

# eps 形式による図形描画マクロ

emathPs.sty ver.0.93

## 使用例

tDB

2005/10/15

### 概 要

従来, emath では, 図形描画に `tpic specials` を用いてきました。それを EPS 形式のファイルを作成してそれを利用するやり方を模索して行きます。

現時点ではプロトタイプで, 仕様の変更などが頻繁に行われますので, そのおつもりでお付き合い願います。

このマクロ集のマクロについてのご質問, バグ報告, 修正・追加の提案等は

<http://emath.s40.xrea.com/>

の掲示板へどうぞ。



## 目 次

1	新設コマンド・環境	1
1.1	pszahyou 環境	1
1.2	pszahyou*環境	1
1.3	¥setlinewidth	2
1.4	¥setdash	2
1.5	¥setarrowsize	3
1.6	クリッピング	5
1.6.1	クリッピング	5
1.6.2	[borderwidth=..] オプション	6
1.6.3	¥truexmax など	9
1.6.4	掲示板から	10
1.7	図と文字のずれ	11
1.8	図形の回転	12
1.9	カラー化	14
1.9.1	EMpscolor 環境	14
1.9.2	¥defineEMpscolor コマンド	14
1.9.3	¥color との併用	15
1.10	EMpsrectbox 環境	16
1.10.1	EMpsrectbox とは	16
1.10.2	罫線と本文との間隔	17
1.10.3	横幅	18
1.10.4	見出しをつける	18
1.10.5	罫線の種類変更	19
1.10.6	罫線の太さ変更	20
1.10.7	rectboxoval オプション	20
1.10.8	mawarikomi 環境内の囲み	21
1.10.9	rectbox 環境の併用	25
1.11	EPSfilename=.. オプション	26
1.12	下線	27
1.12.1	¥pskasen	27
1.12.2	¥underline との比較	27
1.12.3	数式モードでは	27
1.12.4	下線の太さ	27
1.12.5	下線を破線で	28
1.12.6	二重下線	28
1.12.7	下線に色	28
1.12.8	下線上下の間隔	28
1.12.9	左右のアキ	29
1.12.10	下線近辺に文字配置	29
1.12.11	¥pskasen の書式	31
1.13	波下線	31



1.13.1	比較	31
1.13.2	各種 option	31
1.13.3	波線の形状	32
1.14	¥ovalbox	33
1.14.1	fancybox.sty & eepic.sty	33
1.14.2	¥emovalbox	33
1.14.3	¥psovalbox	34
1.14.4	¥psovalbox の書式	35
<b>2</b>	<b>対応コマンド</b>	<b>36</b>
2.1	¥Drawline	36
2.1.1	<linewidth=..>オプション	37
2.1.2	<dash=..>オプション	37
2.1.3	<iro=..>オプション	38
2.2	¥Takakkei	40
2.3	¥zahyouMemori	41
2.4	¥Put の syaei=.. オプション	42
2.5	¥ArrowLine	43
2.6	¥Kuromaru, ¥Siromaru	44
2.7	¥Nuritubusi	44
2.8	¥En	45
2.9	¥Enko	45
2.10	¥ougigata	46
2.11	¥yumigata	46
2.12	¥Daen	47
2.13	¥Daenko	48
2.14	¥YGurafu	48
2.15	¥YNuri	49
2.16	¥YNurii	50
2.17	¥XGurafu	51
2.18	¥Xnuri	51
2.19	¥Xnurii	52
2.20	¥BGurafu	52
2.21	¥RGurafu	53
2.22	¥HenKo	53
2.23	¥Kakukigou	54
2.24	¥Tyokkakukigou	54
2.25	emathPs における線幅，線種の変更	54
2.25.1	¥En などの場合	55
2.25.2	¥YGurafu などの場合	56
2.25.3	斜線塗りの場合	57
2.25.4	¥HenKo の場合	57



# 1 新設コマンド・環境

## 1.1 pszahyou 環境

いままでの zahyou 環境に相当するものを pszahyou 環境と称します。この環境内では、図形描画部分は PostScript 言語に翻訳され eps ファイルに書き出されます。一方、文字を配置する部分は、 $\text{\LaTeX}$  の picture 環境で実行されます。

作成された eps ファイルを読み込むタイミングは、その eps ファイルが存在していれば、pszahyou 環境の冒頭で読み込まれ、pszahyou 環境内で図形描画部分はスキップされ、文字配置のみが行われます。

一方、eps ファイルが存在しないときは、eps ファイルを作成し、pszahyou 環境の終りで読み込まれます。文字配置も同時進行となりますから、その上に図形ファイル (eps) が読み込まれます。従って、描画部分に塗りつぶしがあると、文字が消えてしまう事態が発生します。そのときは、もう一度タイプセットする必要があります。

