

# 数学教育で用いるための BASIC 処理系の試作

白石 和 夫

## Creating an Experimental BASIC System for Use in Mathematics Education

Kazuo Shiraishi

### 要約

国際標準化機構(ISO)の定める Full BASIC 規格は数学教育にとって魅力的な内容を含んでいる。Full BASIC に近い処理系として現在入手可能な True BASIC には、十進演算でないことや、グラフィック描画領域が機種依存であることなどの不満な点がある。一方、True BASIC には行番号不要であることなど実用的に優れた点もある。これらの長所を取り入れた BASIC 処理系を用意し、数学教育におけるコンピュータ教育をより内容のあるものに変えていく基礎を作りたいと考えた。今般、機能的には必要と考えられるものの一部であるが、一応、使用可能なものが完成した。

### 1. 研究の目的

#### (1) はじめに

これからの数学教育においては、狭い枠の中にはまった知識・技能よりも、数学的な手法を身につけ、創造的に数学を生かしていく人間の育成が求められる。そのような教育を考える上で、コンピュータは避けて通ることのできないものである。

平成6年度から実施される新教育課程で、高校数学においては、選択科目にコンピュータに関する内容が大きく取り上げられている。しかし、その内容に関する研究が十分になされないまま、新課程の実施が始まろうとしている。

数学教育におけるコンピュータ教育は、既存の特定のコンピュータに適合したプログラムを書く技能を修得させるのが目的でもないし、ある種の CAI ソフトのように既存の理論の注入を目的とするものであってもならないであろう。

#### (2) 望ましいコンピュータ教育

数学教育におけるコンピュータ教育の目的は何であろうか。

まず、第一には、離散数学の一部としてのアルゴリズムを学ばせること、あるいは、コンピュータのプログラミングを通して離散数学を学ばせることであろう。

もう一つは、生徒自身の考え方に基づいて事実を追求するための道具としてのコンピュータ利用技術の修得をめざすことである。コンピュータのプログラミング技能を修得させることができれば、個々の生徒が自分で納得し確実であると考えている知識に基づいて問

題を解決するプログラムを書くことができる。そのなかから数学的事実を発見していくような学び方ができれば理想的であるし、現在の数学教育のなかで欠けている創意工夫の意欲を引き出すこともできるであろう。

そのためには、本質に関係ない部分を切り捨て、コンピュータの都合によって不必要に複雑なプログラミングを強いることのないプログラミング言語が求められる。

### (3) 本研究のねらい

このように考えた場合、国際標準化機構 (ISO) が定めたプログラミング言語 Full BASIC はたいへん魅力的である。Full BASIC は、J. G. Kemeny, T. E. Kurtz らの True BASIC を土台にして作成されたものである。True BASIC は教育的にみて優れたシステムであるが、Full BASIC 規格で定められた十進演算を採用していないなどの弱点をもっている。その一方で、使いやすいスクリーンエディタを備え、行番号のないプログラムの編集と実行を可能にするなど、Full BASIC 規格にない長所もある。

本研究のねらいは、Full BASIC と True BASIC の長所を兼ね備えた言語システムを提供することにより、数学教育におけるコンピュータ教育が望ましい方向に進むことができるようにすることである。この試作処理系が刺激となって、各メーカーから本格的な Full BASIC 処理系が提供されるようになれば、さらに望ましいことである。

## 2. 試作処理系の概要

### (1) 対応機種と開発手段

対象とする機種は NEC PC-9801 であり、開発には、ボーランド社の Turbo Pascal 6.0 を用いている。Turbo Pascal 6.0 は C++ 類似のオブジェクト指向言語であり、Turbo Vision と呼ばれるライブラリが用意されている。また、アセンブラによる記述も可能である。今回の開発においては、これらの機能

を最大限に利用している。したがって、残念ながら、Turbo Pascal が使えない機種への移植性には乏しい。

処理系の基本構造は、インタプリタであるが、実行開始前に一度プログラムを通読し、構文のチェックと実行のための準備処理を行っている。そのため、速度が遅いこと以外には、インタプリタの欠点は現れていない。しかしながら、今後の機能拡張を考えるとこの構造は限界であろうとも考えている。

