8 プロット

Mathematica の優れた能力の一つは，関数のグラフを色々な方法で，簡単に描く（プロットする）ことができることである．

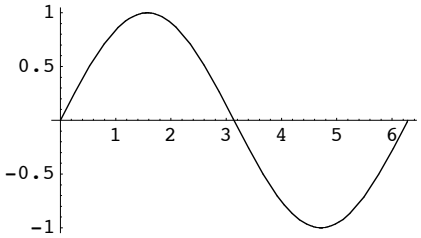
■ 1 変数関数のグラフ

y = f(x) の形の 1 変数関数のグラフを描くには， Plot を用いる．

■ 基本的な使い方

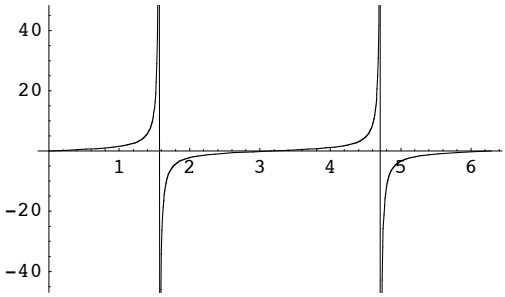
- Plot[f, {x, xmin, xmax}] は，式 f を x の関数としたグラフを， x の範囲が xmin ~ xmax の範囲でプロットする．

```
In[1]:= Plot[Sin[x], {x, 0, 2 Pi}]
```




```
Out[1]= - Graphics -
```

```
In[2]:= Plot[Tan[x], {x, 0, 2 Pi}]
```

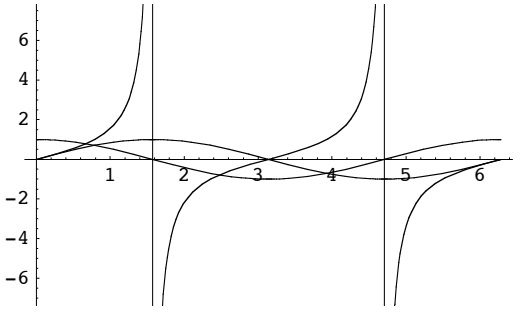



```
Out[2]= - Graphics -
```

y=tan(x) の値域は -∞ から ∞ にわたるので，そのグラフを完全に描くことはできない．このような場合， Plot は値域を適当に絞ってグラフを描く．

- Plot[{f1, f2, ...}, {x, xmin, xmax}] は，複数の関数 f1, f2, ... のグラフを同一座標平面上に一括してプロットする．

```
In[3]:= Plot[{Sin[x], Cos[x], Tan[x]}, {x, 0, 2 Pi}]
```




```
Out[3]= - Graphics -
```

先ほどの y=tan(x) のグラフに描かれた値域よりも，狭まっている．これは， sin(x), cos(x) のグラフが潰れて見えなくなるのを防ぐために，自動的に行なわれている．

■ プロット仕様の指定

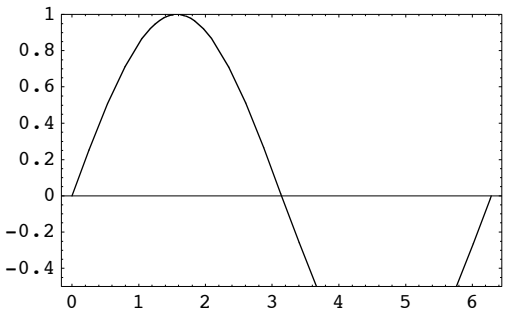
オプションに何も指定しないデフォールトの状態で， Plot が描くグラフは，描かれる値域は自動的に設定され，目盛のついた x，y 軸が表示されたものとなる．これらの描画の仕方は Plot にオプションを指定することにより変更することができる．

オプションは， Plot の 3 番目以降の引き数に option->value の書式で指定する．

- Plot[f, {x, xmin, xmax}, option1->value1, option2->value2, ...] はプロット仕様を指定したグラフをプロットする．

次のグラフは， Frame->True で囲み枠を表示するように指定し， PlotRange-> {-0.5, 1} で描画する値域を -0.5 から 1 の範囲に指定したものである．

```
In[4]:= Plot[Sin[x], {x, 0, 2 Pi}, Frame -> True, PlotRange -> {-0.5, 1}]
```




```
Out[4]= - Graphics -
```

オプション指定で用いる一般的な設置値には次の 5 個がある．

Automatic	内部アルゴリズムを用いて求めた最適地を用いる
None	なし
All	すべて
True	有効
False	無効

以下に Plot でよく使うオプションの一覧とそのデフォルト値（省略値）とその解説を載せておく．

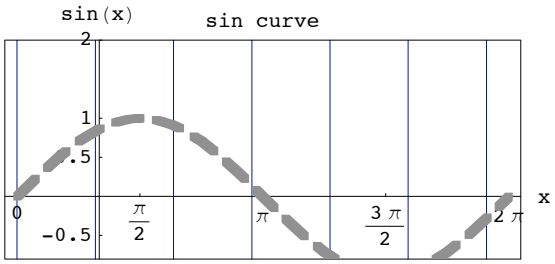
オプション	デフォルト値	解説
AspectRatio	1 / GoldenRatio	プロット画面の縦幅を横幅で割った比．1 / GoldenRatio は約 0.62．Automatic とすると，縦座標の 1 の長さと横座標の 1 の長さとが等し
PlotLabel	None	図の上部につける題目．
PlotRange	Automatic	プロットする値域の範囲．{ymin, ymax} の形で指定．Automatic は適当に All はすべての値域をプロットする．
Axes	Automatic	軸を表示するかどうか．表示しないときは False とする．
AxesLabel	None	x 軸と y 軸に付けるラベル．{xlabel, ylabel} の形で指定．
AxesOrigin	Automatic	軸の原点，すなわち軸の交わる点の座標．
Ticks	Automatic	軸にふる目盛．{{x1, x2, ...}, {y1, y2, ...}} の形で指定．None は目盛なし．
Frame	False	プロット画面の囲み枠を表示するかどうか． True だと表示する．
FrameLabel	None	囲み枠のラベル．ラベルは下，左，上，右の順のリストで指定．
FrameTicks	Automatic	囲み枠に目盛をふるかどうか． None とすればふらない．
GridLine	None	方眼紙状の目盛線を引くかどうか．{{x1, x2, ...}, {y1, y2, .. Automatic は主な目盛に方眼線を引く．
PlotStyle	Automatic	グラフの曲線の太さ (Thickness) 色 (GrayLevel, RGBColor) 破線 {Thickness[0.008], GrayLevel[0.5], Dashing[{0.02, 0.(
PlotPoints	25	グラフを描くためにサンプリングする点の最小数．

すべてのオプションとそのデフォルトの値のリストは， Options という関数をもちいれば得られる．

```
In[5]:= Options[Plot]

Out[5]= {AspectRatio → 1/GoldenRatio, Axes → Automatic, AxesLabel → None,
         AxesOrigin → Automatic, AxesStyle → Automatic, Background → Automatic,
         ColorOutput → Automatic, Compiled → True, DefaultColor → Automatic,
         DefaultFont :→ $DefaultFont, DisplayFunction :→ $DisplayFunction, Epilog → {},
         FormatType :→ $FormatType, Frame → False, FrameLabel → None, FrameStyle → Automatic,
         FrameTicks → Automatic, GridLines → None, ImageSize → Automatic,
         MaxBend → 10., PlotDivision → 30., PlotLabel → None, PlotPoints → 25,
         PlotRange → Automatic, PlotRegion → Automatic, PlotStyle → Automatic,
         Prolog → {}, RotateLabel → True, TextStyle :→ $TextStyle, Ticks → Automatic}
```

次のグラフは正弦曲線のグラフを，縦横の単位長さを等しくして，題目 "sin curve" を付けて，描画値域を指定して，座標軸に "x", "sin(x)" のラベルを付けて，座標原点を a に指定して，座標の目盛を指定して，画面の囲み枠をつけて，囲み枠の目盛は付けずに，縦方向の方眼線をつけて，曲線の太さ(Thickness)色(GrayLevel)破線(Dashing)を指定して表示したものである．

```
In[6]:= sinCurve = Plot[Sin[x], {x, 0, 2 Pi}, AspectRatio -> Automatic,
          PlotLabel -> "sin curve", PlotRange -> {-0.8, 2}, Axes -> True,
          AxesLabel -> {"x", "sin(x)"}, AxesOrigin -> {Pi / 3, 0},
          Ticks -> {{0, Pi / 2, Pi, 3 Pi / 2, 2 Pi}, {-0.5, 0, 0.5, 1, 2}}, Frame -> True,
          FrameTicks -> None, GridLines -> {{0, 1, 2, 3, 4, 5, 6}, None},
          PlotStyle -> {Thickness[0.015], GrayLevel[0.5], Dashing[{0.05, 0.03}]}]
```