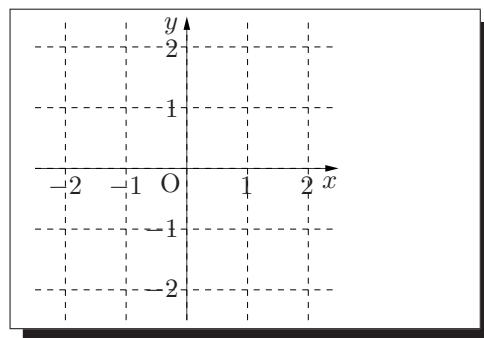
ただし、debug オプション付で

```
\usepackage[debug]{emathPs}
```

としたときは、eps ファイルが存在しても作り直します。従って pszahyou 環境の最後に eps ファイルが読み込まれることになります。

pszahyou 環境

```
\begin{pszahyou}[ul=8mm]%  
  (-2.5,2.5)(-2.5,2.5)  
  \zahyouMemori[g]  
\end{pszahyou}
```



この図において、座標軸の矢印付線分およびグリッド線(破線)は、eps ファイル samplePs1.eps に PostScript 言語で書き出された描画命令により描かれています。

eps ファイルは pszahyou 環境が現れるたびに、

```
\jobname1.eps  
\jobname2.eps  
\jobname3.eps  
.....
```

と作られて行きます。

現時点では、pszahyou 環境内に記述できるコマンドは少なく、上の例で用いられている \zahyouMemori のほか、どのようなものが使用可能であるかを、次節で見ていただくことにします。

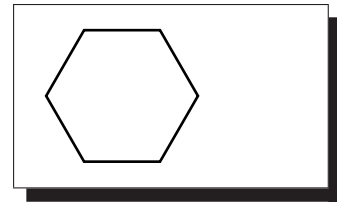
## 1.2 pszahyou\*環境

座標軸を描画しない環境です。



pszahyou\*環境

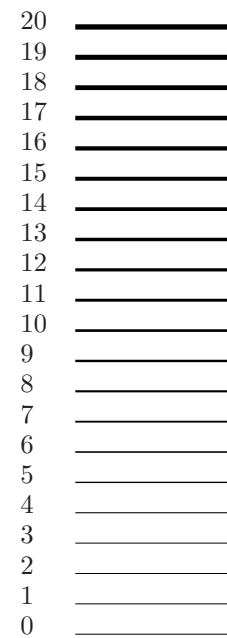
```
%begin{pszahyou*}[ul=10mm]%  
  (-1.1,1.1)(-1.1,1.1)  
  %rttenretu*{A(1,0);B(1,60);C(1,120);%  
    D(1,180);E(1,240);F(1,300)}%  
  %Takakkei{%A%B%C%D%E%F}  
%end{pszahyou*}
```



### 1.3 %setlinewidth

描画するすべてのコマンドに対し、線の太さを指定します。

```
%setlinewidth  
%begin{pszahyou*}[ul=4mm]%  
  (-2,5.5)(-.5,21)  
  %Ifor%w{0}{21}%Do{%  
    %put(-1.8,%w){%w}  
    %setlinewidth{%w}  
    %drawline(0,%w)(5,%w)}  
%end{pszahyou*}
```



%setlinewidth の引数の値とその太さの変化をご覧ください。このあたり、TeX、GS、printer (driver) などの環境によって結果は異なることでしょう。

pszahyou 環境のデフォルトは %setlinewidth{10} となっています。(tpic specials に比して太目です。tpic specials は %setlinewidth{3} に相当するようです。)

### 1.4 %setdash

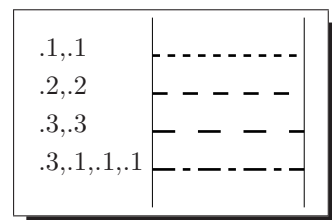
描画する線種は実線がデフォルトですが、これを破線にするコマンドです。



```

%setdash
%begin{pszahyou*}[ul=10mm]%
  (-1.5,2.1)(0,2.5)
  %put(-1.5,2){.1,.1}
  %setdash{.1,.1}%drawline(0,2)(2,2)
  %put(-1.5,1.5){.2,.2}
  %setdash{.2,.2}%drawline(0,1.5)(2,1.5)
  %put(-1.5,1){.3,.3}
  %setdash{.3,.3}%drawline(0,1)(2,1)
  %put(-1.5,.5){.3,.1,.1,.1}
  %setdash{.3,.1,.1,.1}%drawline(0,.5)(2,.5)
%setdash{}
%setlinewidth{1}
%drawline(0,0)(0,2.5)
%drawline(2,0)(2,2.5)
%end{pszahyou*}