### (2) スクリーンエディタと管理機構

普通のワープロと同様に操作できるスクリーンエディタを用意している。ただし、BASIC は行の概念をもつ言語であるので、論理的な 1 行は画面上でも 1 行で表示される。1 行の長さの上限は 255 文字である。画面上でのエディタの横幅を越える部分については横方向スクロールによって見ることができる。また、縦方向のスクロールによって画面の上下の幅に収まらない部分を読むことができる。

画面上部左方には、メニューが表示されている。なお、メニュー関連部分は、かなりの部分が Turbo Vision からの借り物である。

画面下部左方は、システムが使用する領域であり、現在の状態やエラーメッセージなどが表示される。

画面の上下各 1 行を除いた領域がプログラム領域である。ただし、画面の右側の領域はグラフィック出力と、また、上 6 行を除いた下 17 行はテキスト出力と共用する。出力用に表示領域を譲ったときには、縮小された領域内でスクロールする。プログラムの編集のためのコマンドとしては、行移動、行複写、行削除などが用意されている。これらは、メニュー選択によって実行を開始する。

print 文による出力は、一つのテキストエディタに収納され、その表示領域は、print 文が実行されることによって画面下部に出現し、その後の print 文の実行につれて表示領域が上方に拡大していく。ただし、画面上の

表示行数の最大値は16行であり、それを越える出力があるとスクロールして表示する。プログラムの実行終了後は、メニュー選択によって、出力用のテキストエディタにカーソルを移動し、スクロールして全体を見ることができる。必要ならば、ディスクに保存したり、プリンタで印刷することもできる。場合によっては、その際、編集を加えることも可能である。

グラフィックスの出力は、画面の右側400ドット×400ドットの領域に出力される。この領域は、プログラムのウィンドウ、テキスト出力のウィンドウと共用である。グラフィック文が実行されるとこの領域が確保され、プログラム、テキスト出力の両ウィンドウは幅が狭くなる。

テキスト出力用のウィンドウには、参照用のファイルを読み込むことができる。また、そこから必要な部分を複写してプログラムの編集に利用することもできる。参照用のファイルを読み込むディレクトリは、起動ディレクトリとカレントディレクトリとで選択できるようになっている。起動ディレクトリには言語の解説を目的としたファイルをおいておくことを想定している。カレントディレクトリからは、以前作成したプログラムを参照のために読み込むことを想定している。

本システムで用いているファイルは、プログラム、出力ともにMS-DOSの標準テキストファイルである。たとえば、MS-DOS上のTrue BASICで作成したプログラムを本システムに読み込むことも、その逆も、問題なく可能である。

グラフィックの出力は、プログラムやprint文によるテキスト出力と同時に画面上で見られるようになっている。グラフィック出力が邪魔になれば一時的に隠すこともできる。NEC PC-PR系のプリンタを用いれば印刷も可能である。ただし、色の違いは無視して単色で印刷する。

### (3) 言語仕様

基本的にISO Full BASIC規格に準拠させているが、意図的に規格と相違する処理をおこなっている部分が存在する。また、単純に技術的理由から規格通りになっていないところも多い。

なお、Full BASIC規格は、参考文献1)によった。用語等も、文献1)に従っている。まもなく、JIS規格が出版されるものと思われるが、JISと若干の相違が存在する可能性もある。

#### ①行番号

Full BASIC規格では、プログラムのすべての行に行番号がついていることが要求されているけれども、ここでは、反対に、行番号がないことを原則としている。ただし、それでは、規格に従って作成されたプログラムを読み込んで実行させることができなくなってしまうので、インタプリタは行番号を読み飛ばすようにしている。ただし、行番号参照の制御文は使用できない。

