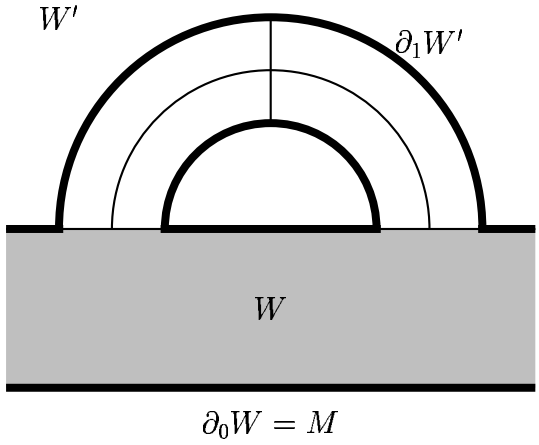
PSTricks 入門

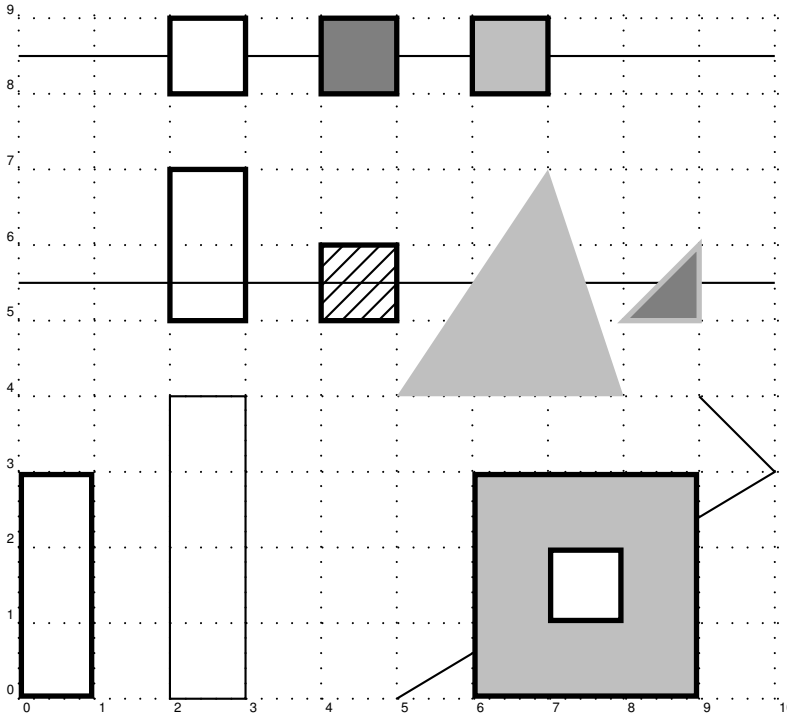
これは城西大学における 2004 年度の幾何学の講義で配布した資料をひとつにまとめたものです。LaTeX で図を描くためのパッケージ PSTricks の入門的事項がまとめてあります。作成には PSTricks マニュアルを参考にしました。多くの例はそこからとったものです。（山崎正之 `http://surgery.matrix.jp/`）



(pic01)

1　導入

まず下の線分や折れ線を用いた絵を作成してみます：



(square01)

1. 次ページのようなファイルを作成する：square01.tex
2. 次ページのようなバッチファイル (pstricks.bat) をコマンドプロンプト上で実行する：
   `pstricks square01`
3. eps ファイル square01.eps ができる。

[square01.tex]:

```
\documentclass{article}
\usepackage{pstricks}%
\usepackage{pst-eps}%
\usepackage{pst-plot}%
\usepackage{pst-node}%
\begin{document}

\TeXtoEPS   % Make an EPS picture out of .dvi file, with dvips -E.
\psset{unit=10mm}
\pspicture(-0.2,-0.2)(10.2,9.2)
\psgrid[subgriddiv=1,griddots=5,gridlabels=5pt](0,0)(10,9)

\psframe[linewidth=2pt](0,0)(1,3)

\psline(0,8.5)(10,8.5)
\psline(0,5.5)(10,5.5)
\psline(5,0)(10,3)(9,4)

\pspolygon[fillstyle=none](2,0)(3,0)(3,4)(2,4)
\pspolygon[fillstyle=none, linewidth=2pt](2,5)(3,5)(3,7)(2,7)
\pspolygon[fillstyle=solid, linewidth=2pt](2,8)(3,8)(3,9)(2,9)
\pspolygon[fillstyle=solid, fillcolor=gray, linewidth=2pt](4,8)(5,8)(5,9)(4,9)
\pspolygon[fillstyle=solid, fillcolor=lightgray, linewidth=2pt](6,8)(7,8)(7,9)(6,9)
\pspolygon[fillstyle=hlines, linewidth=2pt](4,5)(5,5)(5,6)(4,6)

\psframe[linewidth=2pt, fillstyle=solid, fillcolor=lightgray](6,0)(9,3)
\psframe[linewidth=2pt, fillstyle=solid, fillcolor=white](7,1)(8,2)

\pspolygon[linestyle=none, fillstyle=solid, fillcolor=lightgray](5,4)(8,4)(7,7)
\pspolygon[linecolor=lightgray, linewidth=2pt, fillstyle=solid, fillcolor=gray](8,5)(9,5)(9,6)

\endpspicture
\endTeXtoEPS

\end{document}
```

