《急ぐ人のための》 *Mathematica* Graphics 入門

渡部 善隆 *

1 はじめに

この解説記事は、次の方を対象にしています。

1. 研究室に UNIX ワークステーションがあり、 LAN 経由で九州大学大型計算機センターに telnet、ftp できる。
2. でなければ、センター二階オープン室に出向き、センターのワークステーションを扱うだけの意欲がある。
3. 論文や OHP に簡単な graphics を手っ取り早く挿入したいと思っている。
4. とくに、 LaTeX に組み込む PostScript file を作りたい。
5. 研究室に PostScript printer がある。
6. でなければ、センター二階オープン室に graphics の出力を取りに来る意欲がある。カラー出力も欲しい。
7. 具体的な関数の形がわかっていると大変便利です。

これらの人に、センターで公開している数式処理システム *Mathematica* を使った graphics の “とりあえずの” 使い方を例題一辺倒で解説します。この解説を眺めて、使ってみようかなという気になった方は、最後にあげた参考文献を書店でお買い求め下さい。

1.1 *Mathematica* とは

Mathematica は Stephen Wolfram さんにより開発された汎用数学処理システムです。高度な計算機能、数式処理機能、 2 次元、 3 次元 graphics 機能を有しています。数式は Fortran, C, TeX ソースとして出力可能で、 Graphics 出力は PostScript コードで出力されます。

graphics は X Window システムと Tektronix に対応しています。現在のバージョンは 2.2 です。

1.2 どこで使うのか

センター内の計算機室に設置されたアプリケーション・ライブラリ専用の Sun ワークステーションを LAN 経由で利用します。利用者が直接このワークステーションを触ることはできません。仕様は以下の通りです。

WS 名	qapls
ドメイン名	qapls.cc.kyushu-u.ac.jp
IP アドレス	133.5.8.40
機種	S–4/10
OS	SunOS 4.1.3

* 九州大学大型計算機センター・研究開発部

なお、 OS やワークステーションは最近の計算機環境の激変のおり、すぐに変更になる場合があります。変更の際はセンターニュースで直ちにお知らせします。

qapls は「くぁぷるす」と読みます。意味は「九大センターアプリケーション・サーバー」くらいでしょうか[1]。

1.3 使用時間は？

qapls はベクトルプロセッサ、および汎用計算機と独立に運用されています。システム障害や停電など特別な場合がないかぎり、 24 時間 /365 日運用です。深夜に仕事をするのが好きな人には嬉しいニュースです。

1.4 負担金は？

ただでは使えません。センターのワークステーション負担金が適用されます。 1 分の接続時間あたり 3 円いただきます。つまり、 1 時間つないで 180 円です。なお CPU 課金はありません。

1.5 セッション制限は？

ネットワークライセンスの契約上、 telnet で *Mathematica* を利用できる人数は 同時に 2 人 までです。それ以上の利用はできません。従って、一人で *Mathematica* セッションを 2 つ開設すると、独占状態になって他の利用ができなくなります。使用する場合は、一つだけに辛抱願います。

2 とりあえず起動

なにはともあれ、 qapls へ login して *Mathematica* を起動しなくては話が進みません。

2.1 qapls への利用登録

qapls に入って *Mathematica* [2]を起動するには、次の step を越えてください。

1. センターに利用登録する。
   この広報を見ている方なら利用者番号をお持ちでしょうから、これは大丈夫でしょう。大丈夫でない方は、センターの共同利用掛まで御相談下さい。

2. UXP に利用申請をする。
   UXP は汎用機の UNIX OS の名前です[3]。 MSP から`SINSEI` コマンドで利用申請をします。メニュー形式なので、使い方はすぐにわかります。申請から UXP が利用できるようになるまでに、大体 1, 2 時間かかります。

   UXP や MSP の意味を解さない方は、これまた共同利用掛に連絡して、『利用の手引・基本編』を入手下さい。詳しい説明があります。

3. qapls に利用登録する。
   UXP はホスト名を kyu-cc といいます。 kyu-cc に login した後、`touroku` コマンドをたたきます。

[1] 誰が命名したかは教えません。

[2] マニュアルを見る限り、全てイタリックで *Mathematica* と書いてあるので、ここでも真似しています。

[3] ベクトルプロセッサ VP2600/10 にも UXP があります。

```
kyu-cc% touroku qapls  ↵              <--- qapls に利用登録
OK, User a79999a added in qapls.       <--- 登録完了
kyu-cc%
```

これで qapls が使える状態になりました。

2.2 qapls 利用の注意事項

qapls の ID とパスワードは UXP と同じです。 qapls のホームディレクトリは、利用者を a79999a さんとすると、

/export/home/a79999a

に作成されます。この場合、/export/home/a79999a 下は、利用者スペースとして自由にファイルの作成・保存ができます。しかしながら、ディスクにも限界があるので、システム管理者で周期的にファイルの整理を行います。従って、長期にわたって大量のファイルを qapls に保存することはできません。

計算結果はさっさと手元のワークステーションに転送下さい。

2.3 利用者の環境の例

この原稿では、具体的に利用者の環境を以下のように決めます。

WS 名	user
ドメイン名	user.cc.kyushu-u.ac.jp
IP アドレス	133.5.7.80
	X Window
利用者番号	a79999a

2.4 qapls への telnet

その前に...

2.4.1 xhost コマンド

まず、研究室のワークステーション user から xhost コマンドを実行します。「コマンドが見つかりません」と言われる場合がありますが、それは path が通っていないためです。ワークステーションの管理者を捕まえ xhost コマンドの在処を聞いて下さい。

/usr/openwin/bin/xhost や /home/X11R5/bin/xhost など、それっぽいとこにあるはずです。

```
user% xhost 133.5.8.40  ↵                    <--- xhost コマンド
133.5.8.40 being added to access control list
user%
```

133.5.8.40 は、 qapls の IP アドレスです。このまま打ち込んで下さい。

2.4.2 telnet コマンド

qapls への login は、telnet コマンドにより LAN 経由で行います。パスワードは UXP と同じです。

```
user% telnet 133.5.8.40  ↵                              <--- qapls への telnet
Trying 133.5.8.40 ...
Connected to 133.5.8.40.
Escape character is '^]'.


SunOS UNIX (qapls)

login: a79999a  ↵                       <---- 利用者番号
Password:       ↵                       <---- パスワード
Last login: Thu Mar  9 10:00:12 from kyu-cc
SunOS Release 4.1.3-JLE1.1.3 (GENERIC) #1: Tue Aug 4 19:22:57 JST 1992
***** statistics and charges ( a79999a  )               *****
**      total size of reserved files    =     7861 KB      **
**      total charge                    =    62011 yen     **
*************************************************************

Terminal Type (default:vt100) :  ↵
qapls%
```

これで login に成功しました。

2.5 setenv コマンド

さて、 qapls に login できても、まだ *Mathematica* の graphics 機能は使用できません。次に、 qapls から setenv コマンドを打ちます。

```
qapls% setenv DISPLAY 133.5.7.80:0.0  ↵      <--- setenv コマンド
qapls%
```

下線部の "133.5.7.80" は、お使いのワークステーション、または X 端末固有の IP アドレスを入力します。つまり、自分が今端末として使用している IP アドレスです[4]。最後の ":0.0" も忘れずに入力して下さい。

2.6 三つのコマンド

qapls でサポートしている *Mathematica* 関連のコマンドは、

```
math
mathematica
mathbook
```

の三つです。

[4] IP アドレスがわからないときは、自身のワークステーションに telnet するとわかります。

2.6.1 math コマンド

math コマンドは、ラインモードで *Mathematica* を使用するコマンドです。 *Mathematica* は setenv コマンドを打たなくても起動しますが、 graphics 表示はできません[5]。 graphics を表示するためには math コマンドの前に setenv コマンドが必要です。

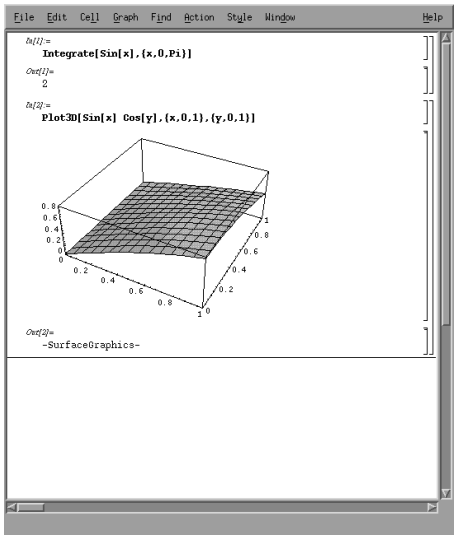
```
qapls% math
Mathematica 2.2 for SPARC
Copyright 1988-94 Wolfram Research, Inc.
 -- Motif graphics initialized --

In[1]:= Integrate[Sin[x],{x,0,1}]

Out[1]= 1 - Cos[1]

In[2]:= N[%,10]

Out[2]= 0.4596976941

In[3]:= D[ArcTan[x],x]

           1
Out[3]= ------
             2
        1 + x

In[4]:= Quit
qapls%
```

ラインモードでの *Mathematica* 処理

2.6.2 mathemactia コマンド

対して mathematica および mathbook コマンドは、setenv コマンドをあらかじめ打たないと起動しません。mathematica とタイプすると、ノートブック・フロントエンドモード[6]で *Mathematica* が起動します。こちらの方が、マウス操作が入るなど、ラインモードに比べ便利です。

