

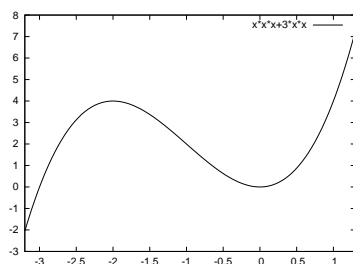
gnuplot 入門

緑川研究室 gnuplot 愛好会

1 多項式

3 次関数

関数 $y = x^3 + 3x^2$ を描いてみよう。

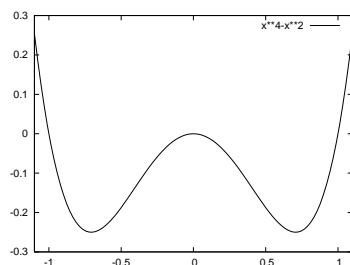


gnuplot を立ち上げて、以下のように入力する。

```
gnuplot> set xrange[-3.2:1.3]
gnuplot> plot x**3+3*x**2
```

4 次式

$$y = x^4 - x^2$$

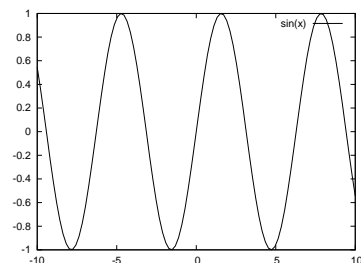


```
gnuplot> set xrange[-1.1:1.1]
gnuplot> plot x**4-x**2
```

2 三角関数

正弦関数

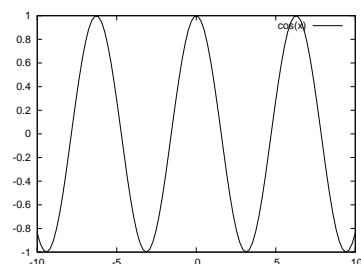
$$y = \sin x$$



```
gnuplot> reset
gnuplot> plot sin(x)
```

余弦関数

$$y = \cos x$$



```
gnuplot> reset
gnuplot> plot cos(x)
```

注意

全ての設定をクリアするときは、

```
gnuplot> reset
```

と打ち込む。

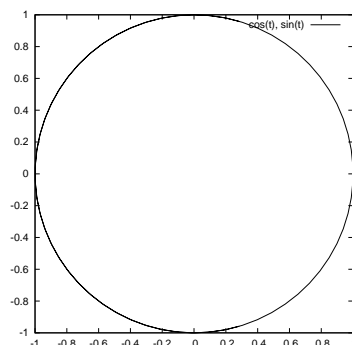
3 パラメトリック曲線

円

$$x = a \cos t, \quad y = a \sin t$$

とおくと、

$$x^2 + y^2 = a^2$$



単位円 ($a = 1$) の場合

```
gnuplot> set size square
```

```
gnuplot> set parametric
```

dummy variable is t for curves, u/v for surfaces

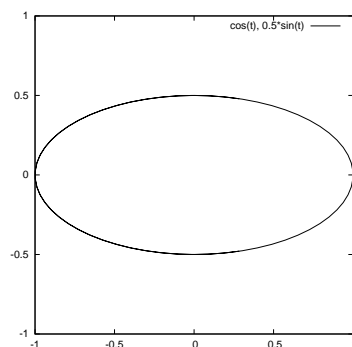
```
gnuplot> plot cos(t), sin(t)
```

楕円

$$x = a \cos t, \quad y = b \sin t$$

とおくと、

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$



$a = 1, b = 0.5$ の場合

```
gnuplot> set size square
```

```
gnuplot> set xrange[-1:1]
```

```
gnuplot> set yrange[-1:1]
```

```
gnuplot> set parametric
```

dummy variable is t for curves, u/v for surfaces

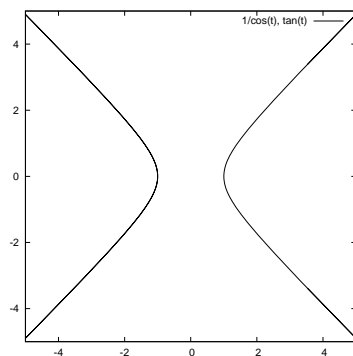
```
gnuplot> plot cos(t), 0.5*sin(t)
```

双曲線

$$x = \frac{a}{\cos t}, \quad y = b \tan t$$

とおくと、

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1$$



$a = 1, b = 1$ の場合

```
gnuplot> set size square
```

```
gnuplot> set xrange[-5:5]
```

```
gnuplot> set yrange[-5:5]
```

```
gnuplot> set parametric
```

dummy variable is t for curves, u/v for surfaces

```
gnuplot> plot 1/cos(t), tan(t)
```