```



すなわち，`%setdash` の引数に偶数個の数値を与えます。描画部分の長さ，描画しない部分の長さの繰り返しです。

描画を実線に戻すには，`%setdash{}` と，空の引数を与えます。

`%setdash` には，`[. .]` オプションで，開始位置のオフセットを指定する機能があります。上の図で，下から 2 番目の `%setdash{.3,.3}` の場合， $2=0.3*6+0.2$  ですから，右端の実線はクリップ機能で 0.3 ではなく，0.2 しかありません。そこで左端の開始オフセットを 0.05 と指定すれば，

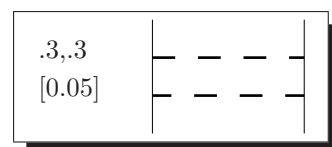
0.25(-) 0.3 0.3(-) 0.3 0.3(-) 0.3 0.25(-)

と，両端の実線部分の長さが揃います。

```

%setdash のオフセット
%begin{pszahyou*}[ul=10mm]%
  (-1.5,2.1)(0,1.5)
  %put(-1.5,1){.3,.3}
  %setdash{.3,.3}%drawline(0,1)(2,1)
  %put(-1.5,.5){[0.05]}
  %setdash[0.05]{.3,.3}%drawline(0,.5)(2,.5)
%setdash{}
%setlinewidth{1}
%drawline(0,0)(0,1.5)
%drawline(2,0)(2,1.5)
%end{pszahyou*}

```



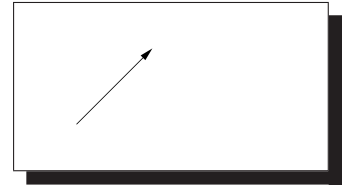
## 1.5 %setarrowsize

矢線の形状を変更するには，`%setarrowsize` を用います。まずは，デフォルトの確認です。以下，これを基準に各種のサイズを変更して見ます。



矢線，デフォルトの形状

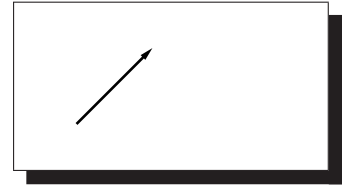
```
¥begin{pszahyou*}[ul=10mm](-.5,1.5)(-.5,1.5)
  ¥def¥A{(1,1)}
  ¥ArrowLine¥0¥A
¥end{pszahyou*}
```



直線部分を太くしてみましょう。

直線部分の太さ

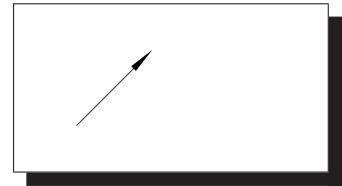
```
¥begin{pszahyou*}[ul=10mm](-.5,1.5)(-.5,1.5)
  ¥def¥A{(1,1)}
  ¥setarrowsize{10}{-}{-}
  ¥ArrowLine¥0¥A
¥end{pszahyou*}
```



次は，鏃を長くしてみます。

鏃の長さ

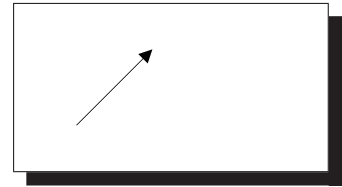
```
¥begin{pszahyou*}[ul=10mm](-.5,1.5)(-.5,1.5)
  ¥def¥A{(1,1)}
  ¥setarrowsize{}{-}{100}
  ¥ArrowLine¥0¥A
¥end{pszahyou*}
```



鏃の太さ変更です。

鏃の太さ

```
¥begin{pszahyou*}[ul=10mm](-.5,1.5)(-.5,1.5)
  ¥def¥A{(1,1)}
  ¥setarrowsize{}{50}{-}
  ¥ArrowLine¥0¥A
¥end{pszahyou*}
```

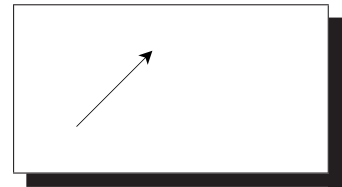


やじりの部分に窪みをつけるには，<数値>オプションをつけます。数値の部分は，やじりの二等辺三角形の高さを1としたときの，窪みの深さを表します。

¥setarrowsize の書式です。

鏃の窪み

```
¥begin{pszahyou*}[ul=10mm](-.5,1.5)(-.5,1.5)
  ¥def¥A{(1,1)}
  ¥setarrowsize<.25>{}{50}{-}
  ¥ArrowLine¥0¥A
¥end{pszahyou*}
```





```
¥setarrowsize<#1>#2#3#4
```