[pstricks.bat]:

```
platex %1
dvipsk -D600 -E -o %1.eps %1.dvi
```

作成した eps ファイルは、あらかじめ

```
\usepackage[dvips]{graphicx}
```

としておけば

```
\includegraphics{square01.eps}
```

のようにして LaTeX 文書の中で使用できます。

2　基礎的事項

●色の名前

- グレースケール：black, darkgray, gray, lightgray, white, ……
- カラー：red, green, blue, cyan, magenta, yellow

●パラメータの指定

- \psline[fillcolor=gray, linecolor=red](1,0)(3,4) のようにオプションで指定する。
- 同じ指定を多数のコマンドに共通に使うには \psset{fillcolor=gray, linecolor=red} のようにまとめて指定できる。なお、グループ{　} の中で指定した場合、有効範囲はそこに制限される。

●単位の指定

- 長さの単位は直接 2pt などのように書き込む。省略してある場合は cm とみなされる。これを変更するには \psset{unit=7mm} などのように書いておく。
- 角度は通常「度」を単位として表す。ラジアンを使いたい場合は \radians とする。また度になおすには \degrees とする。1 周の 100 分の 1 の角度を単位にしたいなどというときは\degrees[100] とする。

●線や塗りつぶしの主なパラメータ

パラメータ	意味	デフォルト
linestyle=スタイル	線のスタイル	solid
	none, solid, dashed, dotted	
linewidth=太さ	線の太さ	0.8pt
linecolor=色	線の色	black
border=長さ	線の縁どりの幅	0pt
bordercolor=色	線の縁取りの色	white
arrows=スタイル	矢のスタイル	-
	-, ->, <-, <->, |-|, [-], (-), o-o, *-*, ……	
linearc=半径	折れ線の角の丸み半径	0pt
framearc=数	frame の角の丸み半径の短辺との比率 (0〜1) の 2 倍	0pt
showpoints=真偽	角の点の表示 true, false	false
dimen=外内中	閉曲線・折れ線の幅 (太さ) の付け方	outer
	outer, inner, middle	
fillstyle=スタイル	塗りつぶしのスタイル	none
	none, solid, vlines, hlines, ……	
fillcolor=色	塗りつぶしの色	white

●直線・線分・折れ線・多角形

- \psline[パラメータ]{矢}(x0,y0)(x1,y1)……(xn,yn) 折れ線
- \pspolygon[パラメータ](x0,y0)(x1,y1)……(xn,yn) 閉折れ線
- \psframe[パラメータ](x0,y0)(x1,y1) フレーム

●弧・円・楕円

- \pscircle[パラメータ](x0,y0){半径} 円
- \qdisk(x0,y0){半径} 円板
- \pswedge[パラメータ](x0,y0){半径}{角 1}{角 2} 扇形
- \psellipse[パラメータ](x0,y0)(x1,y1) 楕円, 中心：(x0,y0), x1：x 軸方向の半径, y1：y 軸方向の半径
- \psarc[パラメータ]{矢}(x,y){半径}{角 A}{角 B} 反時計方向の円弧
  arcsepA=幅　　中心から角 A 方向に延びる半直線と接するように角 A を調節, デフォルトは 0pt
  arcsepB=幅　　中心から角 B 方向に延びる半直線と接するように角 B を調節, デフォルトは 0pt
  arcsep=幅　　角 A, 角 B 両方を調節, デフォルトは 0pt
- \psarcn[パラメータ]{矢}(x,y){半径}{角 A}{角 B} 時計方向の円弧

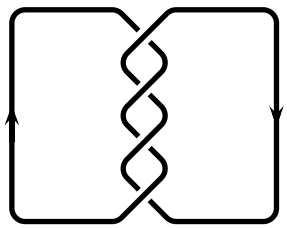
●曲線

- \psbezier[パラメータ]{矢}(x0,y0)(x1,y1)(x2,y2)(x3,y3) ベジエ曲線
- \parabola[パラメータ]{矢}(x0,y0)(x1,y1) 放物線, 始点 (x0,y0), 頂点 (x1,y1)
- \pscurve[パラメータ]{矢}(x1,y1)……(xn,yn) 点をつなぐ曲線
- \psecurve[パラメータ]{矢}(x1,y1)……(xn,yn) 点をつなぐ曲線 (最初と最後の点はむすばない)
- \psccurve[パラメータ]{矢}(x1,y1)……(xn,yn) 点をつなぐ閉曲線