#### ②数値

数値はすべて16桁の十進小数として扱っている。加減乗除の演算は16桁の精度をもっている。べき乗の演算や組込み関数については、できるだけ規格に近づけるように工夫しているけれども、規格に定められた通りの精度を包括的に保証する段階には達していない。

その理由の一部は、組込み関数の計算の主要部分をTurbo Pascalの組込み関数に依存していることである。Turbo Pascalは組込み関数の精度について具体的には言及していないし、実際、Turbo Pascal組込みの正弦、余弦関数に異常な箇所を発見して(補注1)、その部分を避けて利用している。

べき乗の演算については、底が1に近い数値の場合の誤差の低減に留意している。図①は、 $n = 1, 10, 10^2, \dots, 10^{16}$ について

$$\left(1 + \frac{1}{n}\right)^n$$

```

ファイル 編集 実行
let e=2.718281828459045↓
for i=1 to 16↓
  let k=10^i↓
  let x=(1+1/k)^k↓
  print i,(1+1/k),x,x-e↓
next i↓

1          1.1          2.5937424601      -.124539368359045 ↓
2          1.01         2.704813829421526  -.013467999037519 ↓
3          1.001        2.716923932235892  -.001357896223153 ↓
4          1.0001       2.718145926825225  -.00013590163382 ↓
5          1.00001     2.71826823717449   -1.3591284555E-5 ↓
6          1.000001    2.718280469319377  -1.359139668E-6 ↓
7          1.0000001   2.718281692544966  -1.35914079E-7 ↓
8          1.00000001  2.718281814867636  -1.3591409E-8 ↓
9          1.000000001 2.718281827099904  -1.359141E-9 ↓
10         1.0000000001 2.718281828323131  -1.35914E-10 ↓
11         1.00000000001 2.718281828445454  -1.3591E-11 ↓
12         1.000000000001 2.718281828457686  -1.359E-12 ↓
13         1.0000000000001 2.718281828458909  -1.36E-13 ↓
14         1.00000000000001 2.718281828459032  -1.3E-14 ↓
15         1.000000000000001 2.718281828459044  -1.E-15 ↓
16         1          1          1          -1.718281828459045 ↓
F9 Run F10 Menu

```

図1

を計算したものであるが、ほとんど理論通りの数値が得られている。

三角関数についても規格に近づける努力をしている。BASICの組込み関数PIの値は、16桁の精度では、

$$PI = 3.141592653589793$$

であるので、

$$\sin(PI) = \sin(\pi - PI)$$

$$\approx \pi - PI$$

$$= 2.384626433832795 \text{ E-16}$$

となるはずであるが、実際にその通りの結果が得られる。

なお、angle 選択子に DEGREES を指定した場合には、 $\sin(180)$  は正確に 0 になる。

#### ③データ

基本的に数値データの処理のみであり、文字列やファイルを扱う機能は用意していない。ただし、print 文などで、文字列定数を書くことはできる。

数値変数の初期値の扱いは規格の推奨通りになっていて、未代入の変数の値を参照しようとすると、例外状態になる。これによって、変数名の打ち間違いによるバグをかなり減らすことができる。

数値配列の宣言は dim 文で行い、declare 文を用いることはできない。宣言の内容はほぼ規格通り可能である。処理の都合で、添字として使える数値の範囲に制限を設けているが、実用上、問題になるものではない。

#### ④制御構造

制御構造として、if 文、if 区、do 区、for 区を用いることができる。ただし、if 文は、

IF 論理式 THEN 単純実行文  
の形式のみである。

論理式中の AND、OR の解釈は規格で定められた通りに行く。たとえば、 $p \text{ AND } q$  で  $p$  が偽であるときには  $q$  の評価を行わない。

行番号分岐の制御文を用いることはできな

い。select 区は数学で用いることはほとんどないと思われるので用意していない。

利用者定義関数として内部関数定義が利用できる。すなわち、def 文による関数の定義と、function 行で始まる複数行からなる関数定義が可能である。ただし、配列引き数は使えない。外部関数定義の実現はやや面倒であり、基本構造を変更するときにまとめて実現することにしたい。