リサージュ曲線

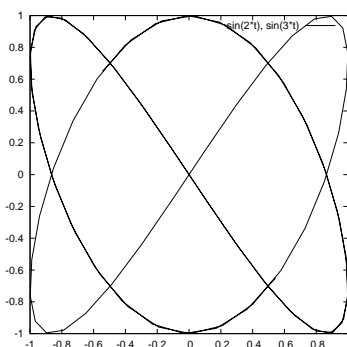
$$x = \sin at, \quad y = \sin bt$$

$a = 2, b = 3$ の場合

```
gnuplot> reset
```

```
gnuplot> set size square
```

```
gnuplot> set parametric
```



dummy variable is t for curves, u/v for surfaces

```
gnuplot> plot sin(2*t), sin(3*t)
```

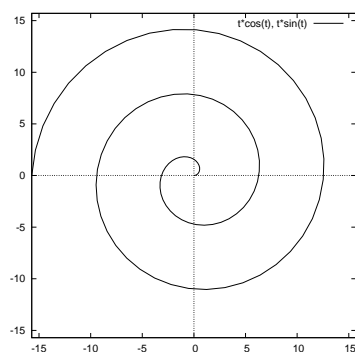
アルキメデスの渦巻線

$$r = at \quad (t \geq 0)$$

$a = 1$ の場合

$$x = t \cos t, \quad y = t \sin t$$

```
gnuplot> reset
gnuplot> set size square
gnuplot> set xrange[-5*pi:5*pi]
gnuplot> set yrange[-5*pi:5*pi]
gnuplot> set parametric
```



dummy variable is t for curves, u/v for surfaces

```
gnuplot> plot [0: 5*pi] t*cos(t), t*sin(t)
```

4 曲面

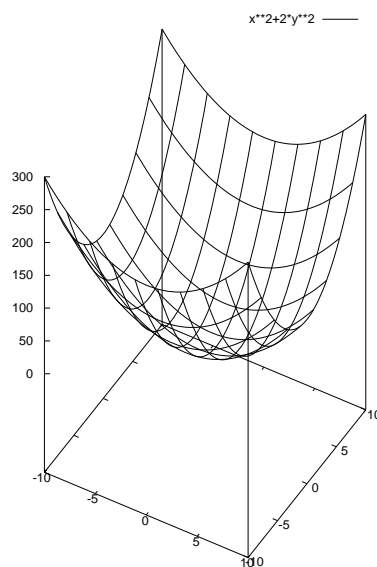
楕円的放物面

$$z = \frac{x^2}{a^2} + \frac{y^2}{b^2}$$

$a = 1, b = 1/\sqrt{2}$ の場合

```
gnuplot> splot x**2+2*y**2
```

マウスでぐりぐり動かすことができます。



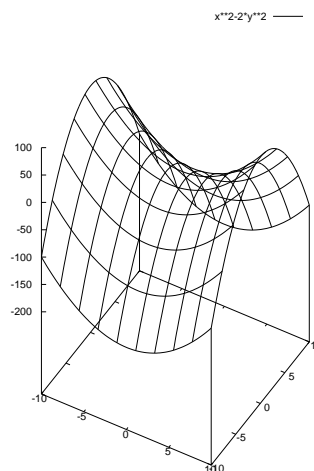
双曲的放物面

$$z = \frac{x^2}{a^2} - \frac{y^2}{b^2}$$

$a = 1, b = 1/\sqrt{2}$ の場合

```
gnuplot> splot x**2-2*y**2
```

マウスでぐりぐり動かすことができます。



5 パラメトリック曲面

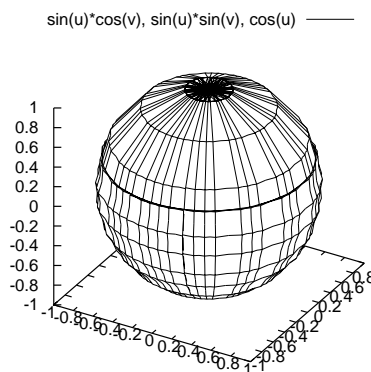
球

$$x = \sin(u) \cos(v)$$

$$y = \sin(u) \sin(v)$$

$$z = \cos(u)$$

```
gnuplot> reset
gnuplot> set view equal xyz
gnuplot> set ticslevel 0
gnuplot> set isosamples 24
gnuplot> set hidden3d
gnuplot> set parametric
```



dummy variable is t for curves, u/v for surfaces

```
gnuplot> splot sin(u)*cos(v), sin(u)*sin(v), cos(u)
```