●その他

- \psdots[パラメータ](x1,y1)(x2,y2)……(xn,yn) 点を指定位置に並べる
  dotstyle=スタイル, スタイル例：* (デフォルト), o, +, triangle, triangle*, square, square*, など
- \psgrid(x0,y0)(x1,y1)(x2,y2) 格子, 目盛りをつける中心, (x1,y1)(x2,y2) 格子の範囲

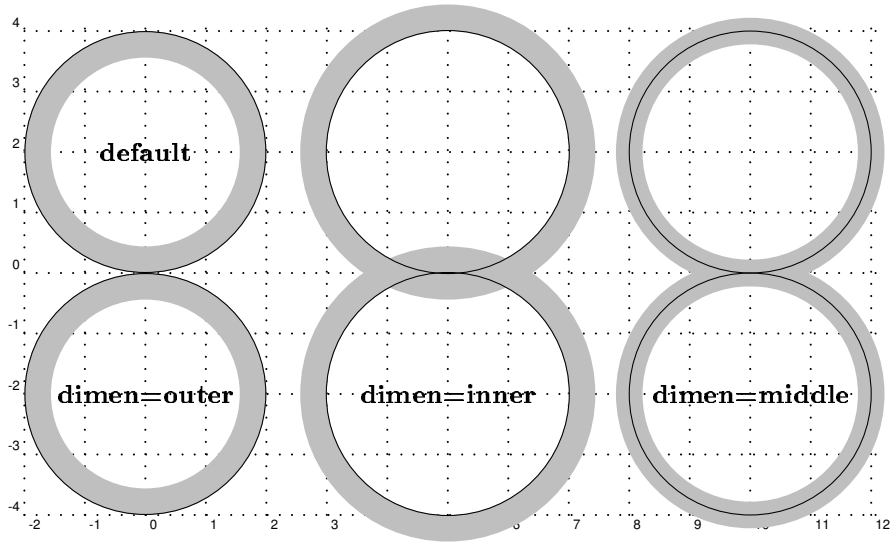
サンプル１：



(knot01)

```
\pspicture(-0.2,-0.2)(5.2,4.2)
\psset{linewidth=2pt, border=2pt, bordercolor=white, linearc=5pt}
\psline{->}(0,1.5)(0,4)(2,4)(3,3)(2,2)(3,1)(2,0)(0,0)(0,2.2)
\psline(5,2)(5,0)(3,0)(2.6,0.4)
\psline(2.4,0.6)(2,1)(3,2)(2.6,2.4)
\psline{->}(2.4,2.6)(2,3)(3,4)(5,4)(5,1.8)
```

サンプル２（dimen パラメータの指定）：

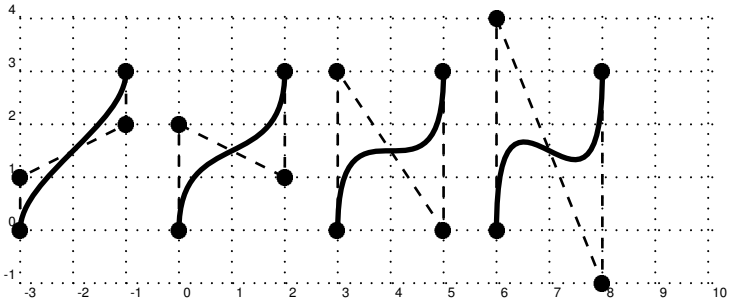


(dimen01)

```
\psgrid[subgriddiv=1,griddots=5,gridlabels=5pt](-2,-4)(12,4)

\pscircle[dimen=outer, linecolor=lightgray](0,-2){2}
\pscircle[dimen=outer, linecolor=lightgray](0,2){2}
\pscircle[dimen=outer, linecolor=black, linewidth=0.4pt](0,-2){2}
\pscircle[dimen=outer, linecolor=black, linewidth=0.4pt](0,2){2}
\rput[c]{0}(0,-2){\bf dimen=outer}
\rput[c]{0}(0,2){\bf default}


\pscircle[dimen=inner, linecolor=lightgray](5,-2){2}
\pscircle[dimen=inner, linecolor=lightgray](5,2){2}
\pscircle[dimen=inner, linecolor=black, linewidth=0.4pt](5,-2){2}
\pscircle[dimen=inner, linecolor=black, linewidth=0.4pt](5,2){2}
\rput[c]{0}(5,-2){\bf dimen=inner}


\pscircle[dimen=middle, linecolor=lightgray](10,-2){2}
\pscircle[dimen=middle, linecolor=lightgray](10,2){2}
\pscircle[dimen=middle, linecolor=black, linewidth=0.4pt](10,-2){2}
\pscircle[dimen=middle, linecolor=black, linewidth=0.4pt](10,2){2}
\rput[c]{0}(10,-2){\bf dimen=middle}
```