この解説では、ラインモードでの処理を画面の例としています。もちろん全く同じ操作がノートブックモードでも可能です。



ノートブック・フロントエンドモード

基本的な操作は以下の通りです。

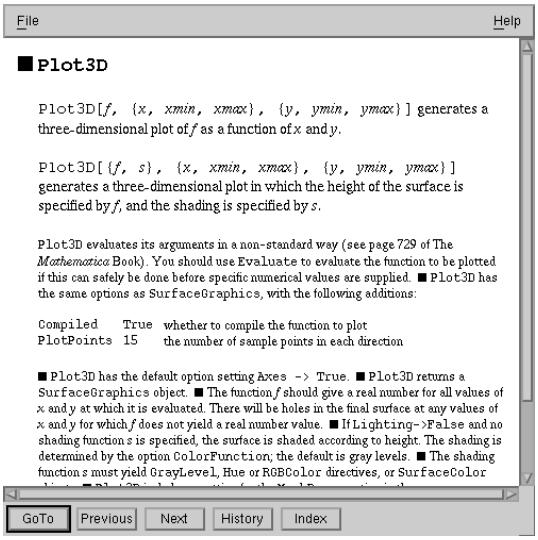
[5]一応 graphics は出ますが、おそまつなものです。

[6]なぜ「ノートブック」と言うのか私は知りません。 *Mathematica* のマニュアルに書いてあったのでそのまま書いているだけです。大した意味はないと思います。

	ノートブックモード	ラインモード
起動	mathematica とタイプ	math とタイプ
入力の終了	Shift + Return	Return
複数行 (改行)	Return	Return
終了	Menu から Quit を選択	Quit と入力

2.6.3 mathbook コマンド

mathbook とタイプすると、 *Mathematica* に組み込まれた全 object がオンラインで検索できます。終了は左上の “File” をマウスでクリックして、 “Quit” を選びます。



mathbook の利用風景

すべてメニュー形式ですので、適当にいじってください。

3 二次元のリストをプロットする

3.1 データの用意

ここでは Fortran や C で数値計算を行なった結果のリストを図にしてみましょう。データは x 座標，y 座標のペア (x, y) のデータが一行づつ入っているファイル ex1.data とします。

```
0.0     20.1
0.1     18.5
0.2     17.3          <--- ex1.data の中身
0.3      7.3
0.4      7.7
0.5      7.8
0.6      7.9
0.7     18.3
0.8     20.5
0.9      4.5
1.0      0.0
```

また、特に改行もせず “ずらずら” とデータが並んでいるファイル ex2.data も考えます。

```
0.0     20.1    0.1     18.5    0.2     17.3    0.3      7.3
0.4      7.7    0.5      7.8    0.6      7.9    0.7     18.3
0.8     20.5    0.9      4.5    1.0      0.0
                                              <--- ex2.data の中身
```

まず、 plot したいデータファイルを qapls に ftp で転送します。 ftp の方法は、多分皆さんご存知と思うので省略します。わからない場合は、共同利用掛に出向いて『KITE 利用の手引』をもらい、お読み下さい。 *Mathematica* を起動した後、ReadList によってデータを読み込みます。

```
In[1]:= ReadList["ex1.data",Number,RecordLists -> True]  ↵

Out[1]= {{0., 20.1}, {0.1, 18.5}, {0.2, 17.3}, {0.3, 7.3}, {0.4, 7.7},

>    {0.5, 7.8}, {0.6, 7.9}, {0.7, 18.3}, {0.8, 20.5}, {0.9, 4.5}, {1., 0.},

>    {0.2, 17.3}}                                  <--- List の読み込み
```

“Number” は入力が数値であることを、また、 “RecordLists -> True” は、各行ごとにリストにすることの指定です。これによって、一行を (x, y) のペアとしたリストを作成します。

さて、ex2.data の様にデータが並んでいる場合は、「二つのデータをペアにします」と教えてやります。

```
In[2]:= ReadList["ex2.data",{Number,Number}]  ↵

Out[2]= {{0., 20.1}, {0.1, 18.5}, {0.2, 17.3}, {0.3, 7.3}, {0.4, 7.7},

>    {0.5, 7.8}, {0.6, 7.9}, {0.7, 18.3}, {0.8, 20.5}, {0.9, 4.5}, {1., 0.},

>    {0.2, 17.3}}                                  <--- List の読み込み
```

出力は先ほどと同じです。 “{Number,Number}” は、連続したペアを独立したリストにするオプションです。

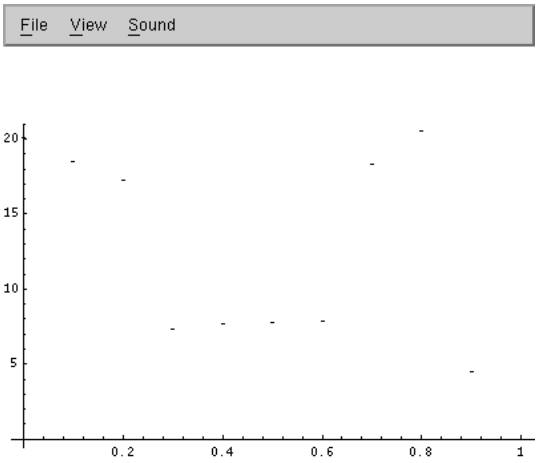
「リスト」は *Mathematica* では極めて重要な概念です。ここではあまり深く立ち入らないで、「x 座標に対する y の値のリストができた」ということで、先に進みます。

3.2 ListPlot

読み込んだリストは ListPlot で表示します。

```
In[3]:= t=%;  ↵                          <--- List を変数 t で定義
                                               ; はリスト出力を抑止する
In[4]:= ListPlot[t]  ↵                   <--- ListPlot

Out[4]= -Graphics-
```

このように入力すると、 graphics window が立ち上がります。勝手に画面が登場する場合もありますが、普通はマウスを使って graphics を表示する場所を決めます。



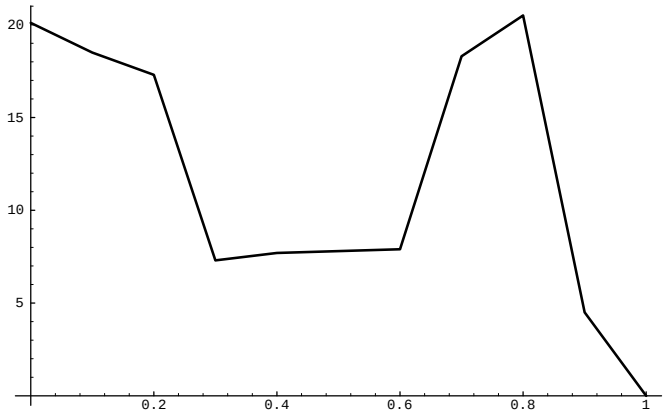
ListPlot 画面

graphics 画面を消す場合は、マウスで左上の “File” の “Quit” をクリックします。

3.3 線をつなぐ

ListPlot の画面を見ておわかりのように、これは単に点のプロットだけです。各点を直線でつなぐには、次のようにします。

```
In[5]:= ListPlot[t,PlotJoined -> True]   ↵          <--- 折れ線で点を結ぶ

Out[5]= -Graphics-
```

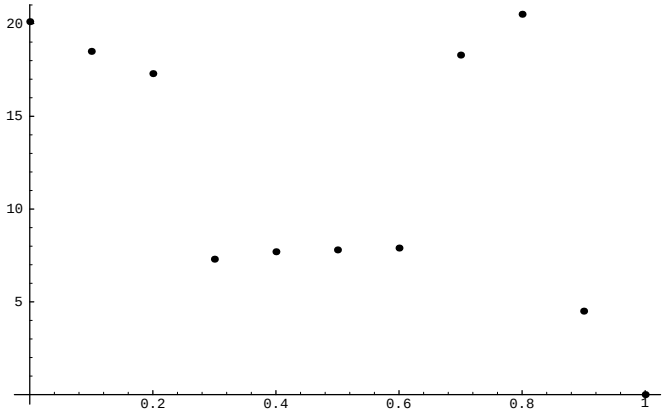


各点を直線でつなぐ

3.4 点の半径を大きくする

plot した点をわかりやすくする場合は、たとえば次のようにします。

```
In[6]:= ListPlot[t,Prolog -> AbsolutePointSize[5]]   ↵

Out[6]= -Graphics-
```



少し大きくする

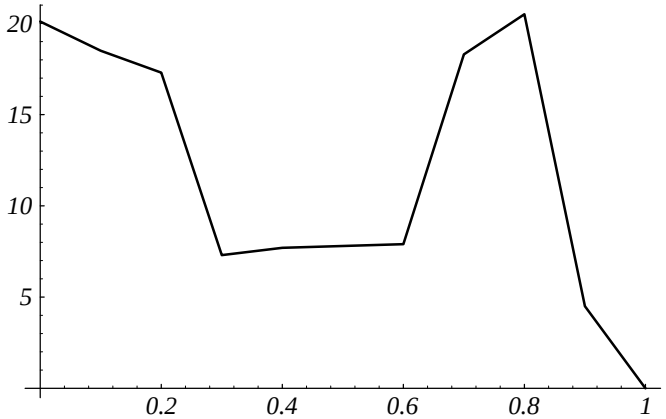
3.5 font を変更する

フォントを大きくしたり書式を変更する時は、DefaultFont オプションで書体と大きさを指定します。ノートブック・フロントエンドモードの場合は、メニューからでも変更できます。

default は “Courier” の “10.0” になっています。フォントはサポートするデバイスによっては使えない場合もあります。

例えばすべての graphics のフォントを “Times-Italic” にしたい場合は *Mathematica* 起動直後に、$DefaultFont を打ち込み、 default 書体を変更します。

```
In[1]:= $DefaultFont  ↵                          <--- 現在のフォントを確認

Out[1]= Courier, 10.

In[2]:= $DefaultFont = {"Times-Italic",18}  ↵     <--- フォントの変更

Out[2]= Times-Italic, 18
```