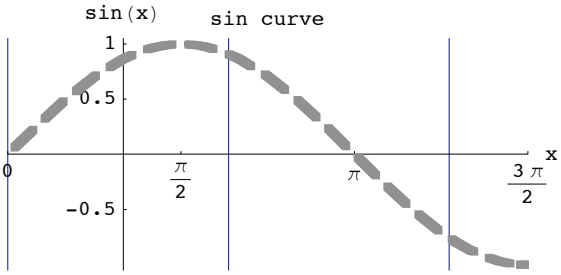



```
Out[6]= - Graphics -
```

Plot で表示された結果はグラフィクスオブジェクトを呼ばれる形のデータとなる．これは， Show を用いるともう一度画面に表示させることができる．この時，以前に指定したオプションを変更することができる．ただし， PlotStyle と PlotPoints は変更できない．

Show[g, option1->value1, option2->value2, ...] はグラフィクスオブジェクト g をオプションを指定して表示する．

```
In[7]:= Show[sinCurve, PlotRange -> {{0, 3 Pi / 2}, All},
         Frame -> False, GridLines -> {{0, 2, 4, 6}, None}]
```

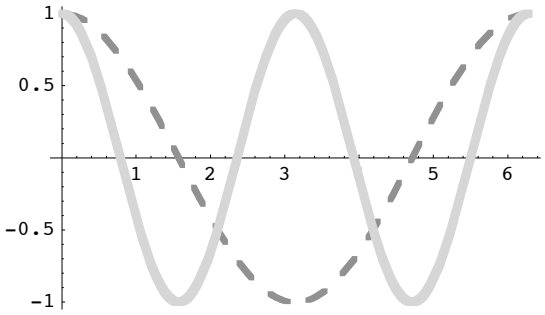



```
Out[7]= - Graphics -
```

PlotRange→{{xmin,xmax},{ymin,ymax}} の形で，値域と同時に定義域も指定することができる．

複数の関数のグラフを同時にプロットし，それらを異なる線で描きたいときには， PlotStyle->{{-Thickness[0.01], ...}, {Thickness[0.02], ...}, ...} のように指定する．

```
In[8]:= cosCurve = Plot[{Cos[x], Cos[2 x]}, {x, 0, 2 Pi},
          PlotStyle → {{Thickness[0.01], GrayLevel[0.5], Dashing[{0.04, 0.05}]},
            {Thickness[0.015], GrayLevel[0.8]}}]
```

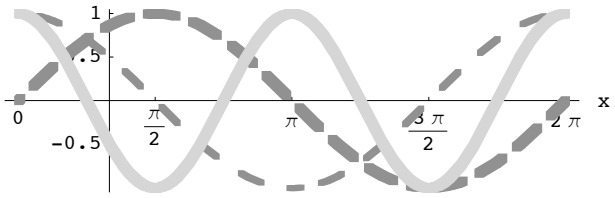



```
Out[8]= - Graphics -
```

Show を用いると，以前に計算した複数のプロット図を一括して表示することもできる.

Show[{g1, g2, ... gk}, option->value, ...] はグラフィクスオブジェクト g1, g2, ..., gk を同一の画面に一括表示する. option, ... で指定されなかった表示オプションは g1 のものが優先される.

```
In[9]:= Show[{sinCurve, cosCurve}, PlotRange -> All, Frame -> False,
          GridLines -> None, PlotLabel -> None, AxesLabel -> {"x", None}]
```




```
Out[9]= - Graphics -
```

g1, g2, ... gk の順に上書きされ, gk が一番上になる.

練習問題： $y = \sin(x) - x/4$ のグラフと $y = \cos(x)$ のグラフの交点は何個あるか.

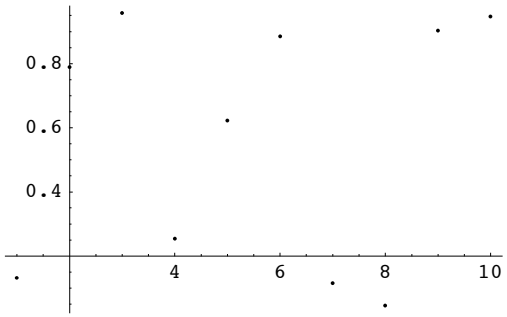
■ 1 次元データ配列のプロット

{d1, d2, ..., dk} という 1 次元データ配列を折れ線グラフなどにしてプロットするには ListPlot を用いる.

- ListPlot[{d1, d2, ..., dk}, option->value, ...] は x, y 平面上に (i, di) という座標の点をプロットする.

```
In[10]:= data = Table[Random[], {10}]

Out[10]= {0.131499, 0.788764, 0.958197, 0.253982,
           0.622596, 0.884813, 0.114871, 0.0448507, 0.902483, 0.946919}
```

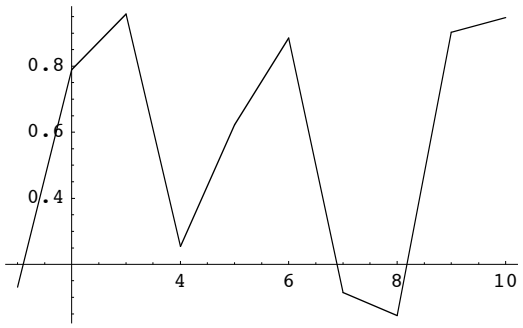
```
In[11]:= ListPlot[data]
```




```
Out[11]= - Graphics -
```

オプションで PlotJoined->True を指定すれば折れ線グラフとなる.

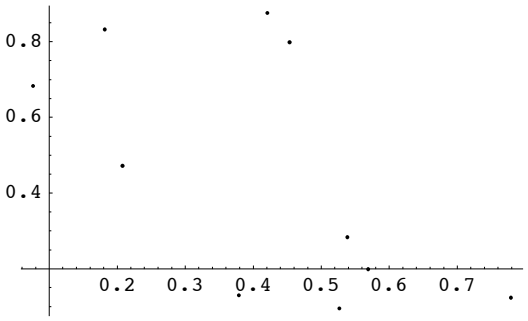
```
In[12]:= ListPlot[data, PlotJoined -> True]
```




```
Out[12]= - Graphics -
```

- ListPlot[{{x1,y1},{x2,y2},...},option→value,...] は (xi,yi) を座標とする点をプロットする.

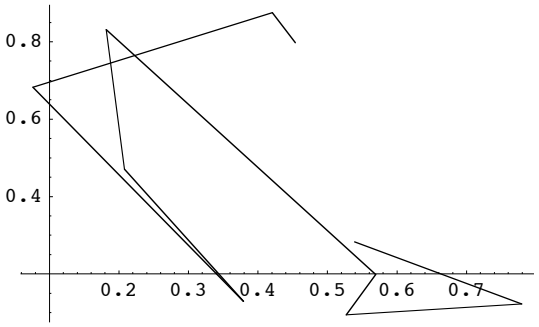
```
In[13]:= data2 = Table[{Random[], Random[]}, {10}]

Out[13]= {{0.538622, 0.282897}, {0.779055, 0.122773},
          {0.526787, 0.0949274}, {0.569404, 0.198285},
          {0.181815, 0.83156}, {0.2078, 0.471361}, {0.379125, 0.129493},
          {0.0763014, 0.682597}, {0.420928, 0.875511}, {0.453705, 0.797783}}

In[14]:= ListPlot[data2]
```




```
Out[14]= - Graphics -
```

これも PlotJoined->True というオプションを指定すると，折れ線となる.

In[15]:= **ListPlot[data2, PlotJoined -> True]**



Out[15]= - Graphics -

その他 ListPlot のオプションには， Plot とほぼ同様の AspectRatio,Axes,AxesLabel,Axes-Origin,Frame,FrameLabel,FrameTicks,GridLines,PlotLabel,PlotRange,Ticks などがある．

■ 媒介変数表示された平面曲線のプロット

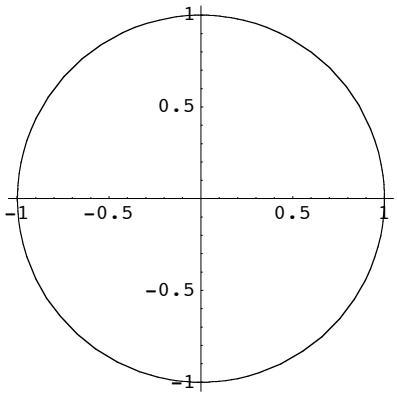
時刻 t における位置が (x(t), y(t)) であるような点の軌跡などのように，一つの変数で x, y の両座標が媒介変数表示されているときの曲線をプロットするには， ParametricPlot を用いる．

■ ParametricPlot

- ParametricPlot[{x, y}, {t, tmin, tmax}] は， x, y を t の関数として， (x, y) を座標にもつような点の軌跡を， t = tmin~tmax においてプロットする．
- ParametricPlot[{{x1, y1}, {x2, y2}, ...}, {t, tmin, tmax}] は複数の曲線を同一の画面にプロットする．

円をプロットするには， (cos(t), sin(t)) という座標の点を t=0~2π で動かせばよい．描画面の縦横比を Automatic とすると，描画された円がきちんと正円になる．

In[16]:= **ParametricPlot[{Cos[t], Sin[t]}, {t, 0, 2 Pi}, AspectRatio -> Automatic]**



Out[16]= - Graphics -

練習問題： (cos(3t), sin(4t)) を座標とする点を t=0~2π で動かしたときの軌跡を表示せよ.