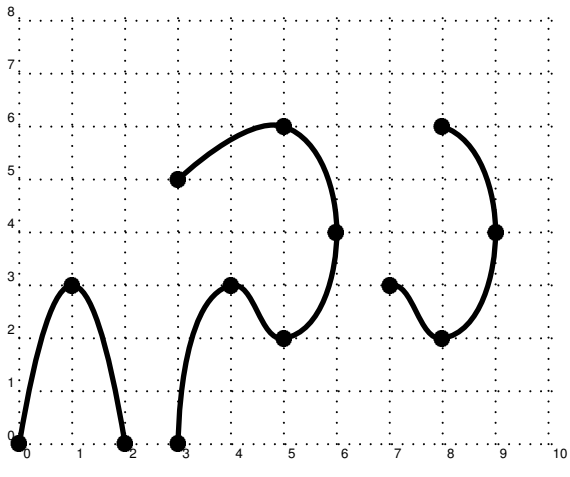
3　曲線・ベジエ曲線

```
\psset{linewidth=2pt, showpoints=true}
\psbezier(-3,0)(-3,1)(-1,2)(-1,3)
\psbezier(0,0)(0,2)(2,1)(2,3)
\psbezier(3,0)(3,3)(5,0)(5,3)
\psbezier(6,0)(6,4)(8,-1)(8,3)
```



(bezier)

```
\psset{linewidth=2pt, showpoints=true}
\parabola(0,0)(1,3)
\pscurve(3,0)(4,3)(5,2)(6,4)(5,6)(3,5)
\psecurve(6,0)(7,3)(8,2)(9,4)(8,6)(6,5)
```

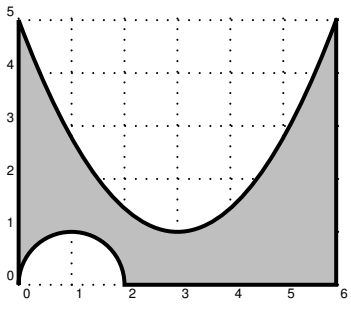


(curve01)

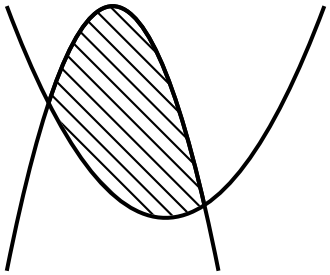
4 pscustom (1)

いくつかの種類の曲線・折れ線をつなげてひとつの図形を使いたい場合もあります。そのときは **pscustom** を用います。始点と終点を指定するタイプの図形の場合は、前の図形の終点が自動的に始点になりますから、始点を省略できます。そうでない場合は、前の図形の終点と次の図形の始点が線分で結ばれます（この動作は変えることができます)。例えば下のように用いると放物線、折れ線、円弧、線分がひとつの図形を作ります（下図左)。これを用いれば、下図右のような絵を作図できます。

```
\pscustom[linewidth=1.5pt, fillstyle=solid, fillcolor=lightgray]{ %% このオプションは有効
  \parabola[fillstyle=vlines, linewidth=3pt](0,5)(3,1) %% こちらのオプションは無視される
  \psline(6,0)(2,0) %% (2,0) は省略可能
  \psarc(1,0){1}{0}{180}
  \psline(0,5)}
```



(testcustom01)

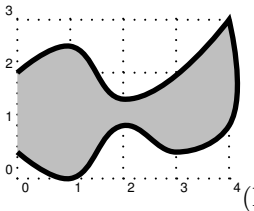

(testcustom02)

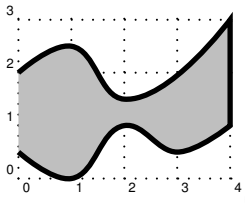
直前の図形の終点をどう扱うかは、次の図形のオプションの中で **liftpen** の値を 0, 1, 2 の中から選んで次のように指定できます。

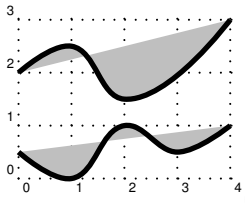
- **liftpen=0** と指定：通常の動作です。直前の図形の終点は次の図形の始点になります。
- **liftpen=1** と指定：直前の図形の終点は次の図形の始点とはなりません。その 2 点は線分で結ばれます。
- **liftpen=2** と指定：直前の図形の終点は次の図形の始点とはなりません。その 2 点は線分で結ばれません。

