三角関数の計算

桂田 祐史

2005 年 7 月 9 日

ハイラー・ワナー [1] を使ったゼミで、sin, cos を計算する話が出て来た。

佐々木正敏著『ゆっくり考えよう!　高校総合学習の数学　— 教育現場からの提案』(講談社, 2003) でも取り上げられていた話題である。

例えば正五角形の作図法[1]からも[2]分かるように $18^\circ = \frac{\pi}{10}$ の sin, cos がルート $\sqrt{}$ を使って正確に表すことができる。一方、30° の sin, cos は有名なので、半角の公式から $15^\circ = \frac{\pi}{12}$ の sin, cos もルートを使って正確に表すことができる。後は加法定理を使って $18^\circ - 15^\circ = 3^\circ = \frac{\pi}{60}$ の sin, cos もルートを使って正確に表すことができる。したがって、ルートの計算さえできれば sin, cos の値が計算できる。

以上述べたストーリーにそって計算しよう。

$$\sin 18^\circ = \frac{\sqrt{5}-1}{4} = 0.30901\ 69943\ 74947\ 42410\ 22934\ 17182\ 81905\ 88601\ 54589\ 90288\cdots,$$

$$\cos 18^\circ = \frac{\sqrt{\frac{5+\sqrt{5}}{2}}}{2} = 0.95105\ 65162\ 95153\ 57211\ 64393\ 33379\ 38214\ 34056\ 98634\ 12575\cdots,$$

$$\sin 15^\circ = \frac{\sqrt{3}-1}{2\sqrt{2}} = 0.25881\ 90451\ 02520\ 76234\ 88988\ 37624\ 04832\ 83490\ 68901\ 31993\cdots,$$

$$\cos 15^\circ = \frac{\sqrt{3}+1}{2\sqrt{2}} = 0.96592\ 58262\ 89068\ 28674\ 97431\ 99728\ 89736\ 76339\ 04839\ 00840\cdots.$$

ここまではルートを用いて表示しても比較的シンプルであるが[3]、3° に対する三角関数は相当複雑になる[4]。

$$\sin 3^\circ = \frac{\sqrt{2}\left(\sqrt{3}+1\right)\left(\sqrt{5}-1\right) - 2\left(\sqrt{3}-1\right)\sqrt{5+\sqrt{5}}}{16}$$
$$= 0.05233\ 59562\ 42943\ 83272\ 21186\ 29609\ 07841\ 87310\ 18253\ 94016\cdots,$$
$$\cos 3^\circ = \frac{\sqrt{2}\left(\sqrt{3}-1\right)\left(\sqrt{5}-1\right) + 2\left(\sqrt{3}+1\right)\sqrt{5+\sqrt{5}}}{16}$$
$$= 0.99862\ 95347\ 54573\ 87378\ 44920\ 58439\ 43658\ 05909\ 52290\ 76778\cdots$$

[1] この種のことは筆者は中学生の頃に矢野健太郎の啓蒙書から学んだが、最近の数学少年少女はどうなのだろう…

[2] せっかちな人も多いので、$\alpha = 18^\circ$ とすると $5\alpha = \pi/2$ なので、$2\alpha = \pi/2 - 3\alpha$ だから…というのが教科書に多いみたいだけど。

[3] Mathematica では、例えば `Sin[15 Degree]` のように気軽に計算できる。

[4] 実は Mathematica でも、`Sin[3 Degree]` では計算してくれない。

ここまで来ると、$\sin 2°$, $\cos 2°$, $\sin 1°$, $\cos 1°$ が計算したくなるが、正多角形の定規とコンパスによる作図の理論から、これらはルート $\sqrt{\ }$ を使うだけでは表現できないことが (現代の我々にとっては明解に) 分かる。