#### ⑤文と組込み関数

数値組込み関数については、文献1)p. 21の表5. 1に掲げられているものはすべて用意した。ただし、双曲線関数のような誘導関数については、定義式通りにプログラムしただけなので、関数値が0に近い場合には桁落ちのために十分な精度が得られていない。また、数値組込み関数であっても上記表5.1にないもの、すなわち、文字列、配列、例外処理等に関係するものは用意していない。

宣言文と単純実行文については、利用可能なものは次の通りである。なお、文字列変数やファイルに関する部分は利用できない。また、(\*)を付加した文については、規格で定められた機能の一部しか実現されていない。

宣言文

DIM

DATA

OPTION BASE

OPTION ANGLE

単純実行文

(代入・入出力)

LET (\*)

PRINT (\*)

INPUT (\*)

READ (\*)

MAT (\*)

(制御)

EXIT DO

EXIT FOR

(図形)

SET WINDOW

SET LINE COLOR

SET POINT COLOR

SET POINT STYLE

PLOT LINES

PLOT POINTS

GET POINT

#### ⑥拡張機能

グラフィック関係で若干の拡張機能を用意している。拡張の方法は規格の許す範囲である。また、リアルタイムのマウス入力に関するもの以外は、完全な規格合致処理系であればその範囲でユーザーが定義できるものである。

リアルタイムのマウス入力

GET MOUSE : x, y, s

軸を描く絵定義

DRAW AXES (a, b)

格子を描く絵定義

DRAW GRID (a, b)

#### ⑦意図的に規格と異なる処理を行う箇所

結果の表示については、意図的に規格と異なる処理をしている。

指数部なし形式を用いると指数部あり形式を用いた場合よりも所定の印字欄に有効数字部の桁数を増やして表示することが可能である場合には、そのように処理している。これは、合理的な処理方法だと思うけれども、規格には合致しないやり方である。