トーラス

$$x = a \cos(u)(d + \cos(v))$$

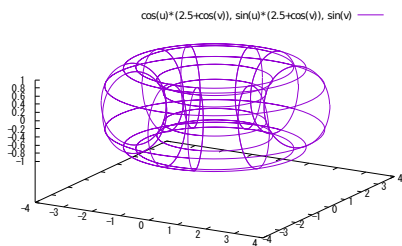
$$y = b \sin(u)(d + \cos(v))$$

$$z = c \sin(v)$$

$a = b = c = 1$, $d = 2.5$ の場合

```
gnuplot> set parametric
```

```
gnuplot> splot cos(u)*(2.5+cos(v)), sin(u)*(2.5+cos(v)), sin(v)
```

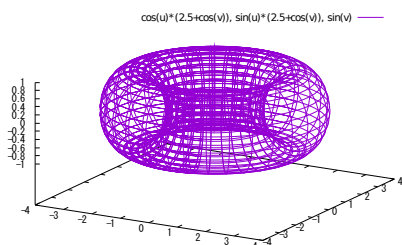


図がちょっと粗いですね。もう少し細かく描画します。

```
gnuplot> set isosamples 50
```

```
gnuplot> replot
```

こうして得られたのが、次の図です。

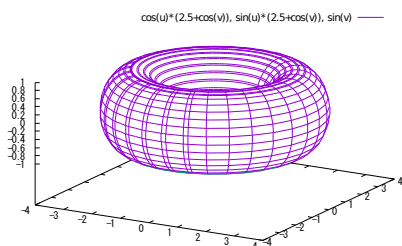


今度は、陰影処理をしましょう。

```
gnuplot> set hidden3d
```

```
gnuplot> replot
```

下の図のように、トーラスらしく見えるようになりました。



6 更に進んだ使い方 - スクリプト形式

今までの使い方では、プロンプト画面から命令をキーボード入力(コマンド入力)で描画を行いました。これでは修正を施して再描画をおこなおうとすると何度も同じ命令を打ち込み、煩わしくなります。そこで、この煩わしさを避けるためには、一群の命令をシートに書いて実行するスクリプト形式で行います。

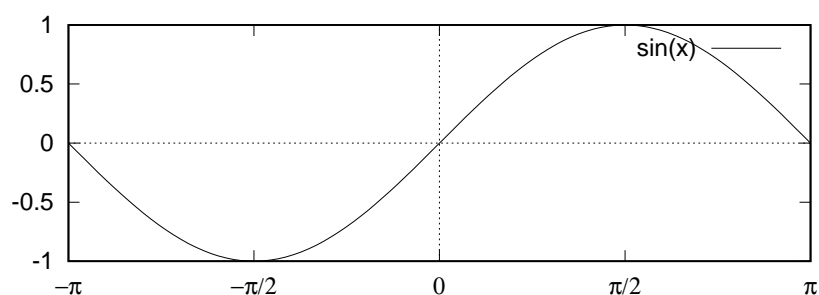
まず、新規作成でテキストドキュメント(メモ帳)を開き、そこに命令を書き込みます。ファイルに新しい名前を付けて保存します。その時の拡張子は、plt です。命令を実行する場合には、`wgnuplot.exe` を起動して File → Open から plt ファイルを開きます。

6.1 再び三角関数

それでは、最初に取り上げた三角関数の描画をスクリプト形式で行ってみましょう。メモ帳に以下のスクリプトを書き込んで、`sine.plt` という名前で保存します。

```
sine.plt
reset
set terminal postscript
set output "sine.ps"
set size 1 , 0.3
set size ratio -1
set xzeroaxis
set yzeroaxis
set xtics ("-p" -pi, "-p/2" -pi/2 , "0" 0.0, "p/2" pi/2 , "p" pi) font "Symbol"
set ytics -1, 0.5, 1
set xrange[-pi:pi]
plot sin(x)
```