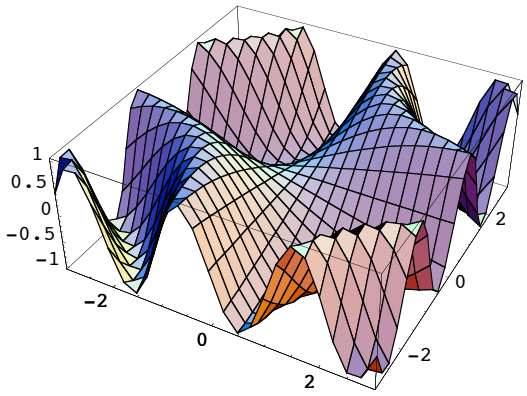
■ 2 変数関数のグラフ

2 変数の関数 z = f(x,y) のグラフを描くには，3 次元曲面プロット Plot3D，密度プロット DensityPlot，等高線プロット ContourPlot などを用いる.

■ 3 次元曲面プロット Plot3D

● Plot3D[f, {x, xmin, xmax}, {y, ymin, ymax}] は f を x と y との関数として， z = f(x, y) で表される 3 次元曲面のプロットをする.

```
In[17]:= sinxy = Plot3D[Sin[x y], {x, -3, 3}, {y, -3, 3}]
```




```
Out[17]= - SurfaceGraphics -
```

このプロット図を見ると，関数値の変化が激しいところでは，曲面が滑らかにつながっていない．このように， Plot3D は関数値の変化の激しい関数のプロットには不向きであるが，オプションでサンプリングの数を増やすことにより，解決できることもある.

Plot3D には非常に多くのオプションがあるが，基本的なものだけを解説する.

オプション	デフォルト値	解説
Mesh	True	曲面上に縦横の網線を引くかどうか.
Shading	True	プロット曲面に濃淡処理を施すかどうか. .
PlotRange	Automatic	プロットする座標軸範囲の指定. All か {zmin, zmax} か {{xmin, xmax}, {ymin, ymax}, {zmin, zmax}} かのいずれかの形て
Axes	Automatic	軸を表示するかどうか. 表示しないときは False とする.
AxesLabel	None	軸につけるラベル. {xlabel, ylabel, zlabel} の形で 3 軸のラベルを リストにしないと z 軸のラベルとなる.
Boxed	True	直方体プロット空間を枠で囲むかどうか.
ColorFunction	Automatic	陰影付けする色の指定.
FaceGrid	None	プロット空間を囲む枠に目盛線をどのようにつけるか.
FrameLabel	None	囲み枠のラベル. ラベルは下，左，上，右の順のリストで指定.

HiddenSurface	True	陰線処理 （手前の曲面で隠れて見えない部分を表示しない処理） をするか
Lighting	True	面の色付けに照明効果を使うかどうか.
PlotPoints	25	グラフを描くためにサンプリングする点の最小数.

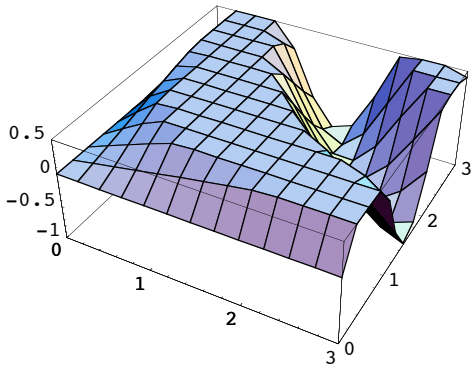
すべてのオプションとそのデフォルト値のリストは次のとおりである.

```
In[18]:= Options[Plot3D]

Out[18]= {AmbientLight → GrayLevel[0], AspectRatio → Automatic, Axes → True,
          AxesEdge → Automatic, AxesLabel → None, AxesStyle → Automatic,
          Background → Automatic, Boxed → True, BoxRatios → {1, 1, 0.4},
          BoxStyle → Automatic, ClipFill → Automatic, ColorFunction → Automatic,
          ColorFunctionScaling → True, ColorOutput → Automatic,
          Compiled → True, DefaultColor → Automatic, DefaultFont :→ $DefaultFont,
          DisplayFunction :→ $DisplayFunction, Epilog → {}, FaceGrids → None,
          FormatType :→ $FormatType, HiddenSurface → True, ImageSize → Automatic,
          Lighting → True, LightSources → {{{1., 0., 1.}, RGBColor[1, 0, 0]},
            {{1., 1., 1.}, RGBColor[0, 1, 0]}, {{0., 1., 1.}, RGBColor[0, 0, 1]}},
          Mesh → True, MeshStyle → Automatic, Plot3Matrix → Automatic, PlotLabel → None,
          PlotPoints → 25, PlotRange → Automatic, PlotRegion → Automatic, Prolog → {},
          Shading → True, SphericalRegion → False, TextStyle :→ $TextStyle, Ticks → Automatic,
          ViewCenter → Automatic, ViewPoint → {1.3, -2.4, 2.}, ViewVertical → {0., 0., 1.}}
```

以前に計算したプロットは Show を用いて再表示することができる．このときオプションの指定を変更することができる．ただし， PlotPoints などは変更できない.

プロット範囲を x=0~3, y=0~3, z=-1~1/2 に指定してプロット.

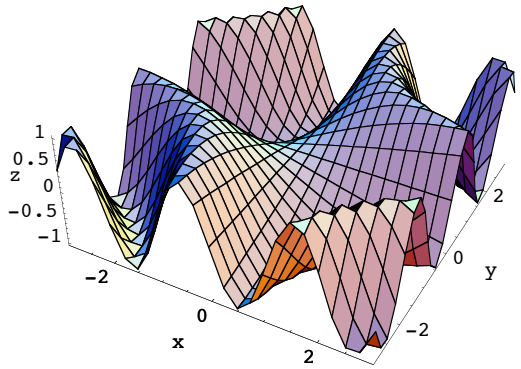
```
In[19]:= Show[sinxy, PlotRange -> {{0, 3}, {0, 3}, {-1, 1 / 2}}]
```




```
Out[19]= - SurfaceGraphics -
```

枠を表示しないで，軸にラベル x, y, z を付けて表示.

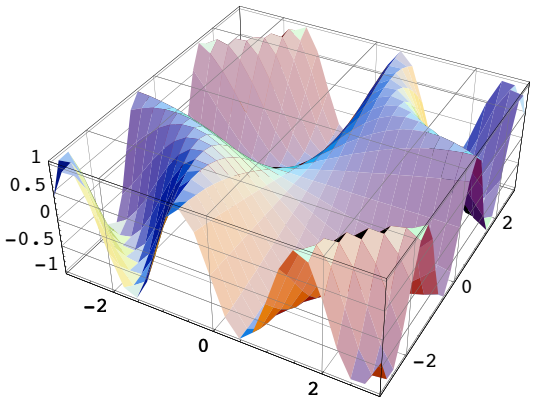
```
In[20]:= Show[sinxy, Boxed -> False, AxesLabel -> {"x", "y", "z"}]
```




```
Out[20]= - SurfaceGraphics -
```

プロット空間の上下左右前後の面に方眼線を表示し，曲面上の縦横の線（メッシュ）を消して表示.

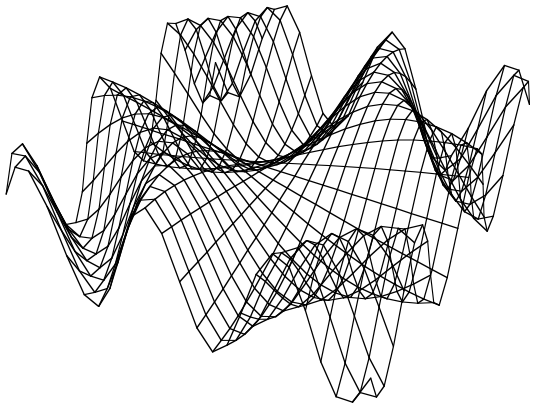
```
In[21]:= Show[sinxy, FaceGrids -> All, Mesh -> False]
```




```
Out[21]= - SurfaceGraphics -
```

軸，枠を消して，曲面を陰線処理を施さないでスケルトンのみを表示.