このように処理した結果、指数部が-4以上の数値は16桁の精度で表示されることになり、結果を表示する際に丸めが行われない。たとえば、

print 1/3\*3

の結果は、

.9999999999999999

と表示される。

#### ⑧True BASIC との相違

Full BASIC 規格に合わせた結果、True BASIC と異なる処理を行う箇所が存在する。

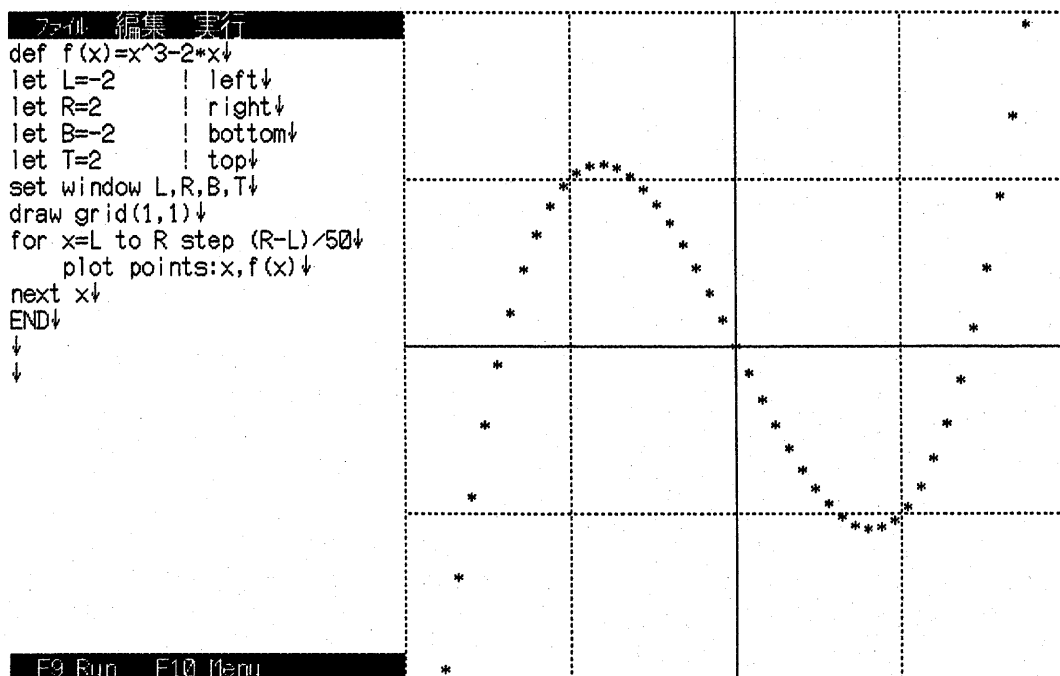


図 2

そのいくつかはすでに述べたが、それ以外にも、たとえば、plot points 文を実行すると、図②に示すように、星印 (\*) を用いて点を描くなどの相違がある。

また、拡張機能として用意した get mouse 文の第 3 番目のパラメータの意味は、True BASIC と異なる。

なお、True BASIC 互換のために plot 文の文法を拡張し、

PLOT LINES : X, Y;

を

PLOT X, Y;

のように書くことを許している。

### 3. 数学教育への適用

#### (1) グラフィックスの利用

関数のグラフや媒介変数表示の曲線を簡単に描くことができる。

描画領域を規格に合わせて正方形としてあ

るので、True BASIC と比較して、画面の実際のサイズに合わせて縦横比を補正して座標系の設定を行う煩わしさが無い。また、軸や格子を描くルーチンは組込みの絵定義として用意してある。

たとえば、極方程式  $r = 2(1 + \cos \theta)$  で表される曲線はつぎのようなプログラムで描くことができる。

```
set window -5, 5, -5, 5
draw grid(1, 1)
for t = 0 to 2*pi step pi/180
  let r = 2*(1 + cos(t))
  plot lines : r*cos(t), r*sin(t);
next t
end
```

上のプログラムに

get point : x, y

print x, y

の2行を追加すると、マウスで画面上の点を指定すればその点の座標を得ることができる。

get point 文は、関数のグラフを描いて、極大、極小となるときに  $x$  や  $y$  の値の概略を知りたい場合などにも役に立つ。

点  $(x, y)$  の偏角を求める関数  $\text{ANGLE}(x, y)$  も用意してあるので、極座標に関係した問題を考察するのにも都合がよい。たとえば、複素数平面上で点  $z$  が点  $1$  を中心とする半径  $1$  の円を描くときの点  $z^2$  の軌跡は、 $z$  の絶対値を  $2$  乗し、偏角を  $2$  倍にした点を描けばよいのだから、図③に示すようなプログラムで実行できる。

get point 文や get mouse 文は、幾何教育の活性化に役立つであろう。たとえば、条件を満たす軌跡を求める場合、get mouse 文を用いて試行錯誤的に条件に合う点をさがし、曲線の全体像を得るような使い方が可能である。

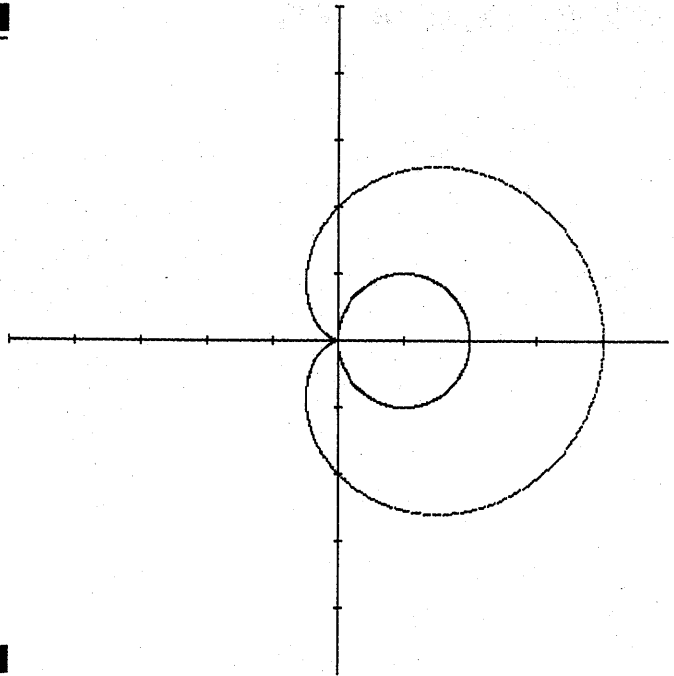
また、get point 文の機能を利用して、図形に関して生徒にいろいろな工夫をさせることのできる問題を用意することができる。たとえば、2数  $a, b$  を入力すると画面上に直線  $y = ax + b$  を描き、次に get point 文を利用して画面上の点を指定すると、その点から、直線  $y = ax + b$  に下ろした垂線を描くプログラムを作成させるような課題が作れよう。