フォント変更後

Mathematica のマニュアルによれば、普通以下のフォントは使えるようです。いろいろ設定されて、お試し下さい。

Courier	Courier-Bold	Courier-Oblique	Courier-BoldOblique
Helvetica	Helvetica-Bold	Helvetica-Oblique	Helvetica-BoldOblique
Times	Times-Bold	Times-Italic	Times-BoldItalic

指定できるフォント例

4 リストを使った三次元グラフ

4.1 注意事項

以降の章でみるように、 *Mathematica* の Graphics は、関数の値が explicit に分かる場合、圧倒的な威力を発揮します。しかし、与えられたリストに対して 3 次元の graphics を描こうとすると、ずいぶん限られたことしかできません。

リストの 3 次元グラフは `ListPlot3D` で作成します。引数は "高さ" の行列が必要です。とりあえず行列の名前を z とすると、 z は $m \times n$ の配列にします。

つまり、 $z = \{z_{ij}\},\ 1 \leq i \leq m,\ 1 \leq j \leq n$ に対し、 z_{ij} は矩形領域を y 方向に $m-1$, x 方向に $n-1$ 等分したときの node i, j の「高さ」を表します。

従って、 graphman の様に $x,\ y$ 座標を与え、それに対する $z = f(x, y)$ の値を読み込ませるといった作図は出来ません。

以上の様に、データを作る場合は注意が必要となります。

4.2 データの用意

それでは、例として以下のデータを用意します。

```
 0.00  0.84  0.91  0.14 -0.76 -0.96 -0.28  0.66  0.99  0.41 -0.54 -1.00 -0.54  0.42  0.99
 0.75  0.96  0.30 -0.64 -0.99 -0.43  0.53  1.00  0.55 -0.40 -0.99 -0.66  0.27  0.96  0.76
 0.79  0.94  0.23 -0.69 -0.98 -0.37  0.59  1.00  0.49 -0.47 -1.00 -0.61  0.34  0.97  0.72
 0.14  0.91  0.84  0.00 -0.84 -0.91 -0.14  0.76  0.96  0.28 -0.66 -0.99 -0.41  0.54  1.00
-0.69  0.24  0.95  0.78 -0.10 -0.89 -0.86 -0.04  0.82  0.93  0.18 -0.73 -0.97 -0.32  0.63
-0.82  0.04  0.86  0.89  0.10 -0.78 -0.95 -0.24  0.69  0.98  0.37 -0.58 -1.00 -0.50  0.46
-0.28  0.66  0.99  0.41 -0.55 -1.00 -0.53  0.42  0.99  0.65 -0.29 -0.96 -0.75  0.15  0.91
 0.61  1.00  0.47 -0.49 -1.00 -0.59  0.37  0.98  0.69 -0.23 -0.94 -0.79  0.09  0.89  0.87
 0.84  0.91  0.15 -0.75 -0.96 -0.29  0.65  0.99  0.42 -0.54 -1.00 -0.55  0.41  0.99  0.66
 0.40  0.99  0.67 -0.27 -0.96 -0.77  0.13  0.90  0.85  0.01 -0.83 -0.91 -0.15  0.75  0.96
-0.52  0.44  0.99  0.63 -0.31 -0.97 -0.74  0.17  0.92  0.82 -0.03 -0.86 -0.90 -0.11  0.78
-0.84  0.00  0.84  0.91  0.14 -0.76 -0.96 -0.28  0.66  0.99  0.41 -0.54 -1.00 -0.54  0.42
-0.51  0.45  0.99  0.63 -0.32 -0.97 -0.73  0.18  0.92  0.82 -0.04 -0.86 -0.89 -0.10  0.78
 0.41  0.99  0.66 -0.27 -0.96 -0.76  0.14  0.91  0.84  0.00 -0.84 -0.91 -0.15  0.75  0.96
 0.84  0.91  0.15 -0.75 -0.96 -0.29  0.65  0.99  0.42 -0.54 -1.00 -0.54  0.41  0.99  0.66
```

このデータは、正方領域を $x,\ y$ の各方向 14 等分した node 上の値 (のみ) を格納したものです。データは i 行について 15 個ずつ x 方向の値が入っています。リストの読み込みは 2 次元と同様 `ReadList` で行ないます。

```
In[1]:= ReadList["ex3.data",Number,RecordLists -> True]

Out[1]= {{0., 0.84, 0.91, 0.14, -0.76, -0.96, -0.28, 0.66, 0.99, 0.41, -0.54,

>     -1., -0.54, 0.42, 0.99}, {0.75, 0.96, 0.3, -0.64, -0.99, -0.43, 0.53,

>     1., 0.55, -0.4, -0.99, -0.66, 0.27, 0.96, 0.76},

                    :                                    <---- List の読み込み
```

"RecordLists -> True" は、ファイルの各行ごとの数値を独立のリスト (つまり列) にすることを指定しています。もしデータに行ごとの改行がない場合は、「15 個ごとに列を作りますよ」という指令を与える必要があります。この場合は、

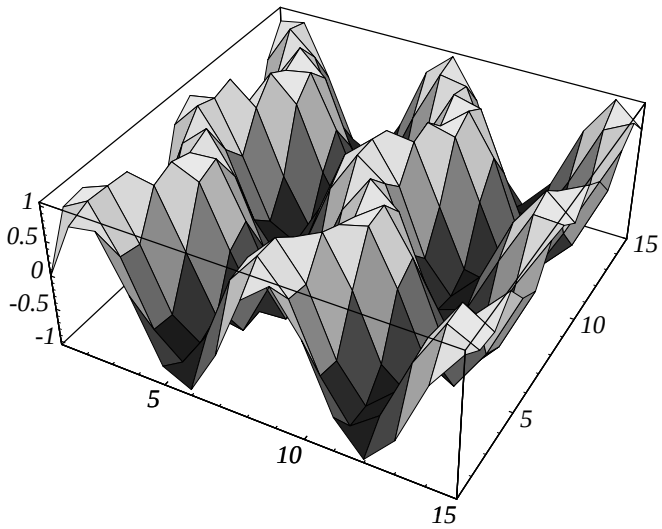
```
ReadList["ex3.data",Table[Number,{15}]]
```

と書きます。15 はデータに応じて変えて下さい。

4.3 ListPlot3D

読み込んだリストの表示は ListPlot3D で行ないます。

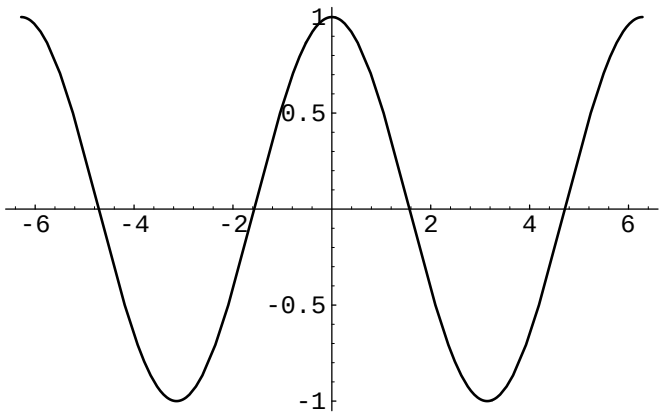
```
In[2]:= m=%;                          <--- リストを m で定義
                                           ; をつけ、表示を抑止
In[3]:= ListPlot3D[m]                 <--- 3 次元グラフ表示

Out[3]= -SurfaceGraphics-

In[4]:= ListPlot3D[m,Lighting -> False]
                                      <--- Grayscale でプリントする場合
Out[4]= -SurfaceGraphics-
```



ListPlot3D の出力結果

5　$y = f(x)$ のプロット集

関数の形が決まっている場合、簡単に関数値をプロットすることができます。まずは、$y = f(x)$ の形で与えられる関数の二次元 graphics の例題を参考文献よりいくつか集めてみました。



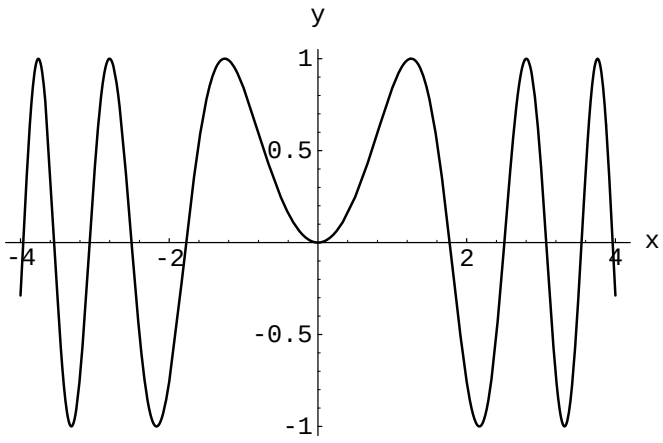
$y = \cos x$ を $-2\pi \leq x \leq 2\pi$ の範囲で作図。

```
In[1]:= Plot[Cos[x],{x,-2Pi,2Pi}]
```

Mathematica は、ぱっと思い浮かぶ数学関数ならば大体サポートしています。関数名は、一般に使われている英語名が (最初を大文字にして) 使われています。初等超越関数を例にとると

`Exp[z]` , `Cot[z]` , `ArcSin[z]` , `Sinh[z]` , `ArcTanh[z]`

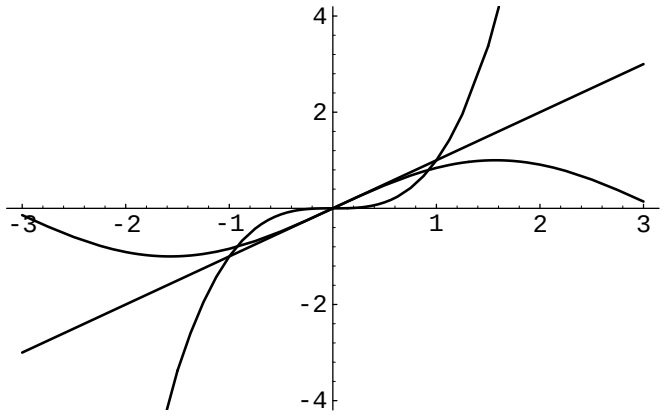
など , すぐに分かる関数として定義されています。全ての数学関数の機能は mathbook を用いオンラインで参照可能です。