もちろん、半角の公式を使って、半分の角度の sin, cos を求めることは簡単である。

$$\begin{aligned}
\sin 1.5° &= 0.02617\ 69483\ 07873\ 15261\ 06116\ 85554\ 11266\ 37933\ 91027\ 68010\cdots, \\
\cos 1.5° &= 0.99965\ 73249\ 75557\ 28003\ 67608\ 88367\ 67987\ 59498\ 75971\ 24107\cdots, \\
\sin 0.75° &= 0.01308\ 95955\ 71344\ 44019\ 02842\ 09702\ 85220\ 90185\ 60558\ 53053\cdots, \\
\cos 0.75° &= 0.99991\ 43275\ 74007\ 03224\ 89220\ 47454\ 88405\ 35790\ 69030\ 03766\cdots.
\end{aligned}$$

sin に注目すると、一段前の角のときの sin のほぼ半分になっていることが見て取れる。$x \fallingdotseq 0$ のとき $\sin x \fallingdotseq x$ であるから、これは自然なことである。これから、$\sin 1° \fallingdotseq \sin 0.75° \times \frac{4}{3}$ という近似が考えられる。試してみたのが次の結果である (正しい桁を赤字で示した)。

$$\begin{aligned}
\sin 1° &\fallingdotseq \sin 0.75° \times \frac{4}{3} \\
&= 0.01745\ 27940\ 95125\ 92025\ 37122\ 79603\ 80294\ 53580\cdots \\
\sin 1° &= 0.01745\ 24064\ 37283\ 51281\ 94189\ 78516\ 31619\ 24722\ 52720\ 30713\cdots
\end{aligned}$$

ところで、もしも円周率 π が精密に求まっているのならば、$\sin x \fallingdotseq x$ という素朴な近似で

$$\begin{aligned}
\sin 1° &= \sin \frac{\pi}{180} \fallingdotseq \frac{\pi}{180} \\
&= 0.01745\ 32925\ 19943\ 29576\ 92369\ 07684\ 88612\ 71344\ 28718\ 88541\cdots
\end{aligned}$$

が得られる。なおこの程度の精度を得るためには、円周率の近似値としては良く知られている $\pi \fallingdotseq 3.1416$ くらいで十分である。実際

$$\frac{3.1416}{180} = 0.01745 33333\cdots.$$

ちなみに $x = \pi/180$ とするとき、

$$\begin{aligned}
x - \frac{x^3}{3!} &= 0.01745\ 24064\ 23787\ 59447\ 12209\ 18992\ 75457\ 98837\ 65646\ 60350\cdots, \\
x - \frac{x^3}{3!} + \frac{x^5}{5!} &= 0.01745\ 24064\ 37283\ 61070\ 28534\ 69098\ 68449\ 39365\ 82463\ 57262\cdots \\
x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} &= 0.01745\ 24064\ 37283\ 51281\ 90048\ 52921\ 40839\ 85769\ 02441\ 53112\cdots, \\
x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} &= 0.01745\ 24064\ 37283\ 51281\ 94189\ 79663\ 13413\ 06454\ 31891\ 11881\cdots
\end{aligned}$$

A　計算の仕方

A.1　お説教

ハイラー・ワーナーの本には、昔の人達の苦心の計算結果がたくさん紹介されている。せっかくゼミで取り上げるのだから、彼らの計算はなるべくきちんと追跡すべきだと思う。彼らは

ほとんど例外なく、ひとかどの人物であって、しばしば相当な手間をかけた仕事であるわけだから、決して簡単ではないが、我々は整理された知識と色々な道具 (特にコンピューター) を持っているのだから、うまくやればかなり能率良く細かいところまで見えてくるものである。ゼミの準備に電卓すら叩いてこないのは何か間違えていると思う (教官の指導の失敗？)。

(特に時代が古いと数学的な知見を定理にまとめていないこともあって、そういう場合はエッセンスは計算例に詰まっているだけ、ということも稀でない。彼らの計算を真剣に検討しないで、一体どうやって理解できるというのだろう。)

A.2 電卓

ルート機能、メモリー機能のついている電卓は、うまく使えばかなり楽に計算できる。メモリー機能の使い方を知らない人が多いようなので、簡単に説明する。

[MC] あるいは [CM]	memory clear の略？メモリーの内容を 0 にする。
[MR] あるいは [RM]	memory recall の略？メモリーの内容を電卓の表示部分 (レジスターと呼ぶべき
[M+]	電卓に表示されている数をメモリーに加える。
[M−]	電卓に表示されている数をメモリーから引く。

例: 積和の計算 (本来的な使い方?)

