (A. C. Hearn)
ブックバーガ (B. Buchberger)
スタウトマイヤ (D. R. Stoutemyer)
ユン (D. Yun)
ノーマン (A. C. Norman)
ダベンポート (J. H. Davenport)
マーチ (J. B. Marti)
フィッチ (J. Fitch)
カルメ (J. Calmet)

外の錚錚たるメンバーである。このシンポジウムは80年代に理研が数式処理研究で一つの中心であることを内外にデモンストレートした[9)]。中でも「内」に対する衝撃は日本の数式処理研究に第二の波を生み出しそれが1990（平成2）年8月21－24日に東京で開催された

The 1990 International Symposium on Symbolic and Algebraic Computation （通称 ISSAC '90）

とその後1992（平成4）年に設立された日本数式処理学会につながっていく。この辺りのことは人脈とともに明らかにしておく必要があるがそれは本稿の主

9）本文で開催日の曜日までも書いたのはこの両日がウイークデーであることを示したかったからである。都内の某研究機関ではこの両日特別にセミナーを開くなどして職員に禁足令を出したという。

旨に馴染まないしまた与えられた紙幅にも制限があるのでここでは割愛せざるをえない。いづれにしても後藤先生の直接的または間接的影響力抜きには語れない。釈迦の掌の大きさは無限で孫悟空がいかに遠くへ飛ぼうとその掌の外へは出られないのである。

5．エピソード

最後にエピソードを二つ紹介しておこう。いづれも状況に応じて素早く的確な判断を下せる後藤先生らしさの出ている話である。ただしそれぞれの話には一定の背景があるのでくどくなるがそれも述べる。

出版社の求めに応じて後藤先生と私が共訳した本の１冊は

Laurent Siklóssy, Let's Talk Lisp, Prentice-Hall, 1976

である。これは

LISP 入門，日本コンピュータ協会，昭和56年

として刊行された。同書の訳者あとがきを見ると訳本は全訳ではあるが脚注を二つ削除したことを断わっている。いま原文で脚注が指定されている文と削除された脚注を示すと次の通りである。

本文　These languages serve the same purpose as English does for communication between two men *1, although they certainly do not like English, ...

脚注文＊1　Our subject being a language of communication between a human being and a machine, we do not feel obliged to include considerations about the communication possibilities among more than two men, or between a man and a woman, or among two or more women.

本文　(((HARRY) (POOR HARRY)) HARRY (HARRY)) *2

脚注文＊2　Is your name Harry?

二番目の脚注を削除した理由は自明であろう。日本には針，梁，鍼などの姓はあっても名前がHarryという人はいないだろうからである。しかし一番目の脚注は違う。何気なく読み飛ばすと二人の男性間のコミュニケーションだから二人以上の男性同士や一人の男性と一人の女性間や二人またはそれ以上多数の女性同士のコミュニケーションと確かに違う。ふむふむ，なるほどそれはそうだ，ということに抵抗なくなってしまって何の問題もないように見える。実は私も初めはそう思って何で当り前のことを特に仰々しく脚注に付けたのか著者の意図を図りかねていた。ところが後藤先生はひと目脚注を見るなり「これはウーマンリブだ」と言われた。この一言で私の目から鱗が落ちた。つまりこういうことである。通常manは「男の人」を意味する。

Man is stronger than woman.

というと

男は女より強い。

と言っていることになる。ところが英語ではmanを男女の区別なしに「人間」という意味で使うことがある。たとえば

Man is mortal.

は

人間は死ぬ。

という意味であって決して

男の人は死ぬ。

という意味ではない。したがってtwo menも解しようによっては「二人の人間」とも取れる。すると，一組の男女，男性二人，女性二人の3通り（性別のみが問題とする）の場合があることになる。

さて，ウーマンリブの主張は色々あるけれどその一つに男を意味するman

が男と女を合わせた人間を表すものとするのはけしからん，男女平等に反するというのがある。このことを頭の中にしっかりと覚えてから第一の脚注文を読んでみよう。するとこの脚注文は訳注をつけて次のようにすると分かり易いことが理解されよう。括弧内は訳注である。

（ここで two men をどう解するかというと）われわれの主題は人と機械の間のコミュニケーション言語なので二人以上の男性間，または男性と女性間，二人またはそれ以上の女性間でコミュニケーションが可能かどうかについて考察を加えねばならないとは感じない。（したがって two men は「二人の男性」を意味しウーマンリブの主張に沿ったものになる）。

しかしウーマンリブと英単語 man の特殊性にこだわることは Lisp とは何の関係もないので後藤先生と相談の上この脚注は削除したのである。

1950（昭和25）年代の一時期数理経済学では何期かの国民所得の間に成り立つ関係を漸化式（Recurrence Formula）で表して国民所得の変動過程を分析するという手法が流行ったことがある。たとえば経済学的にはまったくノンセンスであるがただ例示のためにのみ次のようなモデルを考えてみよう。t 期の国民所得 $Y(t)$ は同期の消費 $C(t)$ と投資 $I(t)$ の合計であるから