例えば次のように記述すると下図の左のようになります。liftpen の値を 1 や 2 に変えると下図中、下図右のようになります。

```
\pscustom[linewidth=2pt, fillstyle=solid, fillcolor=lightgray]{
  \pscurve(0,2)(1,2.5)(2,1.5)(4,3)
  \pscurve[liftpen=0](4,1)(3,0.5)(2,1)(1,0)(0,0.5)}
```


(liftpen0)

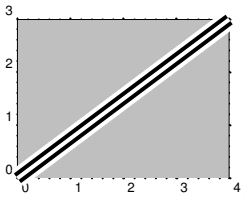

(liftpen1)


(liftpen2)

5 pscustom (2)

pscustom の続きです。以下では曲線・線分・折れ線をひっくるめて単に曲線と呼びます。
\pscustom では、その中で作成された曲線の描画や塗りつぶしは一番最後に行われます。一方、\pscustom コマンド中のある時点において、一度曲線の描画や塗りつぶしを実行したいときは、それぞれ\stroke, \fill コマンドを使います。このときオプションで別の色などを指定することができます。

```
\psline[linewidth=0,fillstyle=solid,fillcolor=lightgray](0,0)(0,3)(4,3)(4,0)
\pscustom[linecolor=white,linewidth=1.5pt]{%
  \psline(0,0)(4,3)
  \stroke[linewidth=5\pslinewidth]
  \stroke[linewidth=3\pslinewidth, linecolor=black]}
```

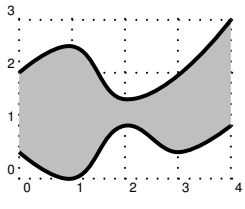


(testcustom03)

上の例で、描画の順がどうなっているのか図を見てチェックしましょう。

\stroke や \fill は、\gsave, \grestore と組み合わせて用いると効果的です。\gsave はその時点でのグラフィックの状態を保存するコマンドです。\grestore はそれを復活させるコマンドです。必ずこの順でペアにして使います。\gsave, \grestore の中間に指定された曲線はなかったものとみなされてしまいます。入れ子にして用いることもできます。

```
\pscustom[linewidth=1.5pt]{
  \pscurve(0,2)(1,2.5)(2,1.5)(4,3)
  \gsave
    \pscurve[liftpen=1](4,1)(3,0.5)(2,1)(1,0)(0,0.5)
    \fill[fillstyle=solid,fillcolor=lightgray]
  \grestore}
\pscurve[linewidth=1.5pt](4,1)(3,0.5)(2,1)(1,0)(0,0.5)
```

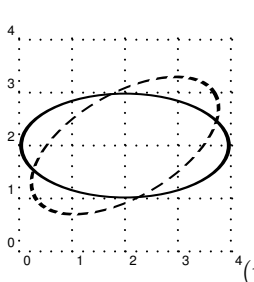


(testcustom04)

座標軸を様々に変換することができます。

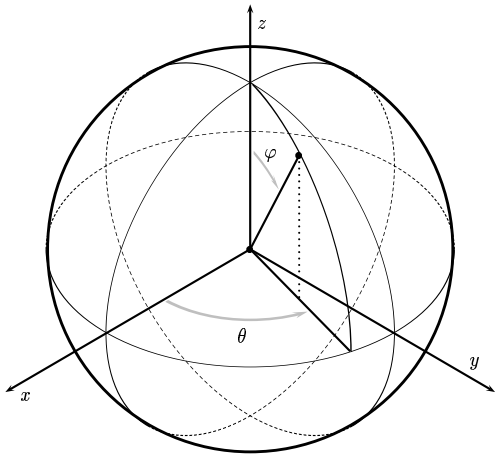
- 平行移動は\translate(x,y) の形で指定します。
- 回転は\rotate{角}の形で指定します。
- 軸の目盛りの変換は\scale{比}または\scale{x 方向の比 y 方向の比}の形で指定します。

```
\pscustom[linewidth=1.5pt]{
  \translate(2,2)
  \scale{1 0.5}
  \pscircle(0,0){2}}
\pscustom[linewidth=1.5pt, linestyle=dashed]{
  \translate(2,2)
  \rotate{30}
  \scale{1 0.5}
  \pscircle(0,0){2}}
```



(testcustom05)

[問] 次のような図を描いてみましょう。文字は`\rput[c]{0}(-5.5,-3.6){$x$}`のようにして出力できます。



(sphere2)