#1 : 窪みの比率

#2 : 直線部分の太さ

#3 : 鋸の太さ

#4 : 鋸の長さ

デフォルトは ¥setarrowsize{3}{25}{50} です。

また、引数を空にした場合はデフォルト値が採用されます。

## 1.6 クリッピング

### 1.6.1 クリッピング

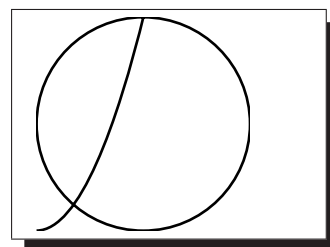
emathPs.sty で作られた EPS ファイルは

```
¥includegraphics[clip=true]{....}
```

で読み込まれます。例えば

clip—

```
¥begin{pszahyou*}[ul=20pt](0,4)(0,4)
¥En{(2,2)}{2}
¥YGurafu{X*X}{0}{3}
¥end{pszahyou*}
```



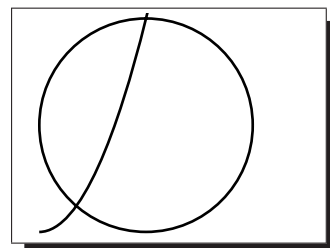
放物線  $y = x^2$  を  $0 \leq x \leq 3$  と範囲指定をしてありますが、描画領域  $0 \leq x \leq 4, 0 \leq y \leq 4$  の外側はクリップ機能で切り取られ、区間  $0 \leq x \leq 2$  だけが描画されています。

さらに、よく見ると、円の上下左右も削られています。これは、線幅 1pt が円周の周囲にはみだすことになってますが、描画領域外は容赦なく(?)切り取られているということなのです。

対応策は、描画領域を少し広めにするることになります。

描画領域を広めに—

```
¥begin{pszahyou*}[ul=20pt](0,4.1)(0,4.1)
¥En{(2.05,2.05)}{2}
¥YGurafu{(X-0.05)*(X-0.05)+0.05}{0.05}{3}
¥end{pszahyou*}
```



でも、面倒ですね。

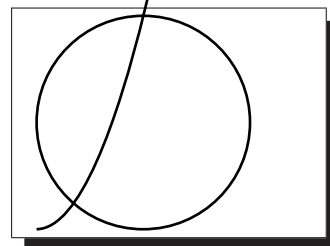
ということで、今回の改定ではクリップを無効にする方法を用意しました。



```

EPSClip=false
\begin{pszahyou*}[ul=20pt,EPSClip=false](0,4)(0,4)
\En{(2,2)}{2}
\YGurafu{X*X}{0}{3}
\end{pszahyou*}

```



注 1. Windows で dviout.exe をご利用の方へ。

dviout には, graphics の表示について多様な方法が用意されています。最後の図がクリップされてしまう, という方は

```

Option(0)
  Setup Parameters
    Graphic
      Ghostscript
        gclip: clip

```

がチェックされているものと思います。ここがチェックされていると, dviout によりクリップが実行されてしまいます。この文書を dviout で見るには, このチェックを外してください。そのほかにも dviout の設定によって, こちらの意図したことと異なる状況が生じることがいくつかあります。emathPs.sty を用いた文書については,

- 1a. dvips(k) で変換した PS ファイル
- 1b. さらに Distiller で作成した PDF ファイル

または

2. dvipdfm で作成した PDF ファイル

をご確認ください。

細かいことを言いますと, 上記 2. の PDF には問題があります。dvipdfm は  $x < 0$  の部分は無視するようで, 円の左端だけは切り取られています。これは dvipdfm の問題と考えています。

注 2. 最後の図では, 当然のことながら放物線も描画領域を逸脱しています。クリップ無効といっても, ほどほどにしてほしい, とお考えの方もあるでしょう。その方向での処理は, 少々時間が必要ですから, 今後の課題とさせていただきます。

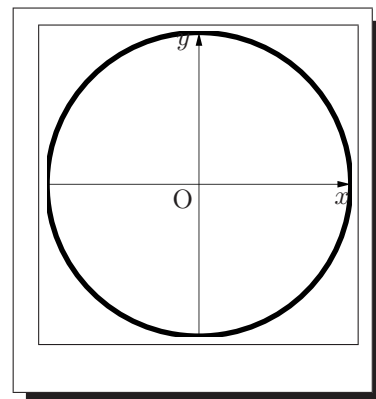
### 1.6.2 [borderwidth=...] オプション

emathPs.sty を用いて図を描いた場合, PDF に変換すると図の左端などが欠ける現象が発生することがあります。



円の左端が欠ける

```
%smallskip
%fbbox{%
%begin{pszhayou}[ul=20mm](-1,1)(-1,1)
  %setlinewidth{20}
  %En%{1}
%end{pszhayou}}
%smallskip
```