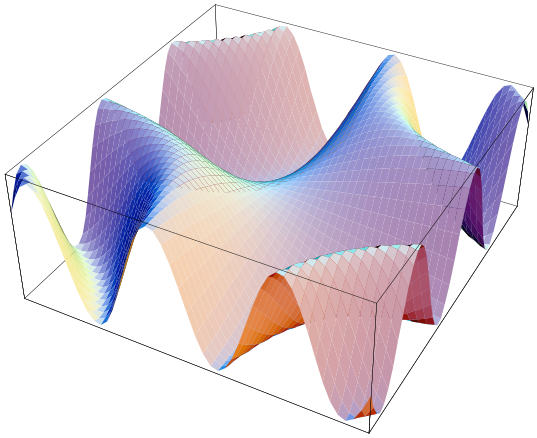
```
In[22]:= Show[sinxy, Axes -> False, Boxed -> False, HiddenSurface -> False]
```



```
Out[22]= - SurfaceGraphics -
```

サンプリング点数を 50 に増やして，メッシュ，軸を消して表示.

```
In[23]:= Plot3D[Sin[x y], {x, -3, 3}, {y, -3, 3},
          PlotPoints -> 50, Mesh -> False, Axes -> False]
```



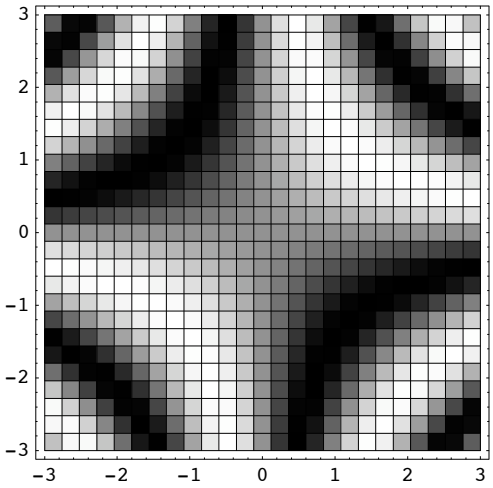
```
Out[23]= - SurfaceGraphics -
```

■ 密度プロット DensityPlot

Plot3D を用いて関数の 3 次元的なグラフを描くのは，関数の値の変化を曲面の形として表示でき，直感的に理解するためには利点がある．しかしながら，その画像からでは，視点に依存して陰線処理で見えない部分があったり，微妙な値の変化を捉えるのが難しかったりする．それに対して DensityPlot は色の変化で関数値の変化を表す 2 次元的な図を作成する．この図は色の変化さえきちんと読み取れば，値の変化を正確に捉えることができる．特に値の変化から何かのパターンを掴みたいときには有用である.

- DensityPlot[f, {x, xmin, xmax}, {y, ymin, ymax}] は，f を x, y の関数として， x = xmin~xmax, y = ymin~ymax の範囲で，関数値に応じた濃淡で表す密度プロットを作る.

```
In[24]:= DensityPlot[Sin[x y], {x, -3, 3}, {y, -3, 3}]
```




```
Out[24]= - DensityGraphics -
```

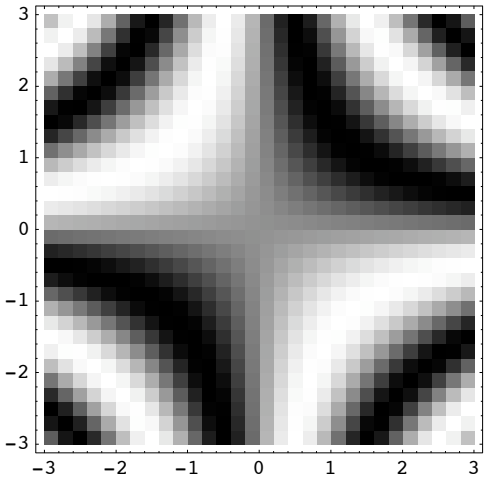
DensityPlot のオプションには次のようなものがある.

```
In[25]:= Options[DensityPlot]

Out[25]= {AspectRatio → 1, Axes → False, AxesLabel → None, AxesOrigin → Automatic,
          AxesStyle → Automatic, Background → Automatic, ColorFunction → Automatic,
          ColorFunctionScaling → True, ColorOutput → Automatic,
          Compiled → True, DefaultColor → Automatic, DefaultFont :→ $DefaultFont,
          DisplayFunction :→ $DisplayFunction, Epilog → {}, FormatType :→ $FormatType,
          Frame → True, FrameLabel → None, FrameStyle → Automatic, FrameTicks → Automatic,
          ImageSize → Automatic, Mesh → True, MeshStyle → Automatic, PlotLabel → None,
          PlotPoints → 25, PlotRange → Automatic, PlotRegion → Automatic,
          Prolog → {}, RotateLabel → True, TextStyle :→ $TextStyle, Ticks → Automatic}
```

色と関数値の関係は，デフォルトでは [関数値小（濃）<-->（淡）関数値大] となっている．これを変更するには ColorFunction を変更する．ColorFunction に指定する関数は 0~1 の実数値を引数にとり，色を返す関数（例えば GrayLevel）を指定する．またサンプリングの個数は x，y 方向とも 15 がデフォルトである．

次の例は，サンプリング数を x，y 方向とも 30 にして，メッシュ（縦横の線）を付けないで，濃淡と関数値の関係をデフォルトと逆にして表示したものである．

```
In[26]:= DensityPlot[Sin[x y], {x, -3, 3}, {y, -3, 3},
          PlotPoints -> 30, Mesh -> False, ColorFunction -> (GrayLevel[1 - #] &)]
```




```
Out[26]= - DensityGraphics -
```

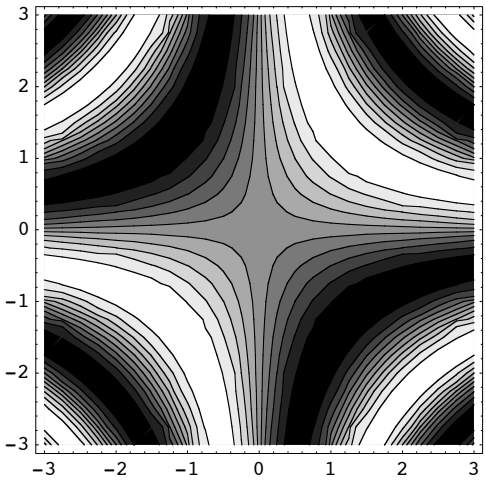
GrayLevel[0] は黒， GrayLevel[1] は白である。

■ 等高線プロット ContourPlot

関数値の等高線を引いて，関数の変化を地形図のように分り易く表示するものが， CountourPlot である．

- ContourPlot[f, {x, xmin, xmax}, {y, ymin, ymax}] は， f を x，y の関数として， x = xmin~xmax, y = ymin~ymax の範囲で，等高線とともに濃淡で関数値の変化を表した，等高線プロットを作る．

```
In[27]:= ContourPlot[Sin[x y], {x, -3, 3}, {y, -3, 3}]
```




```
Out[27]= - ContourGraphics -
```

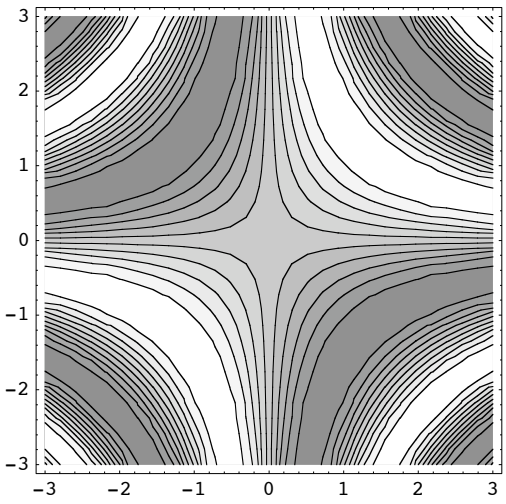
この例を見ると，等高線の形が間違っている部分がある．このようなことは，関数値の変化が緩やかな部分で起こりやすい．すなわち， Plot3D とは正反対に ContourPlot は関数値の変化の緩やかな関数のプロットには不向きである．しかし，オプションでサンプリング数を増やすことにより，この問題を解決できることもある．

ContourPlot のオプションとそのデフォルトのリストは次のとおり．Options[ContourPlot]

```
In[28]:= Options[ContourPlot]

Out[28]= {AspectRatio → 1, Axes → False, AxesLabel → None, AxesOrigin → Automatic,
          AxesStyle → Automatic, Background → Automatic, ColorFunction → Automatic,
          ColorFunctionScaling → True, ColorOutput → Automatic, Compiled → True,
          ContourLines → True, Contours → 10, ContourShading → True,
          ContourSmoothing → True, ContourStyle → Automatic, DefaultColor → Automatic,
          DefaultFont :→ $DefaultFont, DisplayFunction :→ $DisplayFunction,
          Epilog → {}, FormatType :→ $FormatType, Frame → True, FrameLabel → None,
          FrameStyle → Automatic, FrameTicks → Automatic, ImageSize → Automatic,
          PlotLabel → None, PlotPoints → 25, PlotRange → Automatic, PlotRegion → Automatic,
          Prolog → {}, RotateLabel → True, TextStyle :→ $TextStyle, Ticks → Automatic}
```

先ほどの問題は，サンプリングの数を x, y 方向とも 30 とすることで解決できる．また色の濃い部分が黒すぎて等高線がはっきりしないのを防ぐには， ColorFunction 変更するとよい．

```
In[29]:= ContourPlot[Sin[x y], {x, -3, 3}, {y, -3, 3},
          PlotPoints -> 30, ColorFunction -> (GrayLevel[# / 2 + 1 / 2] &)]
```




```
Out[29]= - ContourGraphics -
```

■ 2 次元配列データのプロット

{{f11, f12, ..., f1n}, {f21, f22, ..., f2n}, ..., {fm1, fm2, ..., fmn}} という 2 次元配列の形のデータをプロットするには, ListPlot3D, ListDensityPlot, ListContourPlot の 3 種類の関数を用いる.