解答例：

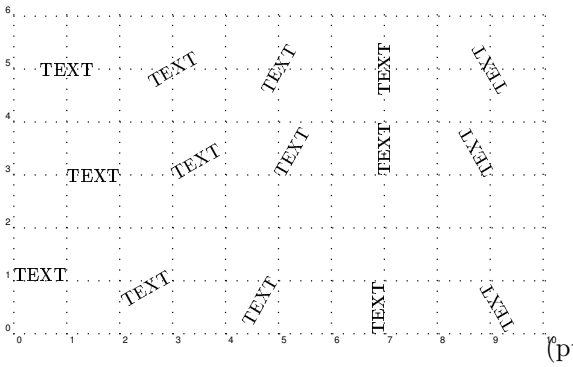
```
\documentclass{article}
\usepackage{pstricks}%
\usepackage{pst-eps}%
\usepackage{pst-plot}%
\usepackage{pst-node}%
\begin{document}

\TeXtoEPS\psset{unit=10mm}
\pspicture(-6,-6)(6,6)
\pscircle[linewidth=2pt](0,0){5}

\pscustom{%
\scale{1 0.577}
\psarc(0,0){5}{180}{360}
}

\pscustom[linestyle=dashed, dash=3pt 2pt]{%
\scale{1 0.577}
\psarc(0,0){5}{0}{180}
}

\pscustom{%
\rotate{120}
\scale{1 0.577}
\psarc(0,0){5}{180}{360}
}

\pscustom[linestyle=dashed, dash=3pt 2pt]{%
\rotate{120}
\scale{1 0.577}
```

```
\psarc(0,0){5}{0}{180}
}

\pscustom{%
\rotate{-120}
\scale{1 0.577}
\psarc(0,0){5}{180}{360}
}

\pscustom[linestyle=dashed, dash=3pt 2pt]{%
\rotate{-120}
\scale{1 0.577}
\psarc(0,0){5}{0}{180}
}

\pscustom[linewidth=2pt]{%
\rotate{108}
\scale{1 0.405}
\psarc(0,0){5}{231}{320}
}

\psline[linewidth=1.5pt]{<->}(0,6)(0,0)(-6,-3.5)
\psline[linewidth=1.5pt]{->}(0,0)(6,-3.5)
\psline[linewidth=1.5pt](0,0)(2.45,-2.5)
\psline[linewidth=1.5pt, linestyle=dotted](1.2,-1.2)(1.2,2.3)
\psline[linewidth=1.5pt, showpoints=true](0,0)(1.2,2.3)

\pscustom[linewidth=3pt, linecolor=lightgray]{%
\scale{1 0.577}
\psarc{->}(0,0){3}{227}{298}
}

\pscustom[linewidth=3pt, linecolor=lightgray]{%
\rotate{108}
\scale{1 0.405}
\psarc{<-}(0,0){3}{293}{318}
}

\rput[c]{0}(-0.2,-2.1){\Large$\theta$}
\rput[c]{0}(0.5,2.3){\Large$\varphi$}
\rput[c]{0}(-5.5,-3.6){\Large$x$}
\rput[c]{0}(5.5,-2.8){\Large$y$}
\rput[c]{0}(0.3,5.5){\Large$z$}

\endpspicture
\endTeXtoEPS
\end{document}
```

6 rput コマンド

前回、文字出力に\rput コマンドを用いました。今回はもう少し詳しく説明します。まず下の例をみて下さい。

```
\rput[c]{0}(1,5){TEXT}
\rput[c]{30}(3,5){TEXT}
\rput[c]{60}(5,5){TEXT}
\rput[c]{90}(7,5){TEXT}
\rput[c]{120}(9,5){TEXT}

\rput[l]{0}(1,3){TEXT}
\rput[l]{30}(3,3){TEXT}
\rput[l]{60}(5,3){TEXT}
\rput[l]{90}(7,3){TEXT}
\rput[l]{120}(9,3){TEXT}

\rput[rb]{0}(1,1){TEXT}
\rput[rb]{30}(3,1){TEXT}
\rput[rb]{60}(5,1){TEXT}
\rput[rb]{90}(7,1){TEXT}
\rput[rb]{120}(9,1){TEXT}
```



(put01)