皆さんの環境ではいかがでしょうか。私の環境では、円の上下左右4箇所の円と正方形の接するところに円周の欠落が見られます。円周は少し太めにしていますから、本来なら正方形の外にはみ出すはずなのですが、はみ出していませんね。

これは、描画領域外はクリップされるという EPS の仕様です。

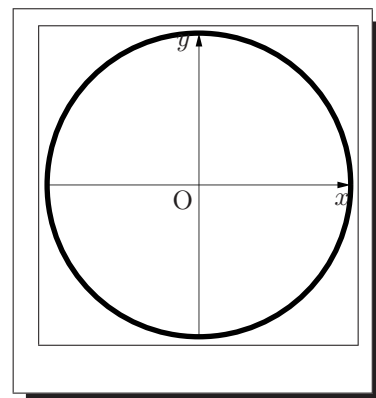
(この文書では、pszhayou 環境の領域を %fbbox で囲んで表示しています。余白をなくするため

%fbboxsep=0pt に設定しています。)

EPS には、このクリッピングを止めさせる方法が用意されています。当てはめてみましょう。

EPSclip=false オプション

```
%smallskip
%fbbox{%
%begin{pszhayou}[ul=20mm,EPSclip=false]%
  (-1,1)(-1,1)
  %setlinewidth{20}
  %En%{1}
%end{pszhayou}}
%smallskip
```



pszhayou 環境の [EPSclip=false] がそれです。私の環境では、上下左右4箇所ともクリップされず、円周は正方形の外にはみ出しています。皆さんの環境ではいかがでしょうか。

左と下はクリップされてしまう、という状況の方がおられると思います。これは pdf への変換法に左右される現象です。現在、確認されているのは

- (1) dvipsk+Distiller では、4 箇所ともクリップされない。
- (2) dvipdfm(x) では、左・下の 2 箇所がクリップされる。

グラフィックは dvi-ware 依存ですから、異なる状況が発生することはありうるわけではありますが.....

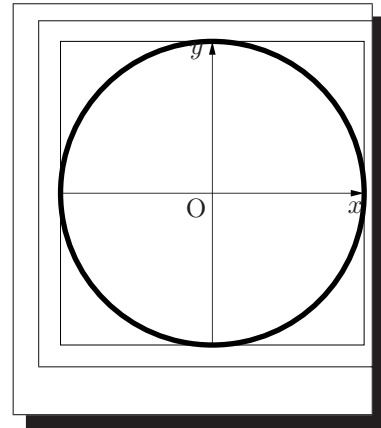
# 私に言わせれば、これは dvipdfm のバグでしょう。

この問題点に対応するため、emathPs.sty では、borderwidth=.. オプションを少し修正してみました。



borderwidth=5pt オプション

```
%\smallskip
%\fbbox{%
%\begin{pszahyou}[ul=20mm,borderwidth=5pt]%
%   (-1,1)(-1,1)
%   %setlinewidth{20}
%   %En%0{1}
%   %setlinewidth{2}
%   %Takakkei{(-1,-1)(1,-1)(1,1)(-1,1)}
%\end{pszahyou}}
%\smallskip
```



さていかがでしょうか。今度の図には、正方形が2つあります。小さい方の正方形が今までの図における正方形で、外側は borderwidth=5pt オプションで周囲に 5pt ずつの外枠をつけたものです。(こちらが pszahyou 環境が認識する描画領域です。)

描画領域を広げたわけですから、クリップする/しないは、関係なく、円周は全て欠損なく描画されているはずです。



### 1.6.3 ¥truexmax など

borderwidth=.. オプションは今までもあったのですが、座標軸が広がった領域全体に描画されていたものを、今回の改定で (-1,1)(-1,1) と指定した領域内 (先の図の小さい正方形内) にとどめることにいたしました。そのために ¥truexmax などの変数を新設しました。

¥xmax など

```
¥smallskip
¥begin{pszahyou*}[ul=5mm,borderwidth=5pt]%
  (-1,4)(-2,3)
  ¥tenretu*{A(¥xmin,¥trueymax)}¥Kuromaru¥A
  ¥Put¥A{¥EMparbox[t]{%
    $¥mbox{¥cmd{xmax}}=¥xmax$ ¥¥
    $¥mbox{¥cmd{xmin}}=¥xmin$ ¥¥
    $¥mbox{¥cmd{ymax}}=¥ymax$ ¥¥
    $¥mbox{¥cmd{ymin}}=¥ymin$
  }}
¥end{pszahyou*}
```

```
¥xmax = 4.35146
¥xmin = -1.35146
¥ymax = 3.35146
¥ymin = -2.35146
```