例えば

品目	単価	個数
ちょこ	105	3
柿ピー	150	4
缶ジュース	120	5

というお買い物表から金額を計算するために、

電卓の操作

[MC] 105 [×] 3 [=] [M+] 150 [×] 4 [=] [M+] 120 [×] 5 [=] [M+] [MR]

とする[5]。C 風に書けば、

```
m = 0;
r = 105 * 3; m += r;
r = 150 * 4; m += r;
r = 120 * 5; m += r;
printf("%g\n", m);
```

[5]ほとんどすべての電卓で [M+], [M-] は計算途中のものの結果を出してからメモリーに加算 or から減算するので、[=] は省略できて、単に [MC] 105 [×] 3 [M+] 150 [×] 4 [M+] 120 [×] 5 [M+] [MR] としても求まるはず。

例: ルートの計算

ルート機能がないが、メモリー機能はある電卓で、与えられた正数 a のルートを計算するにはどうすればよいか？まあ、Newton 法でしょうね。つまり漸化式

$$x_{n+1} = x_n - \left(\frac{x_n^2 - a}{2x_n}\right) \quad \text{整理して} \quad x_{n+1} = \frac{1}{2}\left(x_n + \frac{a}{x_n}\right)$$

で数列 $\{x_n\}$ を生成すると (初期値 x_1 は例えば $x_1 = 1$ でよい)、x_n は急速に $\sqrt{a}$ に収束するので、適当なところで反復を止めればよい、ということである[6]。電卓では次の手順で操作すればよい。

(i) 適当な初期値 (1 でもよい) を入力し、[MC], [M+] でメモリーに記憶する。

(ii) a [÷] [MR] [+] [MR] [÷] 2 [=] で次の値が求まる。[MC], [M+] でメモリーに記憶する。十分な精度になるまでこの (ii) を繰り返す。

$a = 2$ の場合に、初期値を 1 として、8 桁の電卓で実行したところ、順に 1.5, 1.416666666, 1.414215686, 1.414213562 が得られた。 ■

A.3 Mathematica

何だかんだ言って、これが一番便利ですね。商品であってしかも値段がかなり高いのが玉に傷だけど…数学科は 30 ライセンス保持しているので有効利用してください。

度で与えた量の sin を 60 桁計算させたい場合、次のような関数定義をしておくと簡単である[7]。

```
mysin[x_]:=N[Sin[x Degree],60]
```

例えば以下のように使える。

```
In[57]:= mysin[1]

Out[57]= 0.017452406437283512819418978516316192472252720307139642683612 4

In[58]:= mysin[3/2]

Out[58]= 0.026176948307873152610611685554112663793391027680108638218786 3
```

注意: $\sin 1.5°$ を計算させるのに `mysin[1.5]` とするとうまく行かない (`1.5` は有効桁数の少ない入力と判断され、要求精度が低いと扱われてしまう)。小数を分数に直して `3/2` と入力する必要がある。

---

[6] 丸め誤差の影響を小さくするためには、上の整理していない方の漸化式を使うのがお奨めなのだが… [MR] [×] [MR] [−] a [÷] 2 [÷] [MR] [M−] としていって、(修正量が) 0 になったところで [MR] とすれば良い？

[7] `Degree` とは Mathematica で定義されている定数で、$\pi/180$ という値を持つ。

A.4 C 言語によるプログラム

C 言語によるプログラミングは習っているはずだし、フリーの処理系もあるので、本当はこれくらい自力で出来て欲しいところ。

```
/*
 * computesin.c
 */

#include <stdio.h>
#include <math.h>

double halfsin(double cosx)
{
  return sqrt((1-cosx)/2);
}

double halfcos(double cosx)
{
  return sqrt((1+cosx)/2);
}

int main()
{
  double root2, root3, root5;
  double sin18, cos18, sin15, cos15, sin3, cos3, sin1p5, cos1p5;
  double sin0p75, cos0p75;

  root2 = sqrt(2.0);
  root3 = sqrt(3.0);
  root5 = sqrt(5.0);
  sin18 = (root5-1)/4;
  cos18 = sqrt((5+root5)/2)/2;
  sin15 = (root3 - 1)/(2*root2);
  cos15 = (root3 + 1)/(2*root2);
  sin3 = sin18*cos15-cos18*sin15;
  cos3 = cos18*cos15+sin18*sin15;

  sin1p5 = halfsin(cos3);
  cos1p5 = halfcos(cos3);
  sin0p75 = halfsin(cos1p5);
  cos0p75 = halfcos(cos1p5);

  printf("sin18  =%20.15f\n", sin18);
  printf("cos18  =%20.15f\n", cos18);
  printf("sin15  =%20.15f\n", sin15);
  printf("cos15  =%20.15f\n", cos15);
  printf("sin3   =%20.15f\n", sin3);
  printf("cos3   =%20.15f\n", cos3);
  printf("sin1.5 =%20.15f\n", sin1p5);
  printf("cos1.5 =%20.15f\n", cos1p5);
  printf("sin0.75=%20.15f\n", sin0p75);
  printf("cos0.75=%20.15f\n", cos0p75);

  return 0;
}
```

```
oyabun% ./computesin
sin18  =   0.309016994374947
cos18  =   0.951056516295154
sin15  =   0.258819045102521
cos15  =   0.965925826289068
sin3   =   0.052335956242944
cos3   =   0.998629534754574
sin1.5 =   0.026176948307874
cos1.5 =   0.999657324975557
sin0.75=   0.013089595571345
cos0.75=   0.999914327574007
oyabun%
```