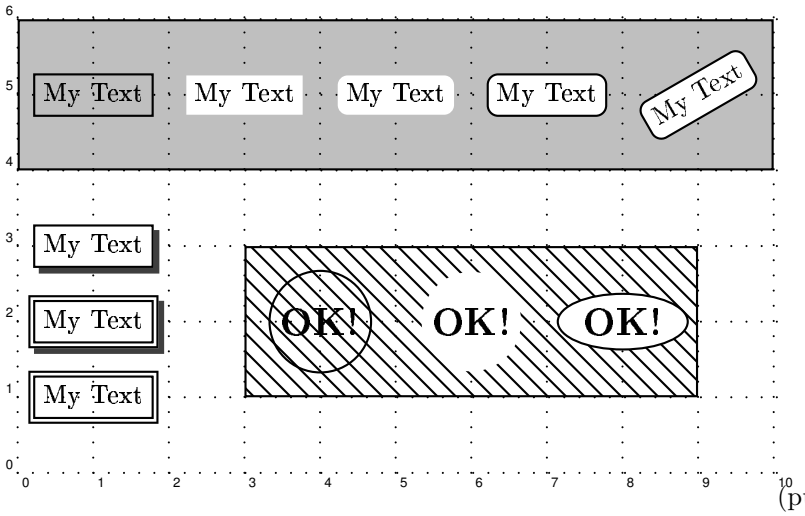
最初の [ ] の中で、表示するものの**基準点**を指定します。この点が指定した座標にくるように出力されます。また回転を指定するときはここが回転の中心になります。次のような指定ができます。

- 中央：c　(デフォルト)
- 水平方向……l：左端、r：右端
- 垂直方向……t：上端、b：下端、B：ベースライン

続く{ }の中で、基準点を中心に回転する角度を指定します。
続いて基準点を出力する位置を直交座標で指定します。
最後の{ }の中に出力するものを記述します。

文字等を長方形 (フレーム) で囲むには\psframebox、二重の長方形 (フレーム) で囲むには\psdblframebox、円で囲むには\pscirclebox、だ円で囲むには\psovalbox を使います。

```
\psframe[fillstyle=solid, fillcolor=lightgray](0,4)(10,6)
\psframe[fillstyle=vlines](3,1)(9,3)
\psgrid[subgriddiv=1,griddots=5,gridlabels=5pt](0,0)(10,6)
\rput(1,5){\psframebox{My Text}}
\rput(3,5){\psframebox*{My Text}}
\rput(5,5){\psframebox*[fillcolor=white, framearc=.5]{My Text}}
\rput(7,5){\psframebox[fillstyle=solid,fillcolor=white,framearc=.5]{My Text}}
\rput{30}(9,5){\psframebox[fillstyle=solid,fillcolor=white,framearc=.5]{My Text}}
\rput(1,3){\psframebox[shadow=true, shadowsize=3pt, shadowangle=-45]{My Text}}
\rput(1,2){\psdblframebox[shadow=true, shadowsize=3pt, shadowangle=-45]{My Text}}
\rput(1,1){\psdblframebox{My Text}}
\rput(4,2){\pscirclebox{\bf\Large OK!}}
\rput(6,2){\pscirclebox*{\bf\Large OK!}}
\rput(8,2){\psovalbox[fillstyle=solid,fillcolor=white]{\bf\Large OK!}}
```



(put02)

7　繰り返し

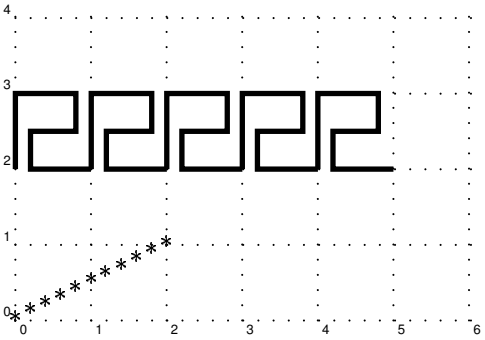
●同じものを線分に沿って等間隔に並べるには\multirput を用います：

\multirput[基準点]{角度}(x0,y0)(x1,y1){繰り返しの回数}{並べるもの}

(x0,y0) が最初の出力を行う位置、(x1,y1) がずらすベクトルを表します。

例：

```
\multirput(0,0)(0.2,0.1){11}{*}
\def\uzu{\psline(0,0)(0,1)(0.8,1)(0.8,0.5)(0.2,0.5)(0.2,0)(1,0)}%
\psset{linewidth=2pt}
\multirput(0,2)(1,0){5}{\uzu}
```


(multi01)