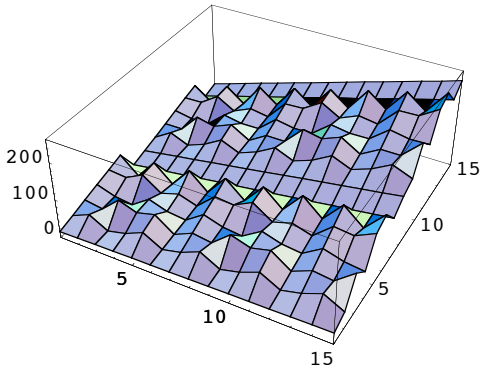
■ ListPlot3D

- ListPlot3D[{{f11, ..., f1n}, ..., {fm1, ..., fmn}}] は, x=i, y=j おける高さが z=fij であるような曲面を表示する.

次のような 15×15 のサイズの 2 次元配列データを ListPlot3D で表示してみる.

```
In[30]:= data = Table[i j + Mod[i^j, 7] 10, {i, 15}, {j, 15}]

Out[30]= {{11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25},
          {22, 44, 16, 28, 50, 22, 34, 56, 28, 40, 62, 34, 46, 68, 40},
          {33, 26, 69, 52, 65, 28, 51, 44, 87, 70, 83, 46, 69, 62, 105},
          {44, 28, 22, 56, 40, 34, 68, 52, 46, 80, 64, 58, 92, 76, 70},
          {55, 50, 75, 40, 55, 40, 85, 80, 105, 70, 85, 70, 115, 110, 135},
          {66, 22, 78, 34, 90, 46, 102, 58, 114, 70, 126, 82, 138, 94, 150},
          {7, 14, 21, 28, 35, 42, 49, 56, 63, 70, 77, 84, 91, 98, 105},
          {18, 26, 34, 42, 50, 58, 66, 74, 82, 90, 98, 106, 114, 122, 130},
          {29, 58, 37, 56, 85, 64, 83, 112, 91, 110, 139, 118, 137, 166, 145},
          {40, 40, 90, 80, 100, 70, 100, 100, 150, 140, 160, 130, 160, 160, 210},
          {51, 42, 43, 84, 75, 76, 117, 108, 109, 150, 141, 142, 183, 174, 175},
          {62, 64, 96, 68, 90, 82, 134, 136, 168, 140, 162, 154, 206, 208, 240},
          {73, 36, 99, 62, 125, 88, 151, 114, 177, 140, 203, 166, 229, 192, 255},
          {14, 28, 42, 56, 70, 84, 98, 112, 126, 140, 154, 168, 182, 196, 210},
          {25, 40, 55, 70, 85, 100, 115, 130, 145, 160, 175, 190, 205, 220, 235}}
```

```
In[31]:= ListPlot3D[data]
```




```
Out[31]= - SurfaceGraphics -
```

ListPlot3D のオプションとそのデフォルト値は次のとおりである.

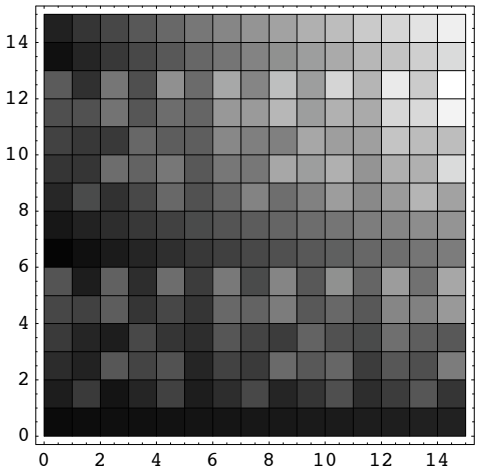
```
In[32]:= Options[ListPlot3D]

Out[32]= {AmbientLight → GrayLevel[0], AspectRatio → Automatic, Axes → True,
          AxesEdge → Automatic, AxesLabel → None, AxesStyle → Automatic,
          Background → Automatic, Boxed → True, BoxRatios → {1, 1, 0.4}, BoxStyle → Automatic,
          ClipFill → Automatic, ColorFunction → Automatic, ColorFunctionScaling → True,
          ColorOutput → Automatic, DefaultColor → Automatic, DefaultFont :→ $DefaultFont,
          DisplayFunction :→ $DisplayFunction, Epilog → {}, FaceGrids → None,
          FormatType :→ $FormatType, HiddenSurface → True, ImageSize → Automatic,
          Lighting → True, LightSources → {{{1., 0., 1.}, RGBColor[1, 0, 0]},
            {{1., 1., 1.}, RGBColor[0, 1, 0]}, {{0., 1., 1.}, RGBColor[0, 0, 1]}}, Mesh → True,
          MeshRange → Automatic, MeshStyle → Automatic, Plot3Matrix → Automatic,
          PlotLabel → None, PlotRange → Automatic, PlotRegion → Automatic, Prolog → {},
          Shading → True, SphericalRegion → False, TextStyle :→ $TextStyle, Ticks → Automatic,
          ViewCenter → Automatic, ViewPoint → {1.3, -2.4, 2.}, ViewVertical → {0., 0., 1.}}
```

■ ListDensityPlot

- ListDensityPlot[{{f11, ..., f1n}, ..., {fm1, ..., fmn}}] は，データ値 fij を色の濃淡で表した密度プロットを作る.

ListDensityPlot も，[データ値小（濃）<-->（淡）データ値大] がデフォルトの色の対応である.

In[33]:= **ListDensityPlot[data]**

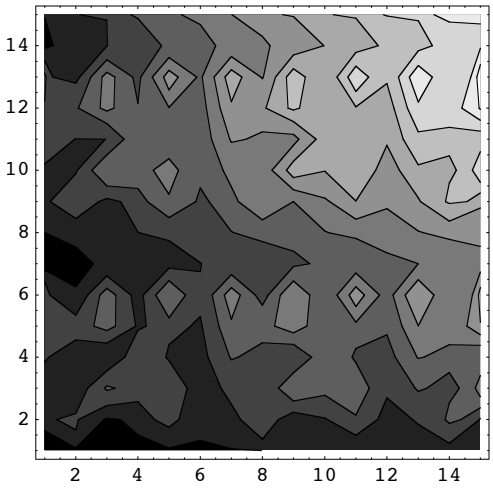


Out[33]= - DensityGraphics -

■ ListContourPlot

● ListContourPlot[{{f11, ..., f1n}, ..., {fm1, ..., fmn}}] は，データ値 fij を色の濃淡と等高線とで表した等高線プロットを作る．

In[34]:= **ListContourPlot[data]**



Out[34]= - ContourGraphics -

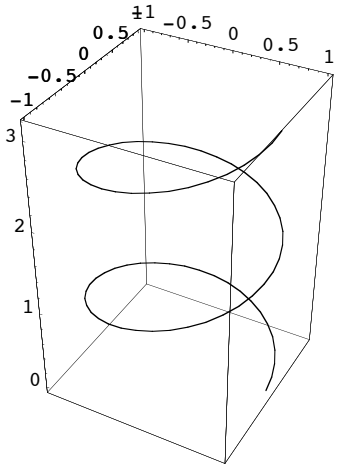
■ 媒介変数表示された空間曲線と曲面のプロット

媒介変数表示された空間曲線と曲面をプロットするには， ParametricPlot3D を用いる．

■ 媒介変数表示された空間曲線

● ParametricPlot3D[{x, y, z}, {t, tmin, tmax}] は， x, y, z を t の関数として，(x, y, z) を座標にもつような点の軌跡を， t = tmin~tmax においてプロットする．

```
In[35]:= ParametricPlot3D[{Cos[t], Sin[t], t / 4}, {t, 0, 4 Pi}]
```



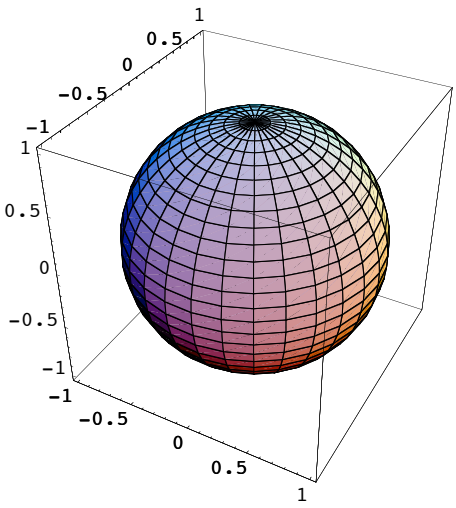
```
Out[35]= - Graphics3D -
```

■ 媒介変数表示された曲面

● ParametricPlot3D[{x, y, z}, {t, tmin, tmax}, {u, umin, umax}] は， x, y, z を t, u の関数として， (x, y, z) を座標にもつような点からなる曲面を， t = tmin~tmax, u = umin~umax においてプロットする．