この様にして描いた図は、次のようになります。



スクリプト(命令)の意味は以下の通りです。

- (1) reset
すべての命令を御破算にする。これを書かないと、思わぬ命令が残っていて、イメージ通りの図が描けなくて苦労することがあります。
- (2) set terminal postscript
最終の出力形式を指定。ここでは、ポストスクリプト言語で出力する。
- (3) set output "sine.ps"
生成物 (画像) の名前を sine.ps とする。
- (4) set size 1 , 0.3
画像の描画領域の大きさ x 軸方向の大きさを 1 に、 y 軸方向の大きさを 0.3 にとる。
- (5) set size ratio -1
 x 軸の単位長さを y 軸の単位長さを同じにする。
- (6) set xzeroaxis
 x 軸を表示
- (7) set yzeroaxis
 y 軸を表示
- (8) set xtics (" -p" -pi, "-p/2" -pi/2 ,"0" 0.0, "p/2" pi/2 ,"p" pi) font "Symbol"
pi は、 π (円周率) を表す。{" -p" -pi } は、-pi すなわち、 $-3.1415\dots$ の位置の目盛にラベル $-\pi$ を付けることを意味します。Symbol フォントは、ローマ字に対応するギリシア文字を表します。
- (9) set ytics -1, 0.5, 1
 y 軸に目盛を -1 から 0.5 刻みで 1 まで付ける。
- (10) set xrange[-pi:pi]
 x の描画範囲を $-\pi$ から π までとする。
- (11) plot sin(x)
 $y = \sin(x)$ のグラフを描画する。

6.2 $y^2 = x^2(x+1) + c$ のグラフ

スクリプト形式を利点の1つとして、パラメータの値を少しずつ変えてグラフを描く場合が考えられます。例として、 $y^2 = x^2(x+1) + c$ のグラフを取り上げます。定数 c の値を 0.01, 0, -0.01 と3つの場合について描いたのが下の図です。

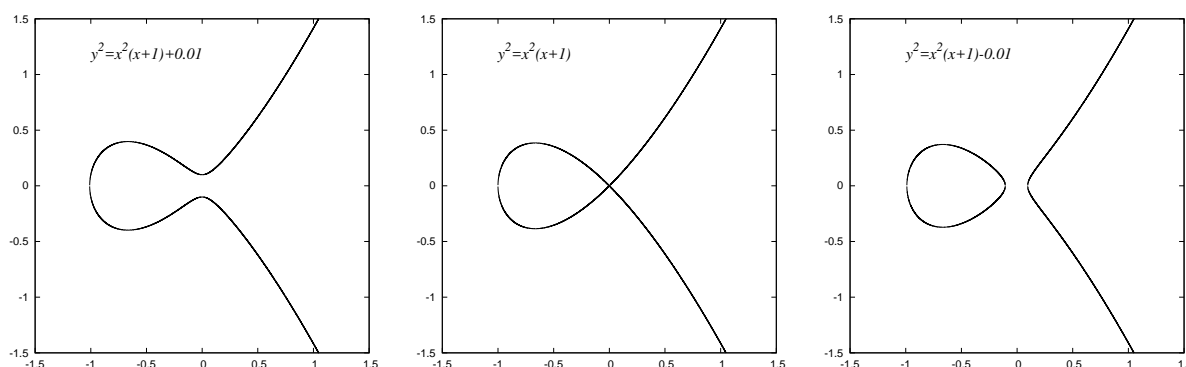


図 1: $y^2 = x^2(x+1) + c$ のグラフ 左から、 $c = 0.01, 0, -0.01$ の場合

ここで、 $c = 0$ の場合のスクリプトを以下に示します。適当に手直しをすることにより、 $c = \pm 0.01$ の場合も描くことができます。

`y2=f(x).plt`

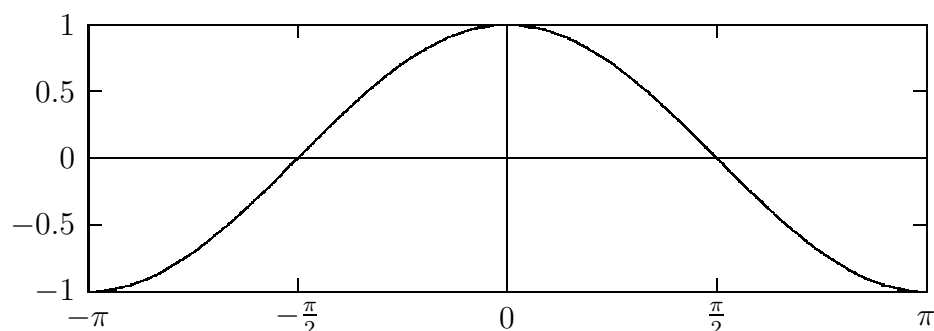
```
reset
set terminal postscript portrait
set output "y2=x2(x+1).ps"
set nokey
set size square
set xrange[-1.5:1.5]
set yrange[-1.5:1.5]
set samples 40000    #標本点の数を4000にする。
c=0.0
set label "y^2=x^2(x+1)" at -1, 1.2 font 'Times Roman-Italic, 20'
plot sqrt(x**2*(x+1)+c), -sqrt(x**2*(x+1)+c) w l ls -1
```