\uzu の定義の行末の%を省略すると、出力位置がずれるので注意しましょう。

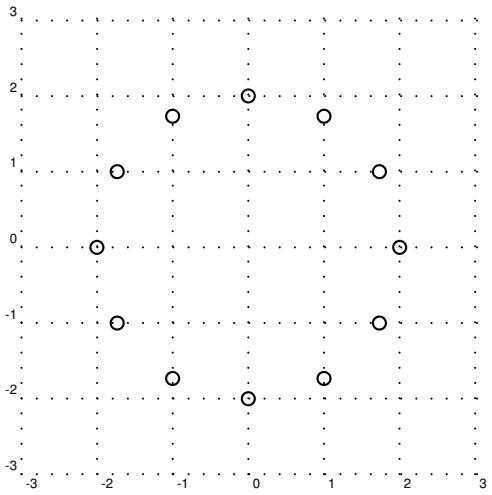
●それ以外の場合には\multido を使って工夫しましょう。

\multido{\var1=初期値+増分,\var2=初期値+増分, ...}{回数}{繰り返すグラフィックの指定}

- ループ用の変数名は、n で始める。整数値なら i で始めるとやや速くなる。
- 増分が負の時は \n=0+-1 のように指定する。

例：

```
\multido{\n=0+30}{12}{%
  \rput{\n}{\rput(2,0){\pscircle{0.1}}}}
```



(multi02)

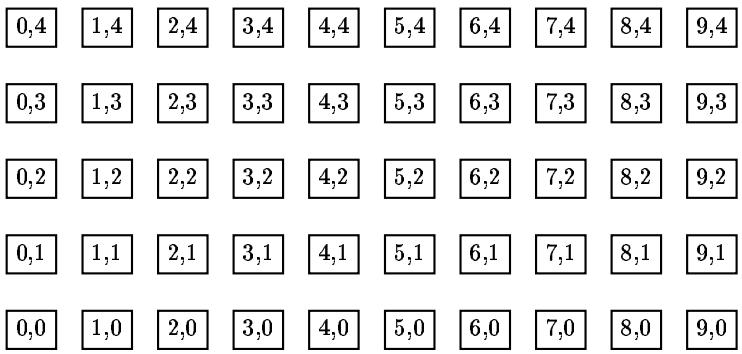
例：

```
\multido{\i=0+1,\n=0+.1}{11}{%
    \newgray{mygray}{\n}
    \psset{fillstyle=solid,fillcolor=mygray,linewidth=1pt}
    \rput(\i,0){\psframe(0,0)(0.8,0.8)}}
```

(multi05)

例：

```
\pspicture(-0.4,-0.4)(10.4,5.4)
\multido{\ix=0+1}{10}{%
    \multido{\iy=0+1}{5}{%
        \rput(\ix,\iy){\psframebox{\small \ix,\iy}}}}
```



(multi06)

8　外部データの利用によるプロット

今回は他プログラムで作成したデータに基づいてグラフを描画する方法を学びます。

●まず、他のプログラムを用いて描画したいグラフの上の点のデータを作成します。例えば次のような Perl スクリプトを plot01.pl という名前で作成します。これは、区間 $0 \leqq x \leqq \pi/2$ を 200 等分した点で、関数 $f(x) = \sin(x)\cos(10x^2)$ の値を計算し、得られる点の座標を順に出力するスクリプトです。

```
$Pi   = 3.141592;
$unit = $Pi/400;
for($i=0; $i<=200; $i++){
  $x = $unit * $i;
  $y = sin($x)*cos(10*($x**2));
  print "($x, $y) \n";
}
```

これをコマンドプロンプト上で、次のように実行してデータファイル plot01.data を作成します。

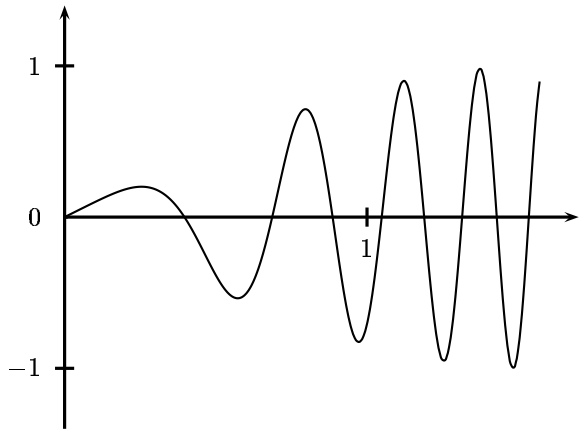
```
perl plot01.pl > plot01.data
```

このデータファイルの中身は次のようになっています：

```
(0, 0)
(0.00785398, 0.00785389776056747)
(0.01570796, 0.0157072662307394)
途中省略
(1.55508804, 0.581812109628891)
(1.56294202, 0.761676409594291)
(1.570796, 0.896610942591696)
```

●上で作成したデータを使用するには次のように記述します。

```
\psset{xunit=40mm, yunit=20mm}
\pspicture(-0.6,-1.4)(1.8,1.4)
\readdata{\mydata}{plot01.data}   %%% データの読み込み
\dataplot[plotstyle=curve]{\mydata}   %%% データのプロット
\psaxes[linewidth=1.2pt]{->}(0,0)(0,-1.4)(1.7,1.4)
\endpspicture
```



(plot01)