$y = \sin x^2$ を $-4 \leq x \leq 4$ の範囲で作図。 x 軸 , y 軸にラベルをつける。

```
In[1]:= Plot[Sin[x^2],{x,-4,4},AxesLabel -> {"x","y"}];
```

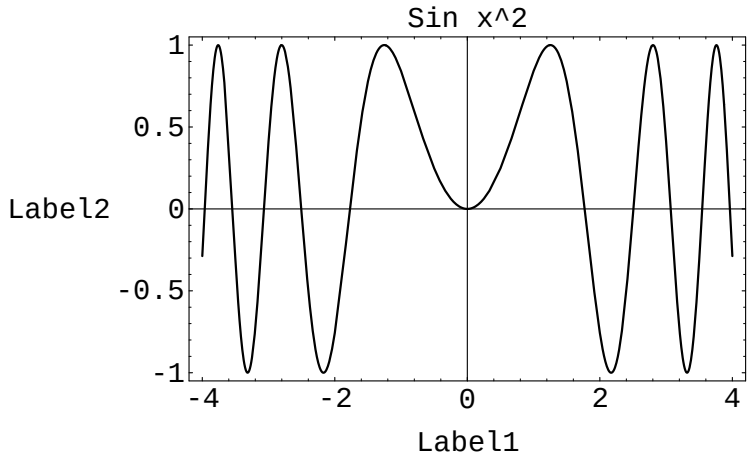
`AxesLabel -> {"x","y"}` の `x` , `y` は何でも構いません。



$y = x,\ y = \sin x,\ y = x^3$ のグラフを -3 から 3 まで描く。

```
In[1]:= Plot[{x,Sin[x],x^3},{x,-3,3}];
```

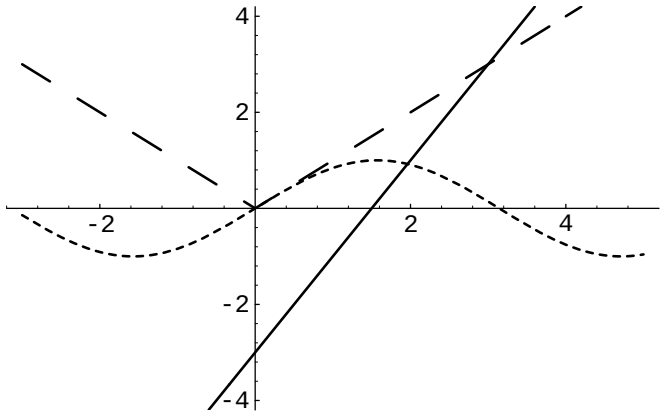
このように、異なる関数の同時作図も可能です。



$\sin x^2$ を -4 から 4 まで描き、枠とラベルを加える。

```
In[1]:= Plot[Sin[x^2],{x,-4,4}, Frame -> True,
        PlotLabel -> "Sin x^2", RotateLabel -> False,
        FrameLabel -> {"Label1","Label2"}];
```

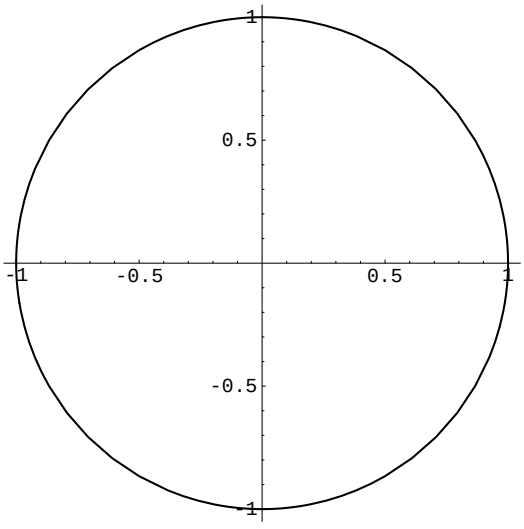
`Frame -> True` で枠を描きます。`FrameLabel` は枠の縁のラベルを与えます。`RotateLabel -> False` はラベルのテキストを回転しないことを指定します。これは実際やってみればすぐに意味がわかります。



$y = 2x - 3$, $y = |x|$, $y = \sin x$ のグラフを -3 から 5 の範囲で描く。

```
In[1]:= Plot[{2x-3,Abs[x],Sin[x]},{x,-3,5},
         PlotStyle -> { {Dashing[{1}]},
        {Dashing[{0.05,0.05}]},
        {Dashing[{0.01,0.01}]}}];
```

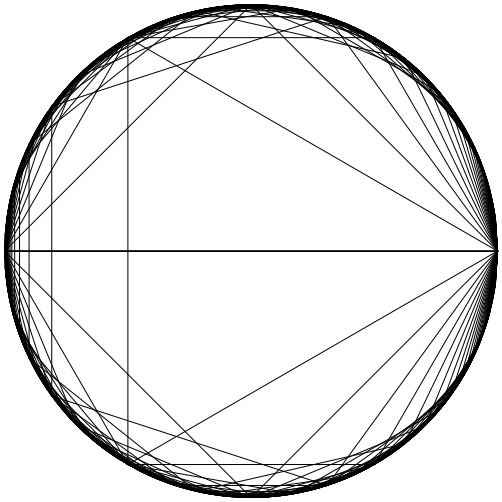
Dashing は、破線とその長さを指定します。



$0 \leq t \leq 2\pi$ の媒介変数 t で表された曲線 $(x, y) = (\sin t,\ \cos t)$ (要するに円) を描く。

```
In[1]:= ParametricPlot[{Sin[t],Cos[t]}, {t,0,2Pi},
         AspectRatio -> Automatic];
```

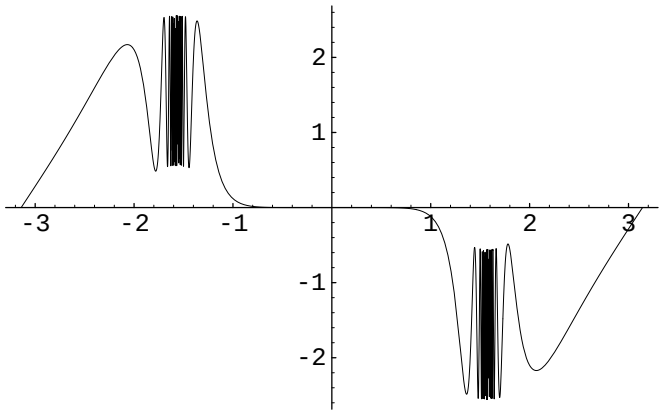
AspectRatio -> Automatic は、x 方向と y 方向の単位長さを等しくするオプションです。この場合、これを指定しないと円がつぶれて表示されます。



任意の正 n 角形を生成する関数 NGON を定義する。更に、正 50 角形までをまとめて作図する。

```
In[1]:= NGON[n_]:= Line[ Table[{Cos[t],Sin[t]},{t,0,2Pi,2Pi/n}]]
In[2]:= Show[Graphics[Table[{AbsoluteThickness[0.01],
        NGON[m]},{m,1,50}]], AspectRatio -> Automatic];
```

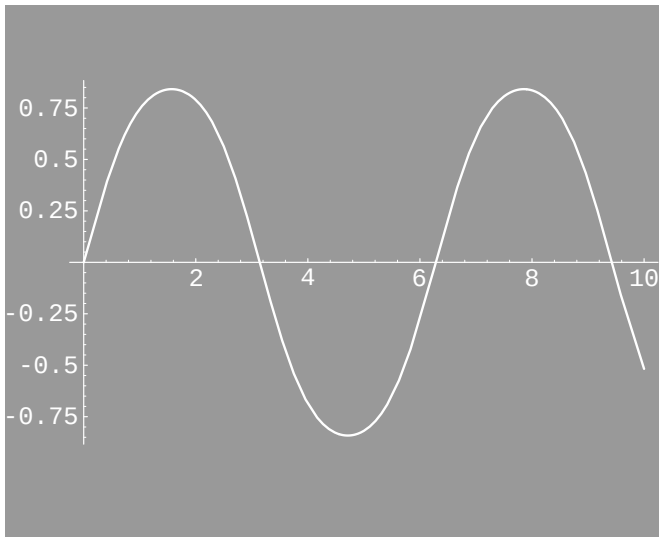
機能については、面倒なのでいちいち説明しません。AbsoluteThickness[0.01] は、線を細くするオプションです。



$y = \sin(\tan x) - \tan(\sin x)$ を $-\pi$ から π で描く。

```
In[1]:= Plot[Sin[Tan[x]] - Tan[Sin[x]],{x,-Pi,Pi},
        PlotPoints -> 100, PlotDivision -> 30,
        PlotStyle -> Thickness[0.0005]];
```

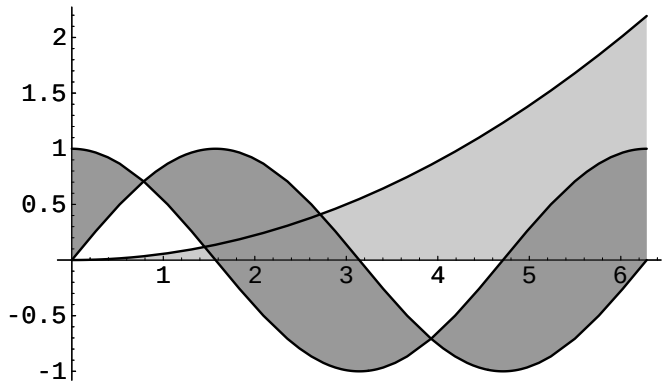
PlotPoints は作図をする際に評価する点の個数です。省略値は Plot の場合 25 です。評価する点を増やすと、滑らかなグラフが描けます。