Solaris では long double の精度が高いので、それを利用すると

```
oyabun% ./computesin-long
sin18  =   0.30901699437494742410229341718281908
cos18  =   0.95105651629515357211643933337938214
sin15  =   0.25881904510252076234889883762404831
cos15  =   0.96592582628906828674974319972889726
sin3   =   0.05233595624294383272211862960907841
cos3   =   0.99862953475457387378449205843943650
sin1.5 =   0.02617694830787315261061168555411346
cos1.5 =   0.99965732497555728003676088836767982
sin0.75=   0.01308959557134444019028420970285323
cos0.75=   0.99991432757400703224892204745488401
oyabun%
```

A.5 BASIC

10 年位前[8]までは、パソコンには BASIC 言語のインタープリターが標準で搭載されているのが普通で、例えば高等学校の数学の教科書にも BASIC のプログラムが掲載されていた。時は流れ、今では BASIC の処理系をインストールしてあるパソコンを探す方が難しくなってしまったが、その気になれば簡単に入手&インストールできる。

私のお奨めは二つある。

UBASIC 数論の研究者である木田祐司氏によって作られたソフトで、`http://www.rkmath.rikkyo.ac.jp/~kida/ubasic.htm` から入手できる。UBASIC が何であるかの説明は作者のホームページよりも、「UBASIC について by 愛知教育大学 数学教室 飯島康之」[9] の方が分かりやすいかも知れない。

[8] これを書いているのは 2005 年である。

[9] `http://www.auemath.aichi-edu.ac.jp/teacher/iijima/ubasic/`

十進 BASIC for Windows 95 作者の白石和夫氏@文京大学[10]によると、数学教育での利用を目的として、JIS Full BASIC を Windows 環境で実現することを目標に作られたそうである。数の精度を 10 進法で 1000 桁に変更することが簡単にできるので、この手の数値計算には便利に使える。必要なファイルは、http://hp.vector.co.jp/authors/VA008683/ から入手できる。インストールは非常に簡単で[11]、使うのも簡単である (と私には思われる)。なお、Linux バージョンも存在し、Knoppix Math に収録されている。配布されているファイルの中にチュートリアル (TUTORIAL.PDF) がある。ちなみに講談社ブルーバックスに、この BASIC を使ったプログラミングの解説書 (木村 [2]) がある。

十進 BASIC プログラム例

次のプログラムは基本的に C プログラムと同じことをしている。

computesin.bas

```
DEF halfcos(COSX)=SQR((1+COSX)/2)
DEF halfsin(COSX)=SQR((1-COSX)/2)
LET  SIN18=(SQR(5)-1)/4
LET  cos18=(SQR((5+SQR(5))/2))/2
LET  SIN15=(SQR(3)-1)/(2*SQR(2))
LET  COS15=(SQR(3)+1)/(2*SQR(2))
LET  SIN3=COS15*SIN18-SIN15*COS18
LET  COS3=COS15*COS18+SIN15*SIN18
LET  SIN1p5=halfsin(COS3)
LET  COS1p5=halfcos(COS3)
LET  SIN0p75=halfsin(COS1p5)
LET  COS0p75=halfcos(COS1p5)
LET  SIN1=SIN0p75*4/3
PRINT "sin(18)=";SIN18
PRINT "cos(18)=";COS18
PRINT "sin(15)=";SIN15
PRINT "cos(15)=";COS15
PRINT "sin(3)=";SIN3
PRINT "cos(3)=";COS3
PRINT "sin(1)=";SIN1
END
```