$$Y(t) = C(t) + I(t)$$

となる。消費が国民所得の一定割合を安定して占めるとすれば

$$C(t) = aY(t)$$

がえられる。また前期と前前期の国民所得の差額に応じて投資誘因が存在するとすれば

$$I(t) = b\,(Y(t-1) - Y(t-2))$$

がえられる。ここで a，b は経済構造，技術，習慣などから決まってくる比例定数である。これらの三式より次の漸化式が求められる。

$$Y(t) = \frac{b}{1-a}\,(Y(t-1) - Y(t-2))$$

数理経済学では漸化式の未知関数を陽に求めるのでこれを定差方程式（Difference Equation）と呼んで関数方程式とみなしていた。数学ではこのほか差分方程式ともいう。上の式は，微分方程式にならっていえば，定数係数線型二階同次定差方程式である。この種類の定差方程式は二次式の特性方程式の根を求めて一般解を作り初期条件から未知定数の値を定めるというルーチン的方法で解けるため私はもっぱらこの種類の漸化式を愛用した。以下では漸化式と呼ぶのをやめて私にとっては使い古した定差方程式という用語を使う。私は修士学位論文

経済変動の動学分析

で定差方程式を使ったので定差方程式には特別の思い入れがある。特に比較的簡単な定差方程式の特解なり一般解なりがなかなか求められないのに興味を持った。以下よく知られた例を上げておこう。n, m は非負整数とする。

$$fac\ (n) = n \cdot fac\ (n-1)$$

という変数係数線型同次一階定差方程式が初期条件

$$fac\ (0) = 1$$

に対して特解

$$fac\ (n) = n!$$

を持つことはよく知られている。fac は階乗を計算する関数である。定数係数線型同次二階定差方程式フィボナッチ（Fibonacci）数

$$fib\ (n) = fib\ (n-1) + fib\ (n-2)$$

も初期条件

$$fib\ (0) = 0,\ fib\ (1) = 1$$

に対して特解

$$fib\ (n) = \frac{\frac{(1+sqrt(5))^n}{2^n} - \frac{(1-sqrt(5))^n}{2^n}}{sqrt\ (5)}$$

が知られている。ここで qr は平方根を取る関数である。REDUCE によるこの特解の導出は別の機会に試みたので，それを参照してほしい[10]。やや複雑な例として二変数の

$$c\,(n,\,m) = c\,(n-1,\,m-1) + c\,(n-1,\,m)$$

という定差方程式の境界条件

$$c\,(n,\,n) = 1,\quad c\,(n,\,0) = 1$$

に対応する特解は次の形になることが知られている。

$$c\,(n,\,m) = \frac{fac\,(n)}{fac\,(n-m)\cdot fac\,(m)}$$

$c\,(n,\,m)$ は二項係数である。ところが二項係数の定差方程式をほんの少し改変した次の変数係数線型同次一階定差方程式

$$s\,(n,\,m) = s\,(n-1,\,m-1) + m\cdot s\,(n-1,\,m)$$

の境界条件

$$s\,(n,\,m) = 0\ for\ n < m$$

$$s\,(0,\,0) = 1$$

$$s\,(n,\,0) = 0\ for\ n \neq 0$$

を満たす特解の関数形はまだ知られていないようである。これは私の気になる問題の一つである。$s\,(n,\,m)$ をスターリング（Stirling）数という。

さて，前掲の共著を書くにあたって私は関数の帰納的定義の実例として従来引用される機会が多かった fac の変数係数線型同次一階定差方程式よりも特解の関数形が不明であるにもかかわらず任意の引数と初期条件に対応した関数値が計算されるスターリング数の変数係数線型同次一階定差方程式の方が興味深いのではないかと考えた。そしてその旨を後藤先生に申し出てみた。すると後藤先生の考え方はだいぶ違っていた。後藤先生の考えはこうであった。関数の帰納的定義の説明に用いられる式はシンプルであればあるほどよい。なぜならシンプルでなければ式の意味に気を取られてしまって肝心の帰納の説明の理解

10) 戸島　熙，「Reduce 活用のために」，archive　No. 12，CQ 出版社，1990年 8 月，pp. 115－116.

がおろそかになってしまうであろう。よってここはマンネリに見えるかも知れないが意味が直感的に明らかな *fac* でいきたい。そう言われてみると誠にもっともな話であってスターリング数の提案は取り下げとなった。クヌース（D. E. Knuth）によればスターリング数は n 個の要素からなる有限集合を，互いに素でかつ空でない m 個の部分集合に分割する方法が何通りあるかを表している[11]。案外この辺から特解を見いだす道が開けてくるような気がしないでもない。とにかく共著の関数の帰納的定義の説明は伝統的に *fac* から始めている。

6．結　　語

本稿はあくまで「私の見た後藤先生」，「私の感じた後藤先生」をよりどころにしている。そのため主観性が非常に強い。そうした立場からもっと記録しておきたいことがあるが既に紙数も尽きかけた。他の機会を待つほかないが最後にただ一言付け加えるならばアイディア無限の後藤先生の落穂を徹底的に拾いまくる人が出ることを期待するものである。

本稿が将来書かれるであろう「後藤　英一先生本伝」を補完するものになっていれば幸いである。なお記憶違いや事実誤認による誤りは再録の機会をまって訂正するのでE-mailで私まで連絡してほしい。

---

11) D. E. Knuth, *The Art of Computer Programming, Volume 1/Fundamental Algorithms, Second Edition*, Addison-Wesley, 1975, p. 73.