$y = \sin(\sin x)$ を 0 から 10 まで描く。背景をグレーに、ラインは白線に変更する。

```
In[1]:= Plot[Sin[Sin[x]],{x,0,10},
        Background -> GrayLevel[0.6],
        DefaultColor -> GrayLevel[1]];
```

GrayLevel は、0 が黒、1 が白で、間の色は $[0, 1]$ の値で指定します。



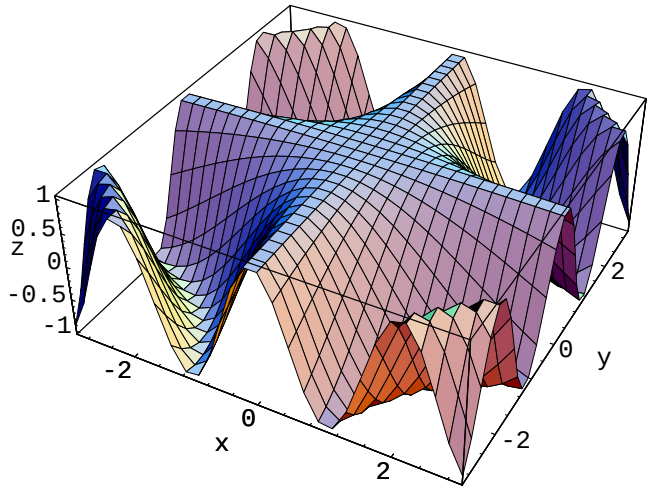
$y = x^2/18$, $y = \cos x$, $y = \sin x$ を 0 から 2π まで描き、曲線と軸で囲まれた領域を塗りつぶす。

```
In[1]:= <<Graphics`FilledPlot`
In[2]:= FilledPlot[{x^2/18,Cos[x],Sin[x]},{x,0,2Pi},
        Fills -> {{{1,Axis},GrayLevel[0.8]},{{2,3},GrayLevel[0.6]}},
        Curves -> Front];
```

FilledPlot は、“packages” とよばれるファイル群の一つの関数です。通常は定義されていませんので、使う時は上のように指定する必要があります。

6　3 次元 graphics 集

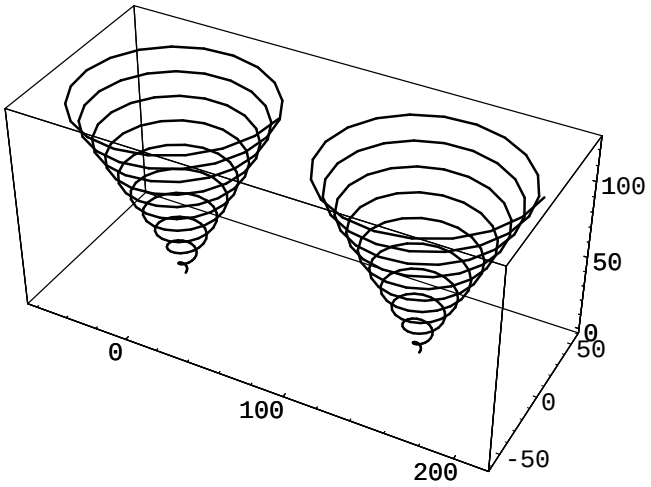
続いては、 3 次元の graphics の例です。



$z = \cos(xy)$ を $-3 \leq x, y \leq 3$ の範囲で描く。

```
In[1]:= Plot3D[Cos[x y],{x,-3,3},{y,-3,3},
        AxesLabel -> {"x","y","z"},PlotPoints -> 30];
```

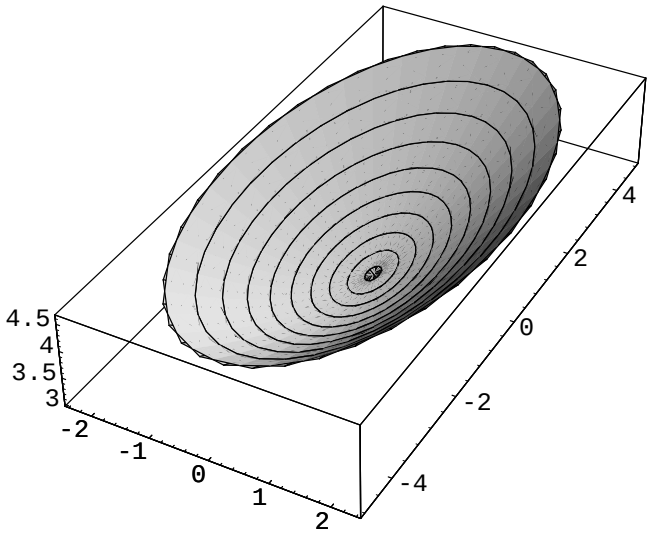
3 次元のプロットも、変数が一つ増えただけで、 2 次元と同じく簡単に作図できます。ただし、各方向の評価点の数は 15 なので、より滑らかな作図をしたい場合は、例のように PlotPoints を増やします。ただし、その分計算時間がかかり、メモリーも大量に消費します。



$0 \leq t \leq 20\pi$ の媒介変数 t で表された空間曲線
$(x, y, z) = (t\cos t, t\sin t, 2t)$, $(x, y, z) = (t\cos t + 150, t\sin t, 2t)$, を並べて描く。

```
In[1]:= ParametricPlot3D[{ {t Cos[t], t Sin[t], 2t},
        {t Cos[t] + 150, t Sin[t], 2t} },
        {t,0,20Pi}, PlotPoints -> 200];
```

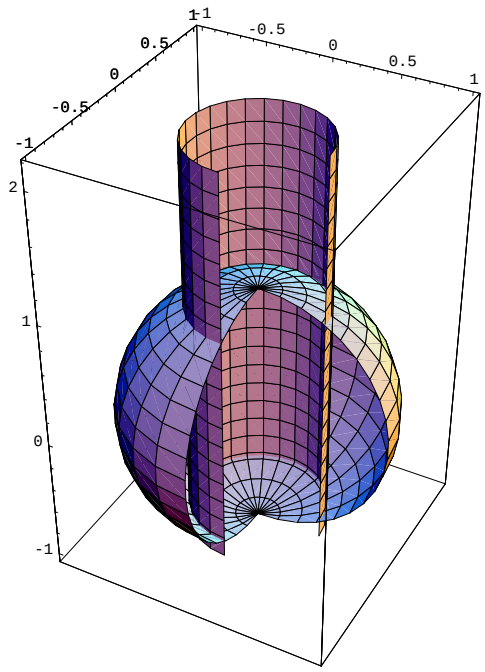
ParametricPlot3D は、ParametricPlot の 3 次元版です。



$0 \leq t \leq 2\pi,\ -1 \leq u \leq 1$ の媒介変数 $t,\ u$ で表された空間曲面
$(x, y, z) = ((e^u - e^{-u})\cos t,\ 2(e^u - e^{-u})\sin t,\ 3(e^u + e^{-u})/2)$ を描く。

```
In[1]:= ParametricPlot3D[ {(Exp[u]-Exp[-u])*Cos[t],
        2(Exp[u]-Exp[-u])*Sin[t], 3(Exp[u]+Exp[-u])/2 },
        {t,0,2Pi},{u,-1,1},ColorOutput -> GrayLevel];
```

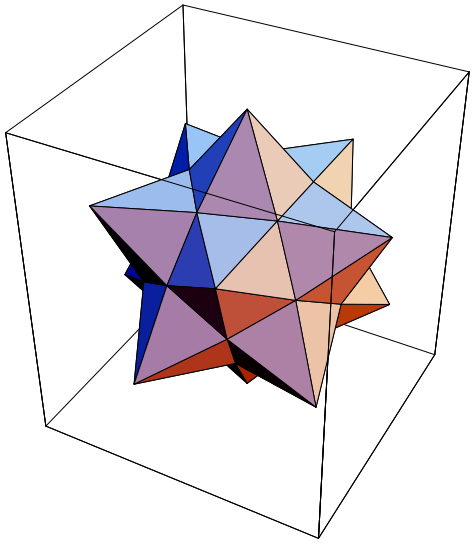
上のようにParametricPlot3D にもう一つの媒介変数の変域を指定すると、曲面が描けます。



$0 \leq u \leq 3\pi/2,\ -0 \leq v \leq \pi$ の媒介変数 $u,\ v$ で表された空間曲面
$(x, y, z) = (\sin(v)\cos(u),\ \sin(v)\sin(u),\ \cos(v))$ と $(x, y, z) = (\cos(u)/2,\ \sin(u)/2,\ v - 1)$ を描く。

```
In[1]:= ParametricPlot3D[{{Sin[v] Cos[u], Sin[v] Sin[u], Cos[v]},
        {Cos[u]/2, Sin[u]/2, v-1} },{u,0,3Pi/2},{v,0,Pi}];
```