球面上の点の座標を緯度 t $\left(-\frac{\pi}{2} \le t \le \frac{\pi}{2}\right)$ と経度 u $(0 \le u \le 2\pi)$ で表すと $(\cos(t)\cos(u),\ \cos(t)\sin(u),\ \sin(t))$ となる．これを用いて球面を表示する．

```
In[36]:= sphere = ParametricPlot3D[
           {Cos[t] Cos[u], Cos[t] Sin[u], Sin[t]}, {t, -Pi / 2, Pi / 2}, {u, 0, 2 Pi}]
```

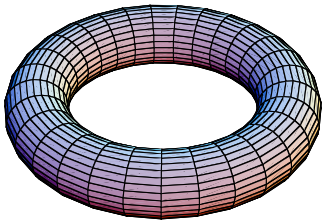


```
Out[36]= - Graphics3D -
```

ドーナツ面 (torus) はチューブの中心が描く円の半径を R チューブの半径を r とすると，
$(\cos(t)(R + r\cos(u)), \sin(t)(R + r\cos(u)), r\sin(u))$ で表される．

```
In[37]:= R = 2; r = 1 / 2;
```

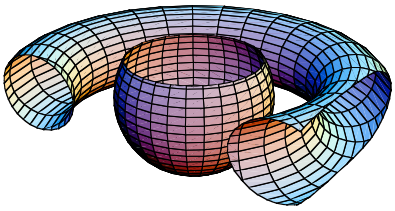
```
In[38]:= torus = ParametricPlot3D[{Cos[t] (R + r Cos[u]), Sin[t] (R + r Cos[u]), r Sin[u]},
           {t, 0, 2 Pi}, {u, 0, 2 Pi}, Axes -> False, Boxed -> False]
```



```
Out[38]= - Graphics3D -
```

Show を用いると，以前に計算したプロット図を重ねて表示したり，プロット範囲や視点などを変更したりできる．

```
In[39]:= Show[torus, sphere,
          PlotRange -> {{-3, 2}, {-1, 3}, {-1, 1 / 2}}, ViewPoint -> {1.3, -2, 1}]
```



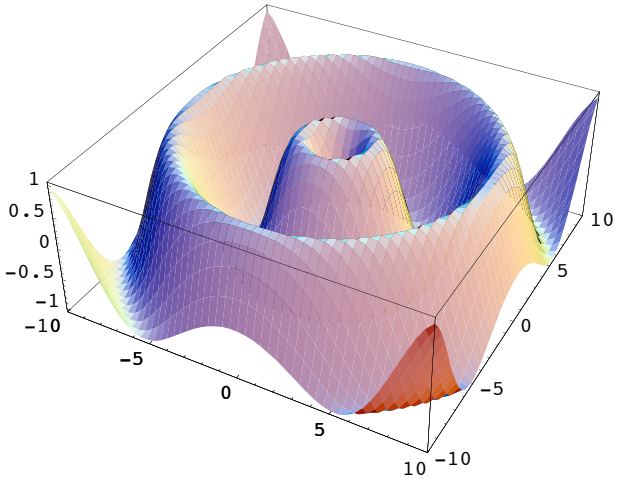
```
Out[39]= - Graphics3D -
```

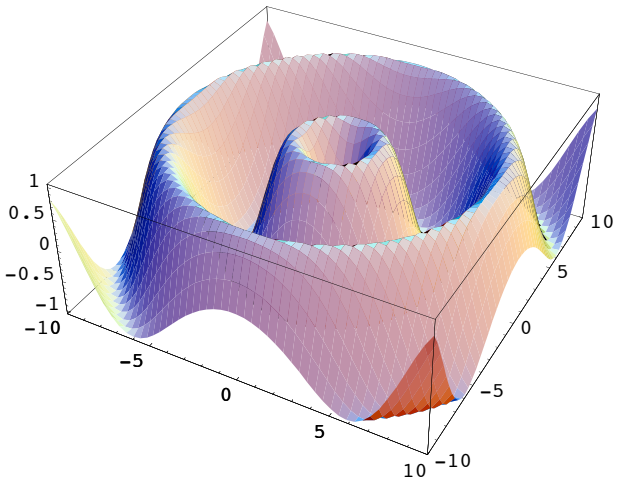
■ アニメーション

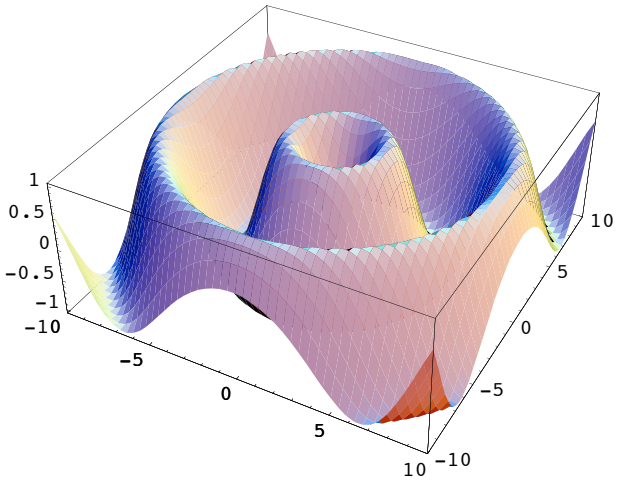
アニメーションを作成するには，次のステップで行う．
1．プロット図のリストを作成する．
2．表示されたグラフィックスの全体を囲むようにウインドウの右端にある] を選択して黒くする．
3．セル (Cell) メニューにある グラフィックスのアニメーション化 (Animate Selected Graphics) を選択する．

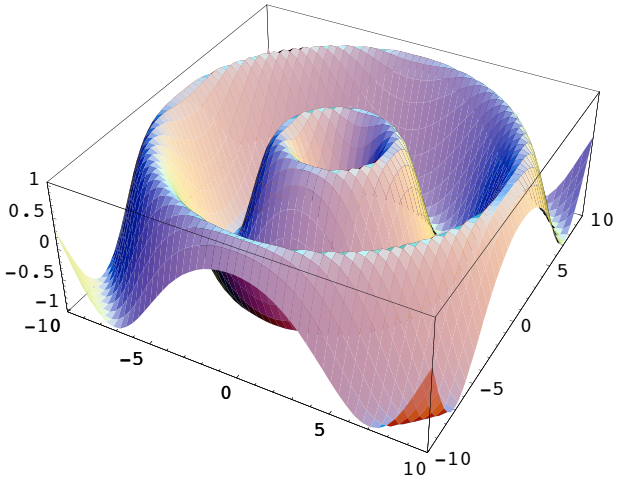
次の例でアニメーションを見てみよう．

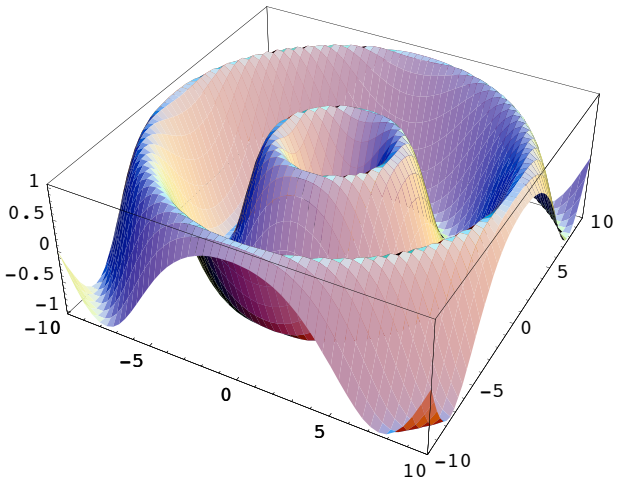
```
In[40]:= Table[Plot3D[Sin[Sqrt[x^2 + y^2] - t], {x, -10, 10},
           {y, -10, 10}, PlotPoints → 50, Mesh → False], {t, Pi / 10, 2 Pi, Pi / 10}]
```