【注】 `sqrt(x**2*(x+1)+c), -sqrt(x**2*(x+1)+c) w l ls -1` について

- (1) 関数の定義において、 x^2 は `x**2` と表します。一方、ラベルに x^2 と書きたいときは、`x^2` と打ち込みます。
- (2) `plot` コマンドでは、 $y = \sqrt{x^2(x+1) + c}$ と $y = -\sqrt{x^2(x+1) + c}$ を同時に書かせています。特に、指定しないと2番目の関数は点線で描きますので、改めて実線で描くように指示します。`w l ls -1` は、with line linestyle -1 の略で、`ls -1` は、実線を表します。

7 L^AT_EX に出力

前のセクションでサイン曲線を描いたとき、 x 軸の目盛のラベルが π の分数の場合には、 $\pi/2$ のように表しました。これを、 $\frac{\pi}{2}$ のように表したいときには、ターミナルへの出力を L^AT_EX にします。スクリプトには、L^AT_EX のコマンドを書くことができます。ただし、数式モード (math mode) における命令は、 $\pi \dots$ の代わりに $\pi\pi \dots$ と π を重ねて打ちます。例えば、分数 $\frac{\pi}{2}$ と書きたい場合には、 $\text{\textbackslash frac\{\textbackslash pi\}\{2\}}$ となります。次は、コサイン曲線を L^AT_EX に出力した例です。



plt ファイルは、以下の通りです。

```
cosTex.plt
reset
set terminal latex
set nokey
set output "cosTex.tex"
set size ratio -1

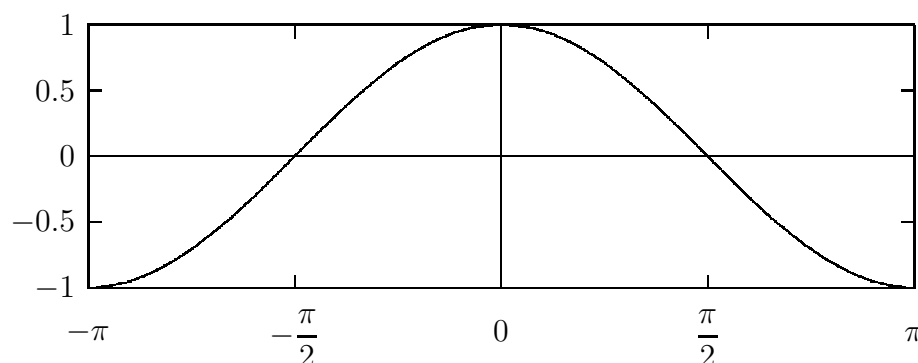
set xtics ("{-\pi}" -pi, {"-\frac{\pi}{2}}" -pi/2, {"0"} 0.0, \
{"\frac{\pi}{2}}" pi/2, {"\pi}" pi)

set ytics -1, 0.5, 1
set xzeroaxis
set yzeroaxis
set xrange [-pi:pi]
set yrange [-1:1]
plot cos(x)
```

【注】

- (1) \backslash は、半角の π と同じです。
- (2) 1 つの命令が長すぎて、途中で改行したい場合には、 $\backslash(\pi)$ を打ってから改行します。

先の方法では、分数 $\frac{\pi}{2}$ の活字 π が小さくなっています。L^AT_EX では、この調整を自動で行います。これが気に入らない場合には、 $\text{\texttt{\${\displaystyle \frac{\pi}{2}}\$}}$ として、さらにラベルの位置を微調整をおこないます。そうして得られた結果が、次のグラフです。



plt ファイルは、以下の通りです。

```
cosTex2.plt
reset
set terminal latex
set nokey
set output "cosTex.tex"
set size ratio -1
set xtics ("\\lower4ex\\hbox{\\$\\{-\\pi\\}$}" -pi, \
"\\lower6ex\\hbox{\\$\\{\\displaystyle -\\frac{\\pi\\}{2}\\}$}" -pi/2, \
"\\lower4ex\\hbox{\\$0\\$}" 0.0, \
"\\lower6ex\\hbox{\\$\\{\\displaystyle \\frac{\\pi\\}{2}\\}$}" pi/2, \
"\\lower4ex\\hbox{\\$\\{\\pi\\}$}" pi)
set ytics -1, 0.5, 1
set xzeroaxis
set yzeroaxis
set xrange [-pi:pi]
set yrange [-1:1]
plot cos(x)
```