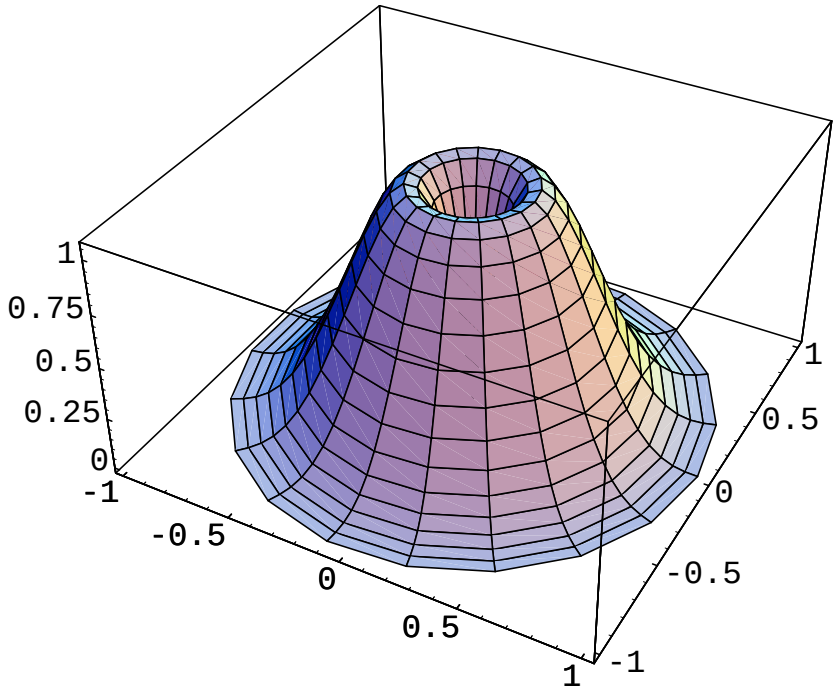
パラメータを制限すると、このような球と円柱がともに開かれた図も簡単に作図できます。



星型の 12 面体を描きます。

```
In[1]:= <<Graphics`Polyhedra`;
        Show[Stellate[Polyhedron[Dodecahedron]]];
```

簡単な多面体ならば、 Polyhedra パッケージを用いて作図できます。



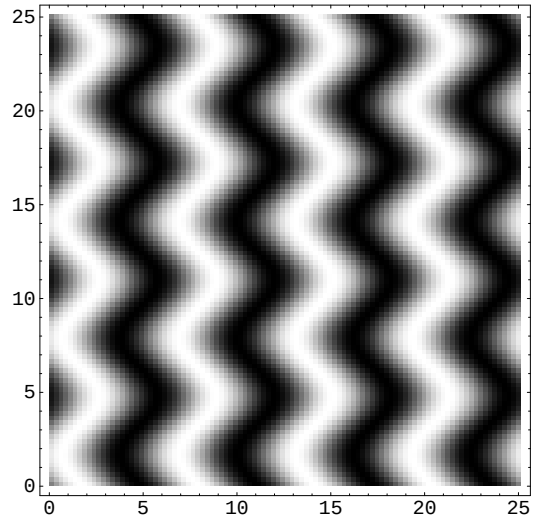
$y = 10x(1-x)^3$ を $0 \leq x \leq 1$ の間で z 軸について回転した曲面を描きます。

```
In[1]:= <<Graphics`SurfaceOfRevolution`
        SurfaceOfRevolution[10x*(1-x)^3,{x,0,1}];
```

回転曲面は SurfaceOfRevolution パッケージをロードして作図します。

7　その他の graphics 集

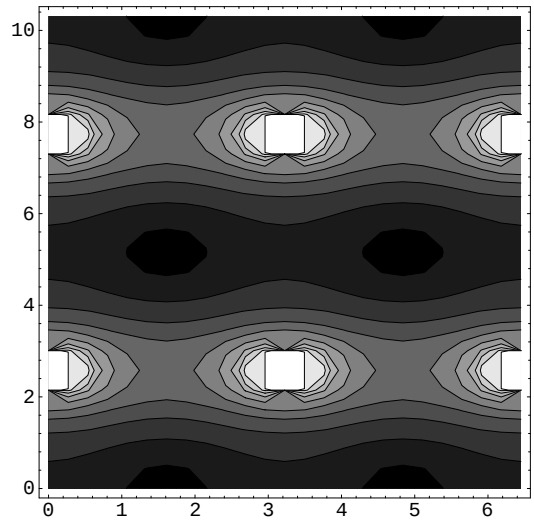
ここでは、等高線や密度分布，流れ場などの作成例をあげます。



$z = \sin(x + \sin(y))$ の密度グラフを $0 \leq x, y \leq 8\pi$ の範囲で作成します。

```
In[1]:= DensityPlot[Sin[x+Sin[y]],{x,0,8Pi},{y,0,8Pi},
        Mesh -> False, PlotPoints -> 100];
```

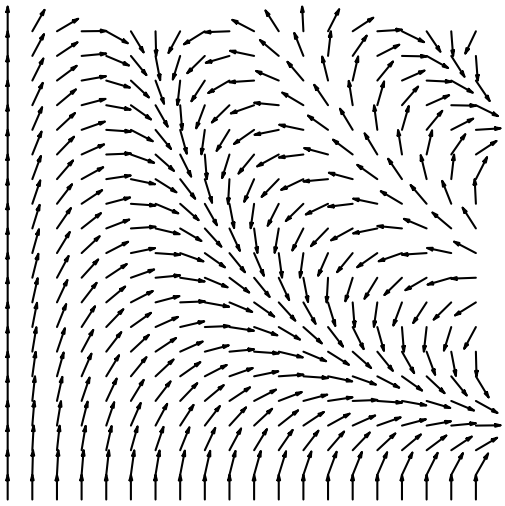
密度グラフは、与えられた関数の規則的な点の配列における値を表します。たくさんの領域に分割した場合は、メッシュを取り除いた方が奇麗です。



Jacobi 楕円関数 $\mathrm{cn}(x + iy|1/10)$ の絶対値の等高線を $0 \leq x \leq K(1/10)$,
$0 \leq y \leq K(9/10)$ の範囲でプロットします。なお $K(x)$ は第 1 種完全楕円積分です。

```
In[1]:= ContourPlot[Abs[JacobiCN[x+I y,1/10]],
        {x,0,4EllipticK[1/10]},{y,0,4EllipticK[9/10]},
        PlotPoints -> 25];
```

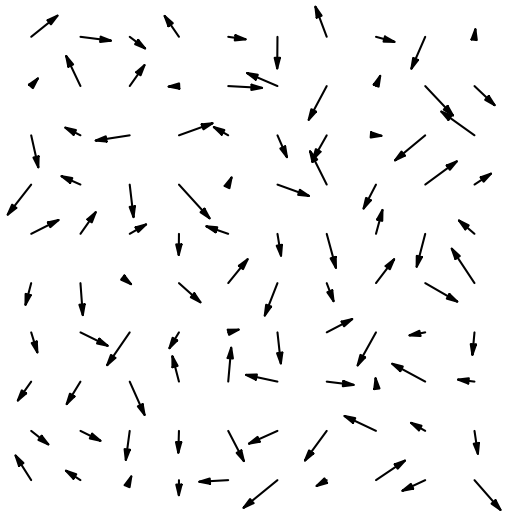
等高線グラフは、関数の「地形図」を与えます。等高線は、同じ高さの点を結んで得られる曲線です。



ベクトル値関数 $u = (\sin(xy), \cos(xy))$ のベクトル場を $0 \leq x, y \leq \pi$ の範囲で作図します。

```
In[1]:= <<Graphics‘PlotField‘
In[2]:= PlotVectorField[{Sin[x y],Cos[x y]},{x,0,Pi},{y,0,Pi},
        PlotPoints -> 20, ScaleFunction ->(.5#&)];
```

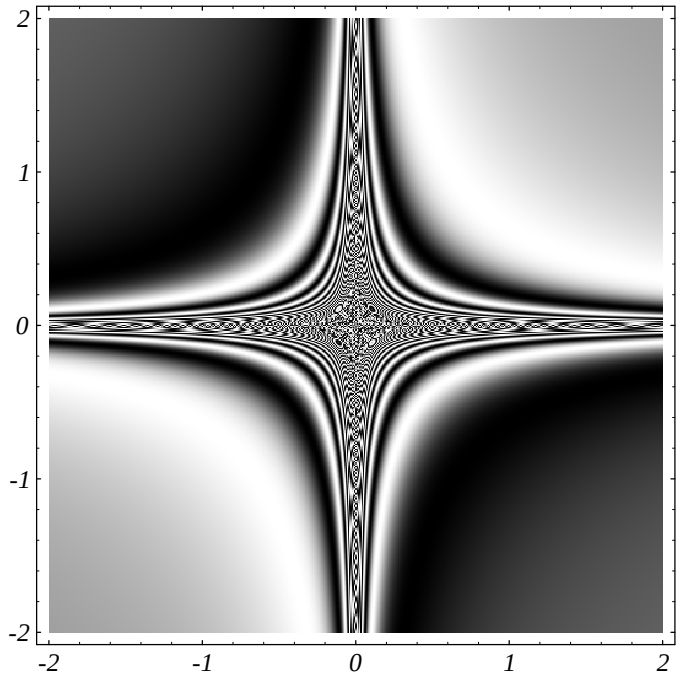
ベクトル場の作図には、PlotField パッケージを読み込みます。通常のベクトル場の他に、gradient vector field や Hamiltonian vector field なども作図できます。



10×10 のランダムなベクトルを作成し、リストをプロットします。

```
In[1]:= <<Graphics‘PlotField‘
In[2]:= varray = Table[{ Random[Real, {-0.7,0.7}],
        Random[Real,{-0.7,0.7}]},{i,10},{j,10}];
In[3]:= ListPlotVectorField[varray];
```

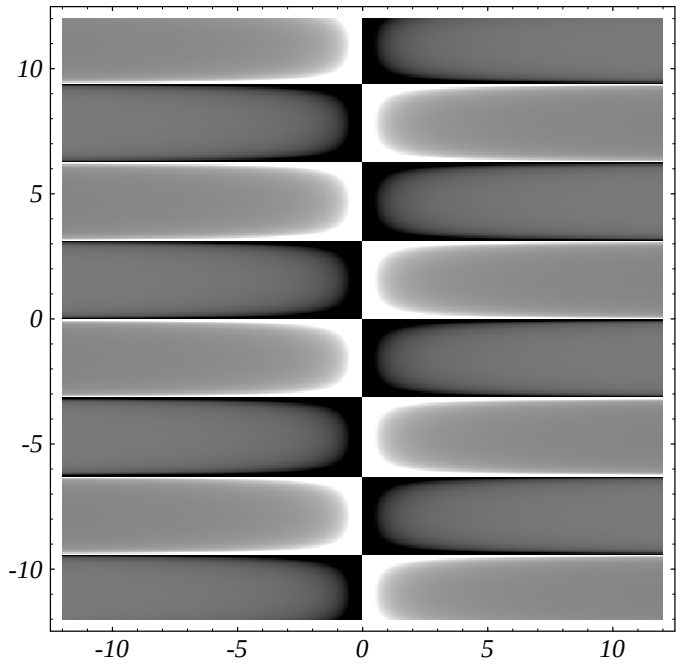
他のプログラムで作成したデータを、このようにリストとして作図することも可能です。