¥xmax などは、pszahyou 環境で指定した値に borderwidth を増減した値となっています。もとの値は ¥truexmax などに保存することにしました。

¥xmax など

```
¥smallskip
¥begin{pszahyou*}[ul=5mm,borderwidth=5pt]%
  (-1,4)(-2,3)
  ¥tenretu*{A(¥xmin,¥trueymax)}¥Kuromaru¥A
  ¥Put¥A{¥EMparbox[t]{%
    $¥mbox{¥cmd{truexmax}}=¥truexmax$ ¥¥
    $¥mbox{¥cmd{truexmin}}=¥truexmin$ ¥¥
    $¥mbox{¥cmd{trueymax}}=¥trueymax$ ¥¥
    $¥mbox{¥cmd{trueymin}}=¥trueymin$
  }}
¥end{pszahyou*}
```

```
¥truexmax = 4
¥truexmin = -1
¥trueymax = 3
¥trueymin = -2
```

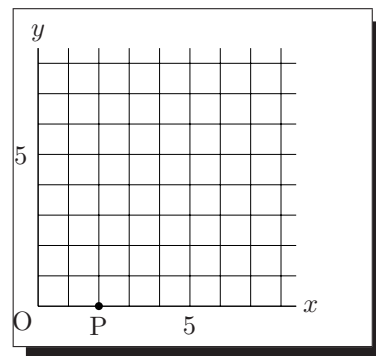


#### 1.6.4 掲示板から

今回の改定の発端となった，掲示板の投稿を見ておきます。

掲示板 No.235

```
%bigskip
%begin{pszahyou}
[ul=4mm,
zikusensyu=%drawline,
yokozikukigou=$x$,
tatezikukigou=$y$,
gentenkigou=0,
yokozikuhaiti={(3pt,0)[l]},
tatezikuhaiti={(0,3pt)[b]},
xscale=1,yscale=1,EPSclip=false]
(0,8.5)(0,8.5)
%zahyouMemori[g][n]<dx=1,dy=1,dash={}>
%def%P{(2,0)}%Put%P(0,-4pt)[t]{P}
%Kuromaru%P
%Put{(5,0)}(0,-4pt)[t]{5}
%Put{(0,5)}(-4pt,0)[r]{5}
%setlinewidth{3}
%end{pszahyou}
%bigskip
```



点  $P(2, 0)$  に打った黒丸の下が欠ける，という投稿でした。  
これに `[borderwidth=2pt]` を附加してみます。

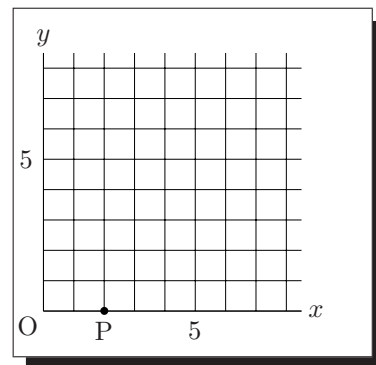


掲示板 No.235 修正

```

%bigskip
%begin{pszahyou}
[ul=4mm,borderwidth=2pt,
zikusensyu=%drawline,
yokozikukigou=$x$,
tatezikukigou=$y$,
gentenkigou=0,
yokozikuhaiti={(3pt,0)[l]},
tatezikuhaiti={(0,3pt)[b]},
xscale=1,yscale=1,EPScip=false]
(0,8.5)(0,8.5)
%zahyouMemori[g][n]<dx=1,dy=1,dash={}>
%def%P{(2,0)}%Put%P(0,-4pt)[t]{P}
%Kuromaru%P
%Put{(5,0)}(0,-4pt)[t]{5}
%Put{(0,5)}(-4pt,0)[r]{5}
%setlinewidth{3}
%end{pszahyou}
%bigskip

```



## 1.7 図と文字のずれ

emathPs.sty における pszahyou 環境では,

図形  
と  
文字

の配置について,

図形については, PostScript で

文字については,  $\text{\TeX}$  における zahyou(picture) 環境

という, 分列行進となっています。

図が小さいときは目立たないのですが, 大きな図ですと

図と文字のズレ

```

%begin{pszahyou*}[ul=19mm](0,7)(-.1,.25)
%Drawline{%XMIN%XMAX}%
%Ifor%x{0}{8}%Do{%
%Drawline{(%x,-.05)(%x,.05)}%
%Put{(%x,0)}(0,5pt)[b]{%x}%
}%
%end{pszahyou*}