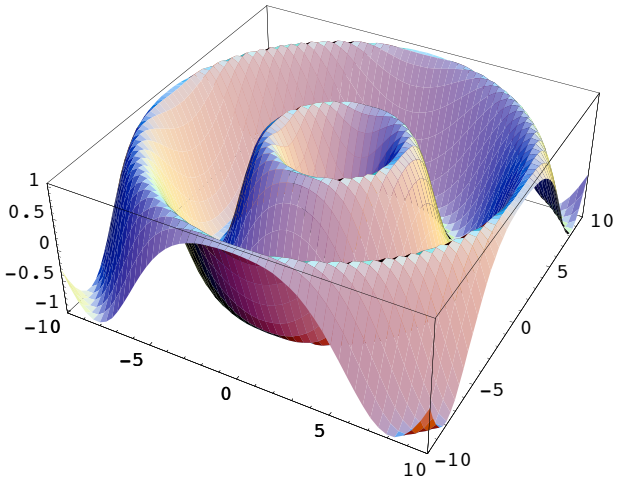

1
0.5
0
-0.5
-1
-10
-5
0
5
10
10
5
0
-5
-10

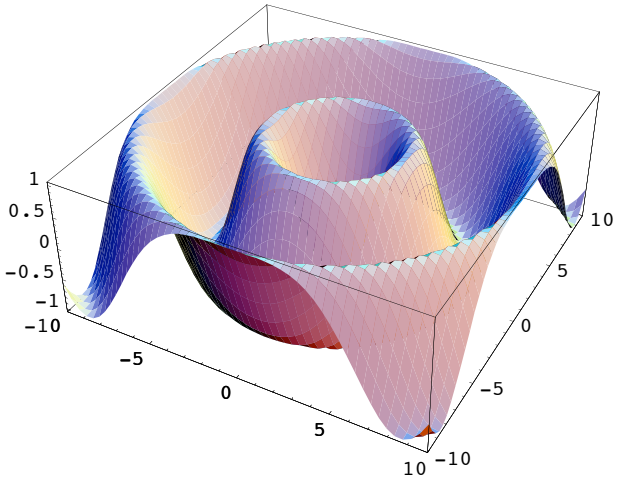

1
0.5
0
-0.5
-1
-10
-5
0
5
10
10
5
0
-5
-10

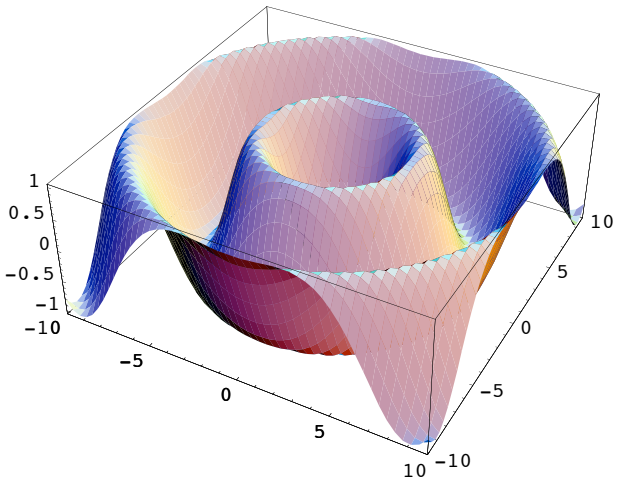

1
0.5
0
-0.5
-1
-10
-5
0
5
10
10
5
0
-5
-10

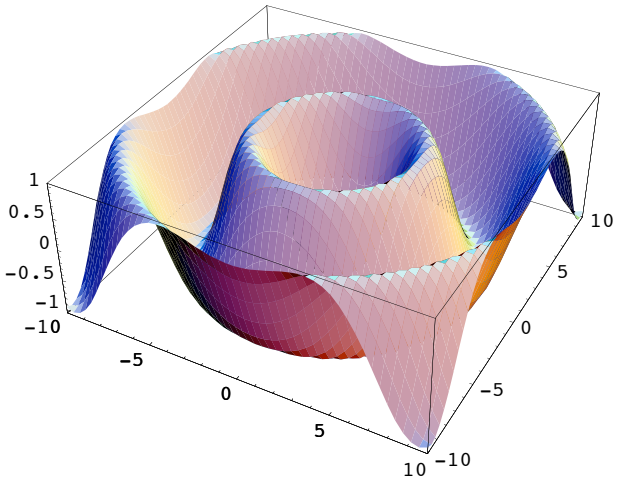

1
0.5
0
-0.5
-1
-10
-5
0
5
10
10
5
0
-5
-10

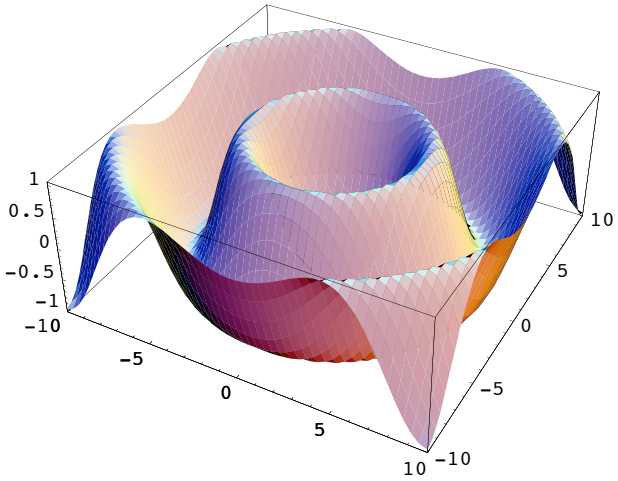

1
0.5
0
-0.5
-1
-10
-5
0
5
10
10
5
0
-5
-10

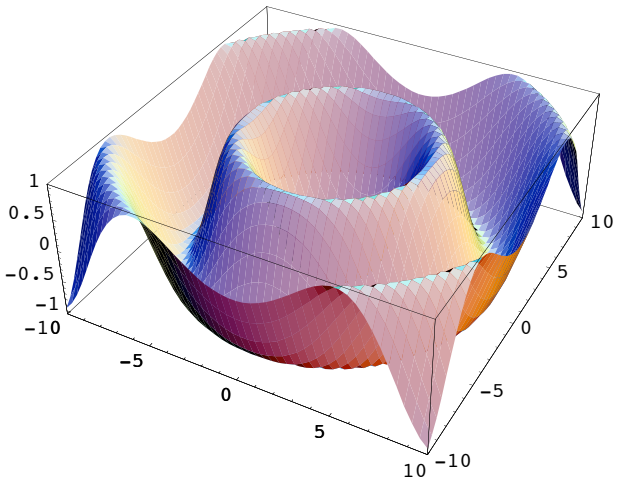

1
0.5
0
-0.5
-1
-10
-5
0
5
10
10
5
0
-5
-10

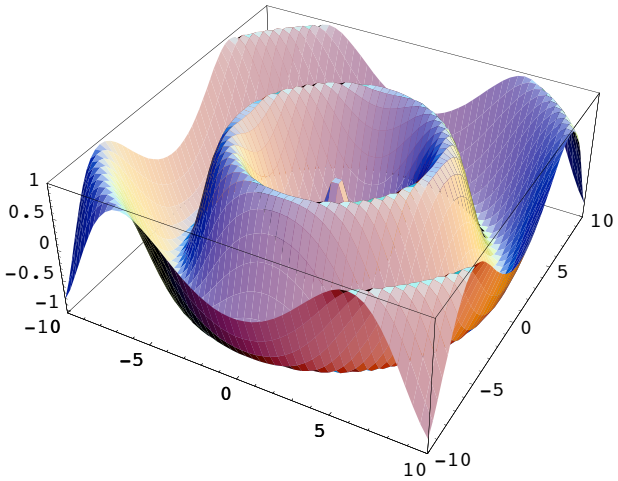

1
0.5
0
-0.5
-1
-10
-5
0
5
10
10
5
0
-5
-10

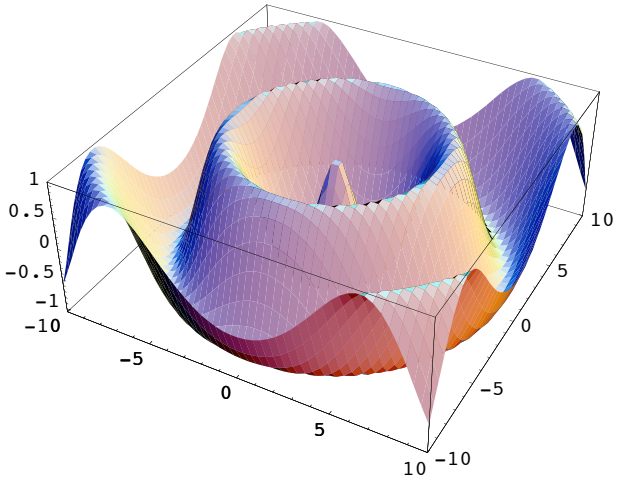

1
0.5
0
-0.5
-1
-10
-5
0
5
10
10
5
0
-5
-10

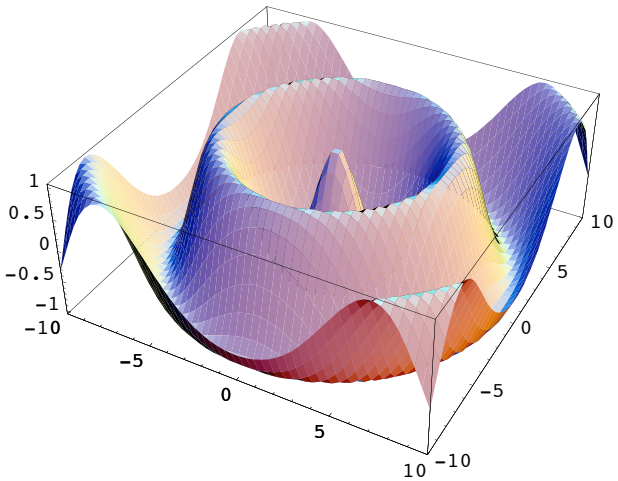

1
0.5
0
-0.5
-1
-10
-5
0
5
10
10
5
0
-5
-10

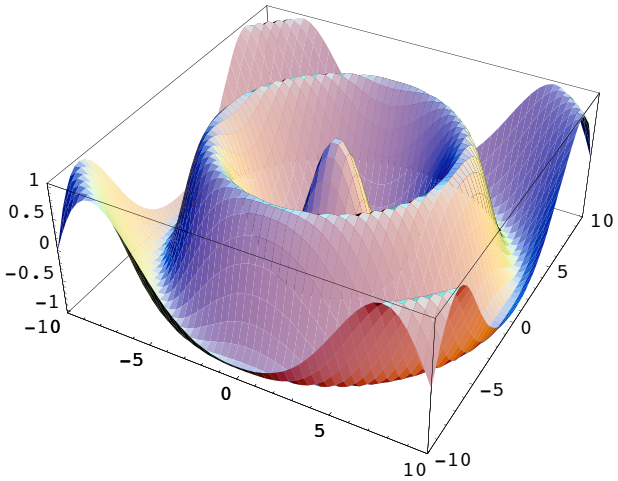

1
0.5
0
-0.5
-1
-10
-5
0
5
10
10
5
0
-5
-10

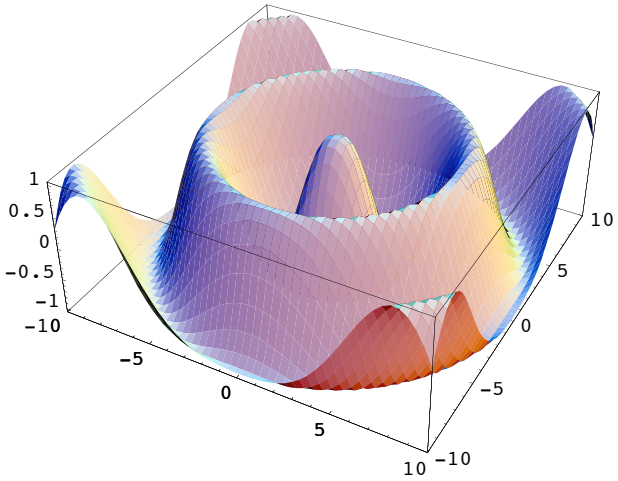

1
0.5
0
-0.5
-1
-10
-5
0
5
10
10
5
0
-5
-10

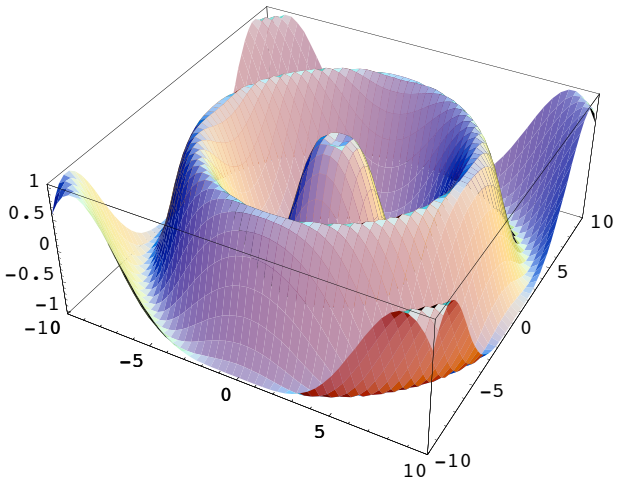

1
0.5
0
-0.5
-1
-10
-5
0
5
10
10
5
0
-5
-10

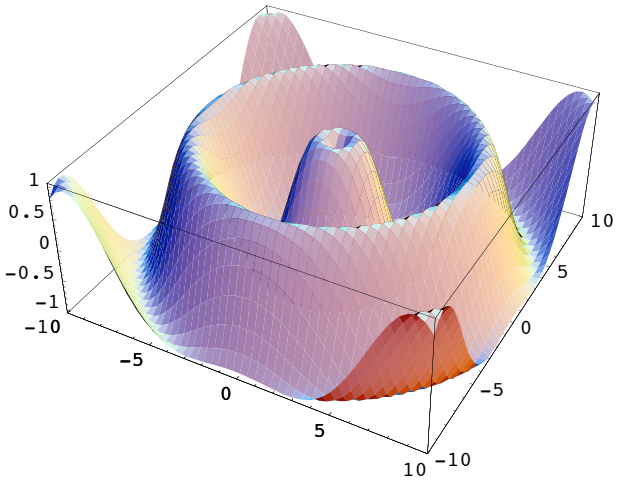

1
0.5
0
-0.5
-1
-10
-5
0
5
10
10
5
0
-5
-10

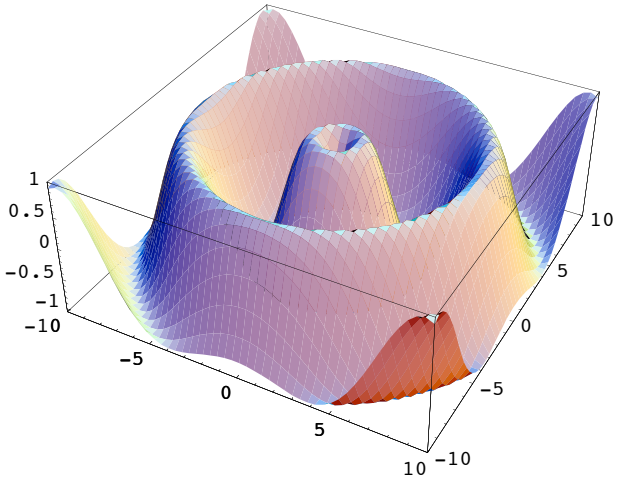

1
0.5
0
-0.5
-1
-10
-5
0
5
10
10
5
0
-5
-10