$z = \sin(1/(xy))$ の密度グラフを $-2 \leq x, y \leq 2$ の範囲で作図。

```
In[1]:= DensityPlot[ Sin[1/(x y)],{x,-2,2},{y,-2,2},
        PlotPoints -> 500, Mesh -> False]
```

出力したプリンターの解像度は 400dpi です。多分この二つはプリンターの解像度が高ければ、もっと奇麗にでるはずです。



$z = 1/(x \sin y)$ の密度グラフを $-12 \leq x, y \leq 12$ の範囲で作図。

```
In[1]:= DensityPlot[ 1/(x Sin[y]),{x,-12,12},{y,-12,12},
        PlotPoints -> 500, Mesh -> False]
```

8　プリンターへの出力

さて、出来上がった graphics をモニターでただ眺めている場合ではありません。早速ファイルに保存するなり、プリンターに出力するなりしましょう。まずはセンター二階の PostScript printer への出力方法です。

印刷は PSPrint で行ないます。その前に UXP(kyu-cc) に設定が必要です。

8.1　どんなプリンター？

PS プリンターです。解像度は 400dpi です。長年使ってきたので、そろそろ引退する計画です。

8.2　.rhosts の修正

自分の UXP(kyu-cc) のホームディレクトリに.rhosts という名前のファイルをを作成し、リモートシェルの実行権を設定します。

エディター (vi, emacs 等) を使い、.rhosts に次の一行を追加します。

```
qapls.cc.kyushu-u.ac.jp a79999a          <--- 下線部は自分の ID
```

.rhosts がない場合は、次のように新規に作成できます。

```
kyu-cc% echo qapls.cc.kyushu-u.ac.jp a79999a >> ~/.rhosts  ↵
```

8.3　PSPrint

kyu-cc の .rhosts の修正によって、 qapls から kyu-cc を経由して 2 階のプリンターに *Mathematica* graphics を出す準備ができました。

プリンターへの出力方法は *Mathematica* 内で graphics の直後にPSPrint[%] と入力するだけです。また、以前の graphics を印刷する場合は、 graphics window に表示してある Out[n] を入力します。例を参照下さい。

```
In[6]:= ListPlot[t,Prolog -> AbsolutePointSize[5]]  ↵

Out[6]= -Graphics-                    <--- Graphics 表示

In[7]:= PSPrint[%]  ↵                 <--- 直前の graphics のプリンター出力

Out[7]= -Graphics-

In[8]:= PSPrint[Out[6]]  ↵            <--- Out[6] のプリンター出力
                                           In[7] と同じこと
Out[8]= -Graphics-
```

ノートブックモードの場合は、メニューで Print を選択して下さい。

9 PS 形式のファイルに保存する

Mathematica の graphics は全て PostScript(PS) 形式で書かれています。 PS 形式で graphics を保存し、 ftp で手元に転送すれば、あとは TEX, JLATEX に取り込んだり、他の形式に変換したり、 Macintosh で自由に加工できたりします[7]。

9.1 Display

graphics のファイルへの保存は Display で行ないます。要領は、先ほどの PSPrint と同じです。保存先のファイル名を tmp とします。

```
In[6]:= ListPlot[t,Prolog -> AbsolutePointSize[5]]  ↵

Out[6]= -Graphics-                                   <--- Graphics 表示

In[7]:= Display["tmp",%]  ↵                          <--- ファイルへの出力

Out[7]= -Graphics-

In[8]:= Display["tmp",Out[6]]  ↵                     <--- In[7] と同じ操作

Out[8]= -Graphics-
```

9.2 psfix コマンド

ただし、このままの PS ファイルは、「生の」PS ファイルなので、 *Mathematica* 以外は解釈できません。

そこで、 qapls のコマンドとして用意されている psfix を使って普通の PostScript file に変換します。もとのファイルを tmp , 修正後を tmp.ps とすると、以下の要領です。

```
In[9]:= !psfix tmp > tmp.ps  ↵                       <--- Mathematica 内で使う

In[9]:=
```

もちろんpsfix コマンドは *Mathematica* を抜けた状態の UNIX コマンドとしても使えます。

```
qapls% psfix tmp > tmp.ps  ↵                         <--- UNIX コマンドとして使う
```

作成した PS ファイルは、カラーで作図していればカラーで、 Grayscale で作図していれば Grayscale で保存されます。なお、 ">" 先のファイルが既存の場合は、その旨を表示して変換してくれないのでご注意下さい。

9.3 LATEX に貼る場合

epsf.sty などを用い、 *Mathematica* で作成した PS ファイルを LATEX に読み込む場合、 graphics

[7] 妙にそういうことに詳しい人が周りにいるはずです。

によっては、余計な空白が入ってしまい納まりが悪いことがあります。

その場合には、PS ファイルの先頭の “BoundingBox” を無理矢理エディターで変更してしまう荒技があります。例として

```
BoundingBox: 72 72 540 720   →   BoundingBox: 72 200 580 580
                             →   BoundingBox: 70 150 580 630
```

などと修正してみて下さい。PS に詳しい人ならおわかりのように、例では縦のスケールを削っています。

10 カラープリンターに出力する

センター 2 階にはフルカラーの PS プリンターが装備されています。カラーで作図した *Mathematica* graphics はここで簡単に出力できます。

10.1 どんなプリンター？

フルカラー対応の PostScript プリンターです。用紙は A4，解像度は 400dpi です。

10.2 お金は？

ランニングコスト分の実費だけいただきます。一枚につき 360 円です。まだまだ、カラー出力はお高いです。

10.3 どこから出力するの？

カラープリンター横のワークステーション medics から “`lp -d npsf`” コマンドで出力します。

詳しくは medics 備えつけの説明書をお読み下さい、以下、出力方法です。

Step 1 まず、 medics に登録します。

方法は qapls と同様で、 UXP から “`touroku medics`” と入力して下さい。

Step 2 medics に login します。

medics のコンソールから login しなくても大丈夫です。 medics を他の人が使っていれば、別の端末から login して下さい。

Step 3 PS ファイルを ftp で medics に転送します。

テキスト形式で転送して下さい。

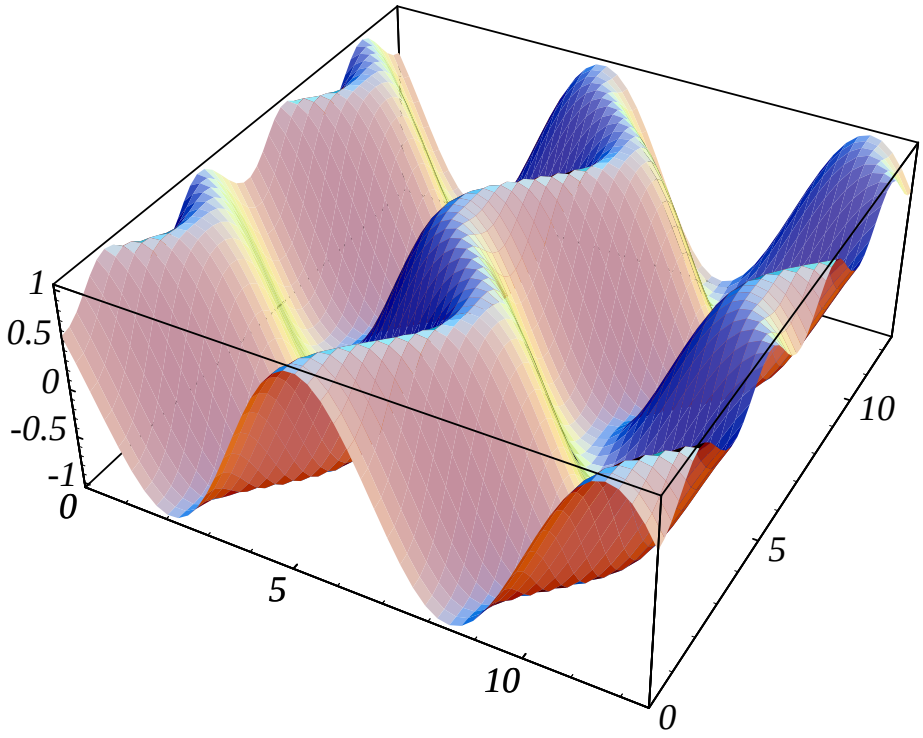
Step 4 出力したいファイルを `tmp.ps` とすると、以下のように入力下さい。

```
medics% lp -d npsf tmp.ps  ↵
```