```





右の方で，目盛り線と目盛りの文字の位置ずれが目立つようになります。

これは，Postscript と  $\text{T}_{\text{E}}\text{X}$  で，1pt の長さが異なるためです。すなわち

Postscript では， $1\text{pt} = 1/72\text{ inch}$

$\text{T}_{\text{E}}\text{X}$  では， $1\text{pt} = 1/72.27\text{ inch}$

ということで， $\text{T}_{\text{E}}\text{X}$  のほうが少し短くなっていますから，右に行くほど，文字が目盛り線よりも左に寄っていく状態がしゅつたいします。

そこで，文字配置について，このズレを修正するため

cPut (c は char の頭文字のつもり)

を新設しました。

```
%cPut
%cPut{(%x,0)}(0,5pt)[b]{%x}%
```



目盛り線と文字のズレは解消しています。

ただし，

文字の配置は cPut で，

図形の配置は Put で

という二重構造となります。

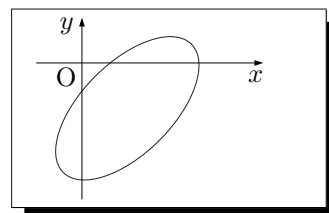
望ましいのは，図形・文字双方を PostScript の管理下に置くことですが，PostScript の文字出力は面倒なので，実現は遠い将来となりそうです。

## 1.8 図形の回転

この節の話は，新機能ではありませんが，BBS で話題となったので，とりあげておきます。

例えば，楕円を回転させるには， $\text{\%rotatebox}$  を利用するのが簡単です。

```
zahyou 環境における \rotatebox
\begin{zahyou}[ul=6mm](-1,4)(-3,1)
  \tenretu*{A(1,-1)}
  \Put%A{\rotatebox{45}{\Daen#0{2}{1}}}
\end{zahyou}
```

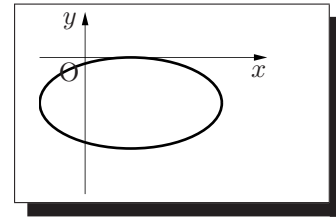


pszahyou 環境ではどうなるでしょうか。



pszhyou 環境における `\rotatebox`

```
\begin{pszhyou}[ul=6mm](-1,4)(-3,1)
  \tenretu*{A(1,-1)}
  \Put%A{\rotatebox{45}{\Daen#0{2}{1}}}
\end{pszhyou}
```



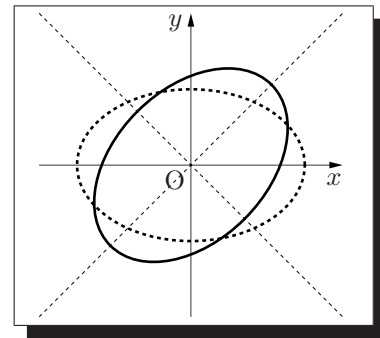
回転しません。すなわち `\rotatebox` は無視されています。emathPs.sty では、`\rotatebox` はサポートされていませんし、今後サポートされる予定はありません。これには、理由があるのですが、ここでは述べません。

それに換えて `\EPSkaiten` コマンドを用意しています。

楕円を回転させてみます。

`\EPSkaiten`

```
\begin{pszhyou}[ul=5mm](-4,4)(-4,4)
  \EPSkaiten#0{45}{\Daen#0{3}{2}}
  \setdash{.1,.1}
  \Daen#0{3}{2}
  \setlinewidth{3}
  \kTyokusen#0{45}{}{}
  \kTyokusen#0{-45}{}{}
\end{pszhyou}
```



回転させるコマンド名が `\EPSkaiten` で、その書式は：

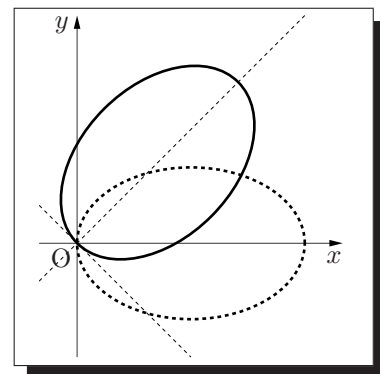
`\EPSkaiten#1#2#3`

- #1 : 回転の中心
- #2 : 回転角 (六十分法)
- #3 : 回転する対象

回転の中心と楕円の中心を別にとってみましょう。点 (3, 0) を中心とする楕円を原点のまわりに回転します。

`\EPSkaiten`

```
\begin{pszhyou}[ul=5mm](-1,7)(-3,6)
  \tenretu*{A(3,0)}
  \EPSkaiten#0{45}{\Daen#A{3}{2}}
  \setdash{.1,.1}
  \Daen#A{3}{2}
  \setlinewidth{3}
  \kTyokusen#0{45}{}{}
  \kTyokusen#0{-45}{}{}
\end{pszhyou}
```