グラフィック機能が使いやすいことは、確率を理論的に計算した結果、あるいは乱数を用いたシミュレーションの結果などを容易にグラフ化できることを意味するから、確率統計の学習においても有用であることが期待できる。

## (2) 三角比 (図形の計算)

True BASIC と共通の性質であるが、角の大きさの単位を度に変更する宣言文があって、なお、かつ、逆余弦、逆正弦の組込み関数を用意されているので、正弦定理や余弦定理を

```
ファイル 編集 実行
! 複素数平面上でzが1を中心とす
set window -5,5,-5,5↓
draw axes(1,1)↓
for t=0 to 2*pi step pi/360↓
  let x=cos(t)+1↓
  let y=sin(t)↓
  plot x,y↓
  let r=x^2+y^2↓
  let a=2*angle(x,y)↓
  plot r*cos(a),r*sin(a)↓
next t ↓
end↓
↓
```



F9 Run F10 Menu

図 3

応用して図形の計量を数値的に扱うのに適している。たとえば、三角形ABCの3辺の長さ $a, b, c$ を入力して角Aの二等分線ADの長さを求めるプログラムなどを考えさせることができるだろう。同様な手法によってさまざまな図形の計量を扱うことが可能であり、それによって正弦定理、余弦定理の価値を学ばせることができる。ただし、平成6年度から始まる新学習指導要領では、逆余弦や逆正弦を扱えないという障害がある。

### (3) アルゴリズムの表現

BASICの構造化構文はプログラムのロジックを明快に表現する点で優れたものである。

数値配列を用いて、分割数など組合せ論に現れるいろいろな数を再帰的な公式に基づいて計算することなどが可能である。

また、内部関数定義では再帰呼び出しも可能であるので、ユークリッド互除法により最

大公約数を求める関数を再帰的に定義することも可能である。ただし、再帰処理に8086のシステムスタックを用いているので、少しネストが深くなるとエラーになってしまう弱点がある。

### (4) わかりやすさと扱いやすさ

BASICの構造化構文は、プログラムの作成や修正を容易にする点でも優れている。試行錯誤的にプログラムを作成するのに、構造化構文と行移動が可能なエディタの存在が有効である。

十進演算の採用によって、十進小数を二進小数に変換して処理することにより派生するトラブルから解放される。また、十進演算は、桁落ちなどの現象が発生した場合にもそれを容易に察知することができることを意味している。たとえば、図④は、正弦関数について増分を少しずつ小さくしながら平均変化率を