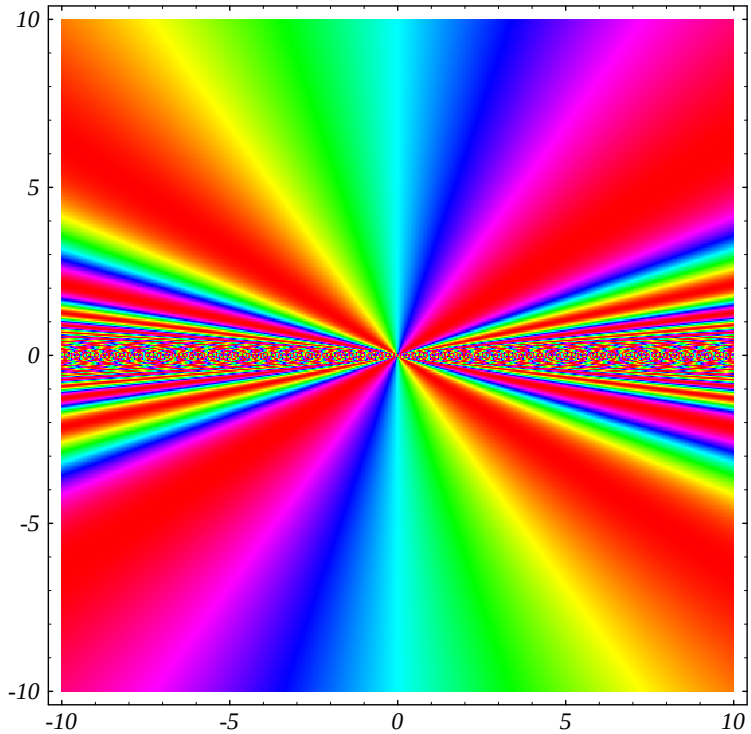
もちろん、上の方法の他に、 rsh を使った出力も可能です。しばらく後、プリンターがうなりはじめて、出力が得られるはずです。出力状態は `lpstat` コマンドで調べることができます。

また、 OHP に直接印刷することも可能です。詳しくは medics 備えつけの手引書を参照下さい。トラブルがあった場合は、受付またはシステム運用掛に御連絡下さい。

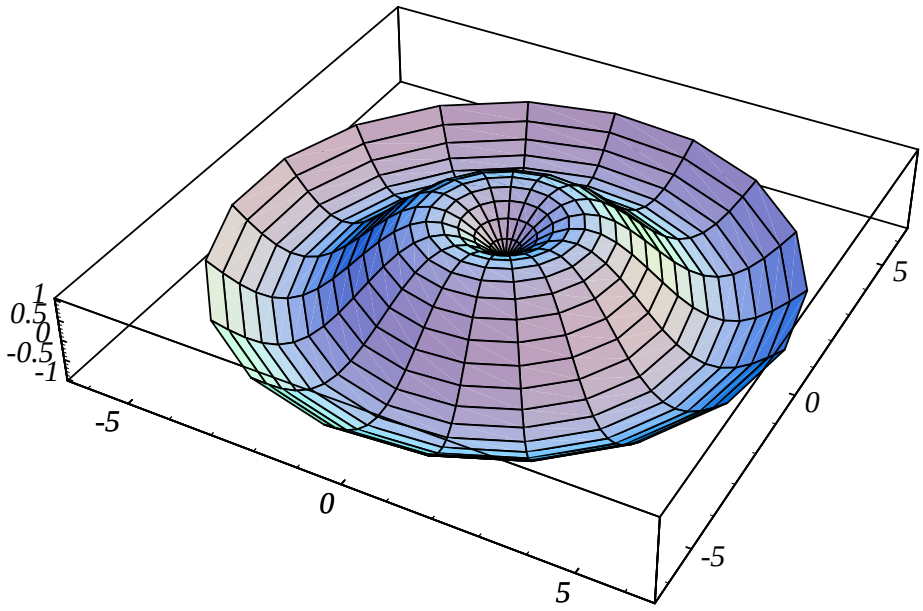
次ページは、カラープリンターで出力した *Mathematica* graphics 出力例です。なお、 4 番目の例の `fra.data` は、私が勝手に作成したファイルなので、そのまま真似しても graphics は出ません。データは自分でお作り下さい。



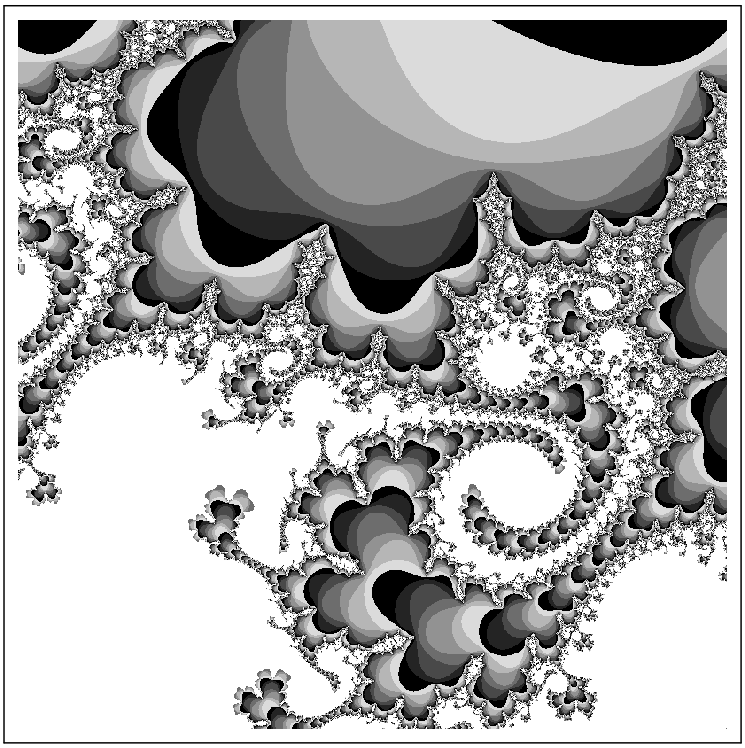

```
In[1]:= Plot3D[Cos[x+Cos[y]],{x,0,4Pi},{y,0,4Pi},
        PlotPoints -> 50, Mesh -> False];
```




```
In[1]:= DensityPlot[ Sin[x/y],{x,-10,10},{y,-10,10},
        PlotPoints -> 500, Mesh -> False, ColorFunction -> Hue];
```




```
In[1]:= <<Graphics`SurfaceOfRevolution`
In[2]:= SurfaceOfRevolution[Sin[x],{x,0,2Pi}];
```



```
In[1]:= ReadList["fra.data",Table[Number,{800}]];
In[2]:= ListDensityPlot[%,Mesh -> False,
        FrameTicks -> None];
```

楽をしましょう

例題を見てわかるように、 *Mathematica* のコマンドやオプションは、妙に長たらしいものが多く、いちいち打ち込むのはとても面倒です。その場合、ファイルに処理過程を書き、 *Mathematica* 内で読み込むことができます。これによって、細かな graphics の調整が少しの手間で済みます。

例として、 qapls の vi, emacs で `test.m` を次のように作成します。

```
DensityPlot[ Sin[x+Sin[y]], {x,0,8Pi},{y,0,8Pi},
             Mesh -> False,
             PlotPoints -> 100
           ];
```

この処理を実行するには、 *Mathematica* 内で次のようにするだけです。

```
In[1]:= << test.m  ↵              <--- test.m の実行

Out[2]:=
```

コマンドに間違いがあればその旨のメッセージが表示されます。そうでなければ graphics 画面が表示されるはずです。

このようにすれば、 qapls の window をもう一つ開いてファイルを編集・保存しながらデバッグ / 調整することができ、なんども同じコマンドを入力する必要もなくなり大幅に効率がアップします。

参考文献

Mathematica を上手に使うには手元に参考文献が最低一冊必要です。どれも結構な値段です。
以下はこの記事を書くにあたって参考にした文献リストです。簡単な説明をつけています。

- Stephen Wolfram : "*Mathematica* — A System for Doing Mathematics by Computer —, Second Edition" Addison-Wesley Publishing Company, 1991.
  - *Mathematica* のマニュアルです。利用者必携。
- Stephen Wolfram / 白水 重明 訳 : "*Mathematica* — A System for Doing Mathematics by Computer(日本語版) —, Second Edition" Addison-Wesley Publishers Japan / 星雲社, 1992.
  - *Mathematica* の日本語版マニュアルです。英語の苦手な利用者必携。
    関係ないですが「テン・マル」です。
- "*Mathematica* User's Guide for the X Front End" Wolfram Research, 1993.
  - Version 2.2 用の notebook front end の使用方法です。 *Mathematica* を買うと付いてきます。 Wolfram Research から直接購入することも可能だと思います。
- "*Mathematica* User's Guide for Unix Systems" Wolfram Research, 1993.
  - Version 2.2 用を Unix 環境で使いこなすための注意事項が書いてあります。これも *Mathematica* を買うと付いてきます。
- "Guide to Standard *Mathematica* Packages Supplement Version 2.2" Technical Report, Wolfram Research, 1993.

  – バージョンアップによるマニュアル記載以外の機能を解説した Technical Report です。 *Mathematica* を買うと付いてきますが、これは欲しい本です。

- "Major New Features in *Mathematica* Version 2.2" Technical Report, Wolfram Research, 1993.

  – Version 2.1 から 2.2 に変わる際に追加された機能の解説です。

- T. W. Gray, J. Glynn / 時田 節, 藤村 行俊 訳 : "*Mathematica* 数学の探索" トッパン, 1994.

  – Gray さんは Wolfram Research の共同設立者 , Glynn さんは数学者です。表題の通り使用方法というより、数学への応用に主眼がおかれています。特に複素平面やフラクタルに興味のある方にはお奨めです。定価 6,900 円。

- M. L. Abell, J. P. Braselton / 川瀬 宏海, 五島 奉文, 佐藤 穂, 田澤 義彦 訳 : "*Mathematica* ハンドブック" 東京電機大学出版局, 1994.

  – 全てのコマンドとオブジェクトをアルファベット順に解説した参考書です。定価 6,000 円。

- 宮岡 悦良 : "*Mathematica* 数学の道具箱" プレーン出版, 1994.

  – *Mathematica* を用いた数学の入門書です。高校数学から大学数学へ進もうとされる人に対し、グラフィックスもふんだんに取り入れ、わかりやすく解説しています。定価 2,800 円。