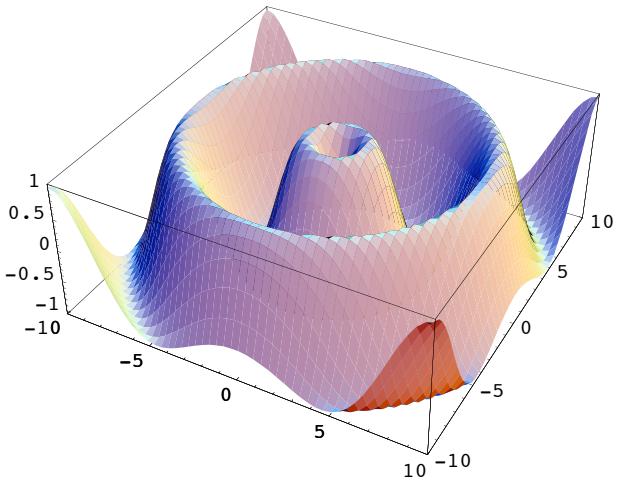

1
0.5
0
-0.5
-1
-10
-5
0
5
10
10
5
0
-5
-10


1
0.5
0
-0.5
-1
-10
-5
0
5
10
-10
-5
0
5
10


1
0.5
0
-0.5
-1
-10
-5
0
5
10
-10
-5
0
5
10


1
0.5
0
-0.5
-1
-10
-5
0
5
10
-10
-5
0
5
10





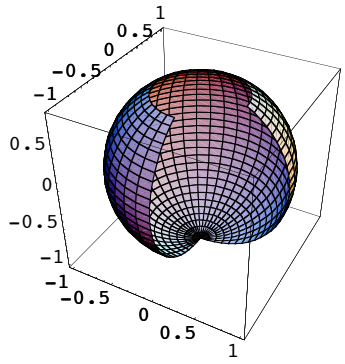

```
Out[40]= {- SurfaceGraphics -, - SurfaceGraphics -, - SurfaceGraphics -, - SurfaceGraphics -,
          - SurfaceGraphics -, - SurfaceGraphics -, - SurfaceGraphics -, - SurfaceGraphics -,
          - SurfaceGraphics -, - SurfaceGraphics -, - SurfaceGraphics -, - SurfaceGraphics -,
          - SurfaceGraphics -, - SurfaceGraphics -, - SurfaceGraphics -, - SurfaceGraphics -,
          - SurfaceGraphics -, - SurfaceGraphics -, - SurfaceGraphics -, - SurfaceGraphics -}
```

■ 演習問題

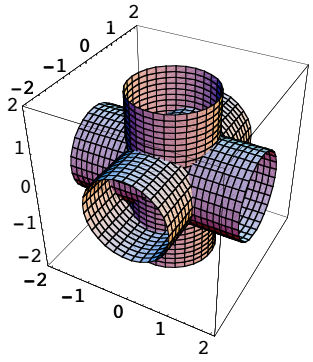
[10-1] $y = \sin(x) + x$ のグラフと $y=\frac{1}{6}x^2$ のグラフとを $-10 \le x \le 10$ の範囲で同一の画面上に描け．二つのグラフは原点以外に交点があるが，それはどのあたりにあるかをグラフから見当をつけよ．プロットする x の範囲をその交点の周辺に絞って，もう一度二つの関数のグラフを描き，交点の座標をグラフで見て小数点以下 1 桁の精度で答えよ．

[10-2] $y = x^3 - 2x^2 - 4x + 3$ のグラフを Plot で描き，(1, -2) を中心とする半径 2 の円を ParametricPlot で描き，それらを Show を用いて同一画面上に表示せよ．二つの曲線の交点は何個あるか．

[10-3] 次のような球面の一部をプロットせよ.



[10-4] 次のような，円筒が 3 本交わっている図を描け.



[10-5] 水面に物を落としたときの波紋のような，次のプロット図を作れ.