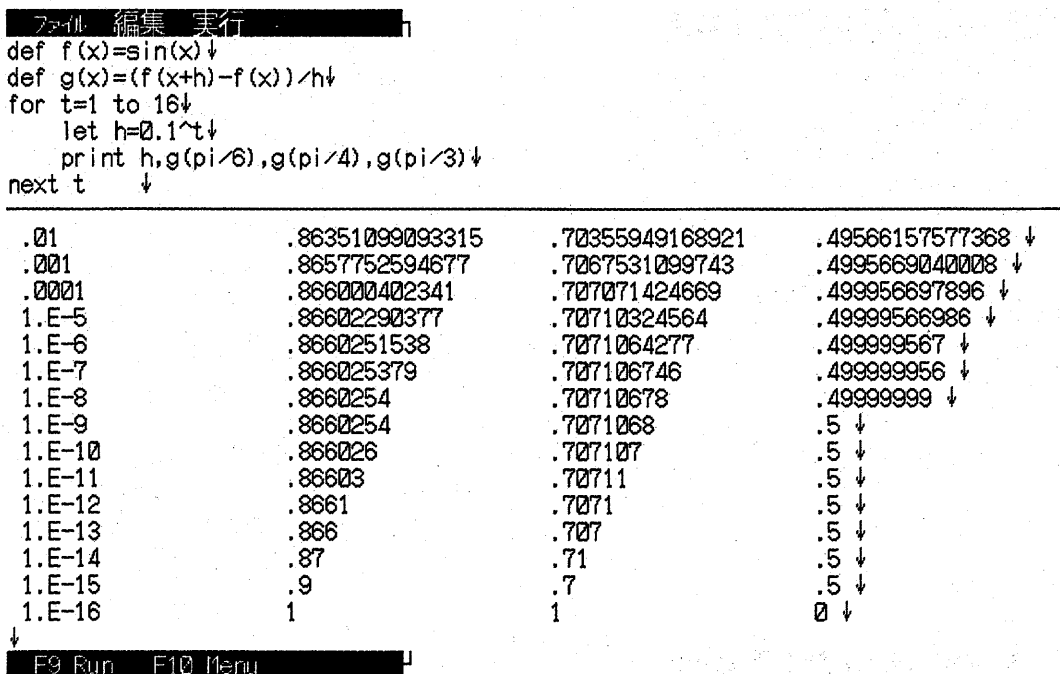


図 4



計算して、それが微分係数に収束する様子を見ようとするプログラムであるが、桁落ちのために十分な精度が得られていないことが一見して分かる。

機能語の綴りの誤りや、区の構造の誤りなど、文法上の誤りは実行を開始する前に検出する。したがって、単純なミスを修正するためにプログラムを無駄に走らせる必要がない。

また、マイクロソフト系 BASIC との互換はないから、マイクロソフト系 BASIC の特徴、たとえば、真偽値が数値であることや、マルチステートメントの存在、一行 IF 文のネストが許されること、などに伴うトラブルとは無縁である。

#### 4. 終わりに

##### (1) 数学教育の今後

数学教育の今後を考えると、解析学については、ほぼ、どのように進むべきか方向は見えてきたといってよいだろう。焦点は離散数学である。離散数学は何をどのように扱うべきであるのか、どんな教材を扱うことが可能であるのかについて研究を進めていく必要がある。また、本稿でも、一部、述べたように、BASIC の図形機能は今後の幾何教育の

方向を考える上で示唆にとんだ内容をもっている。BASIC の図形機能を活用して、あるいは、その活用を目標にして、教育を行うことにすれば、現在とはかなり異ったアプローチが可能である。

##### (2) 本研究の今後

本試作処理系は、今後、かなりの機能追加が必要である。とりわけ、BASIC の特徴である図形の変換を行う機能は、True BASIC において実現されているものであるが、変換幾何学への入門として優れたものである。

また、十進数ばかりでなく、長精度の有理数をも扱えるようにすることを計画している。有理数演算の実現によって、なお、いっそう気軽にコンピュータの利用が可能になると考えている。

(補注 1) Turbo Pascal 6.0 の組込み関数  $\sin(x)$ ,  $\cos(x)$  は、 $\pi/4$  から始まるわずかの区間で定数関数になっている。

#### 参考文献

- 1) 西村怒彦他編：アメリカ規格 Full BASIC—全訳と解説一、共立出版、1990
- 2) J. G. Kemeny, T. E. Kurtz : True BASIC reference Manual, True BASIC inc, 1988