[オプション] → [数値] で「10 進 1000 桁」を選択して実行すると、以下の結果が得られる。

```
sin(18)= .309016994374947424102293417182819058860154589902881431067724311352630231409451224853603602094
cos(18)= .951056516295153572116439333379382143405698634125750222447305644430153170085193501718792810970
sin(15)= .258819045102520762348898837624048328349068901319930513814003207315056974748801996922367974694
cos(15)= .965925826289068286749743199728897367633904839008404550402343076310423213979855516347561741858
sin(3)= .052335956242943832722118629609078418731018253940164920483509381599857104641754546864464598811 8
cos(3)= .998629534754573873784492058439436580590952290767785532441441254831648973733478318635332028233 2
sin(1)= .017452794095125920253712279603802945358080744707373592550453043105768709610006741629578236511 6
```

[10] http://homepage3.nifty.com/ShiraishiKazuo/

[11] ダウンロードしたファイル (自己展開形式) を実行して OK していくだけである (展開したフォルダー中の BASIC.EXE へのショートカットをデスクトップに作ってくれる)。

独り言 (何だか長くなっている)

数学村のおじさん達には UBASIC が圧倒的に有名であると思われるが、UBASIC のユーザー・インターフェイスは、太古の Microsoft BASIC[12] のそれを真似していて[13]、それを知らない若い人に取っては結構難しいかもしれない (杞憂かしらん…でも最近はコンピューターを囲んでものを習おうという空気が希薄なので、何でもないようで結構障害になるかなと心配になります)。私を含めておじさん達には UBASIC 使うのに何の困難もないのだけど…一方、十進 BASIC は有理数計算モード、複素数計算モードも備えていて、中学高校の教室ではなかなか便利ではないだろうか (自分が教師になったら嬉々として使いそうな気がする)。かつて一世を風靡した N88 BASIC(86) も今では事実上世の中から消えてなくなってしまったわけだが、強力な代役が登場したように感じる。

色々WWW ページを見ていて、高校の教員をしている人が、パソコンに BASIC が載っていないので教科書のプログラムが試せない、十進 BASIC が普及しているが、デフォールトが N88 BASIC 互換でないので…と悩んでいるのを見つけた。そうねぇ…コンピューターがらみの授業をするとき、実際に触ってもらうには何かを選ばなければいけないけれど、何を選ぶかは悩ましいことが多い、というのは共通することなのだろう。私の悟り (開き直り？) は、

そのときの目的に最適だと信じるものを堂々と使って見せる使わせる
(すべての場合に良い点取れるものなんかないさ)

というものである。というわけで「なかなかいいよ十進 BASIC」。

やや脇道にそれるようだが、最近高等学校の数学の教科書を読んでいて、JavaScript で書かれたプログラムが載っているのを見て仰天した。うーん… (一体誰の趣味だと執筆者一覧を見てみたり)

参考文献

[1] E. ハイラー, G. ワナー, 解析教程 上, 下, シュプリンガーフェアラーク東京 (1997).

[2] 木村 良夫, パソコンを遊ぶ 簡単プログラミング CD-ROM 付　コンピュータを自由に操る「十進 BASIC」入門, 講談社ブルーバックス B-1398 (2003).

[3] 佐々木 正敏, ゆっくり考えよう! 高校総合学習の数学— 教育現場からの提案, 講談社ブルーバックス (2003).

[12] 若い人のためにウンチクを書いておくと、Microsoft BASIC は Bill Gates の出世作である (色々なパソコンメーカーに売り付けた)。Visual BASIC というずっと良いものが出てきたので、Microsoft はそれを買い取って乗り換えた。今では、Microsoft の BASIC とは Visual BASIC のこととなってしまった。お仕事がらみで否応なく Visual BASIC プログラマーになる人はとても多い。

[13] というよりも、日本で一番ユーザーの多かった N88 日本語 BASIC (86) (NEC PC-9801 シリーズに搭載されていた BASIC) を真似したという方が正確か？