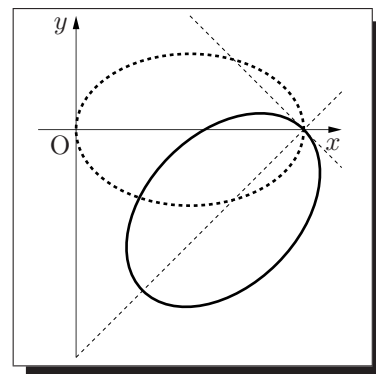
次は、回転の中心を楕円の右端 (6, 0) にとります。



```

%EPSkaiten
\begin{pszahyou}[ul=5mm](-1,7)(-6,3)
  \tenretu*{A(3,0);C(6,0)}
  %EPSkaiten%C{45}{%Daen%A{3}{2}}
  %setdash{.1,.1}
  %Daen%A{3}{2}
  %setlinewidth{3}
  %kTyokusen%C{45}{}{}
  %kTyokusen%C{-45}{}{}
\end{pszahyou}

```



## 1.9 カラー化

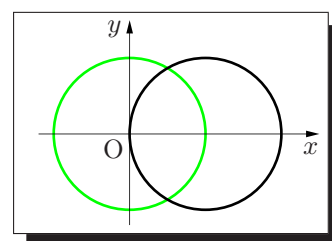
### 1.9.1 EMpscolor 環境

L<sup>A</sup>T<sub>E</sub>X の `%color` コマンドは、`pszahyou` 環境の図形描画部分には無効です。図形をカラー描画するために `EMpscolor` 環境を用意しました。

```

EMpscolor 環境
\begin{pszahyou}[ul=10mm](-1.2,2.5)(-1.2,1.5)
  \begin{EMpscolor}{green}
    %En%0{1}
  \end{EMpscolor}
  %En{(1,0)}{1}
\end{pszahyou}

```



(注) `dviout` でご覧の方は、`Graphic` の設定いかんでは、カラーになりません。

### 1.9.2 %defineEMpscolor コマンド

現時点で、`emathPs` で定義されているカラーは

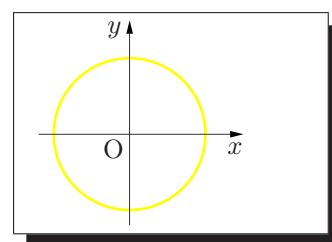
red, green, blue  
white, black

の 5 種類です。その他のカラーを使用したいときは `%defineEMpscolor` コマンドで定義してください。

```

%defineEMpscolor
\begin{pszahyou}[ul=10mm](-1.2,1.5)(-1.2,1.5)
  %defineEMpscolor{yellow}{1}{1}{0}
  \begin{EMpscolor}{yellow}
    %En%0{1}
  \end{EMpscolor}
\end{pszahyou}

```



`%defineEMpscolor` コマンドは



色の名前に続き, R, G, B の数値 (0 ~ 1)

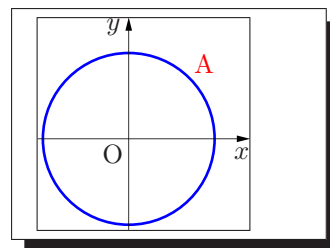
を引数に与えます。

### 1.9.3 `%color` との併用

文字に色をつけたいときは,  $\text{\LaTeX}$  の `%color` コマンドを用いますが, 現時点では, 空白の混入がおきやすくなっています。次の例では,  $\text{\LaTeX}$  の認識している `picture` 環境を `%fbox` で囲んでいますが, 図がはみ出していることがお分かりでしょう

`%color` との併用

```
%fboxsep=0pt%relax%fbox{%  
%begin{pszhayou}[ul=8mm](-1.5,2)(-1.5,2)%  
%def%A{(1,1)}%  
%begin{EMpscolor}{blue}%  
%En%O{1.414}%  
%end{EMpscolor}%  
{%color{red}%Put%A[ne]{A}}%  
%end{pszhayou}}%
```

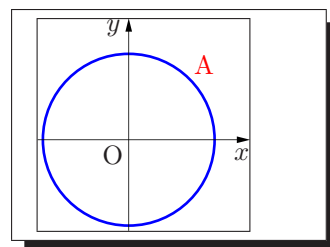


ただし, 2 回タイプセットをすれば, 空白の混入はなくなります。

また, 1 回目のタイプセットでもこの混入を回避するには, `%color` 文を `%Put` の引数に入れてしまうのが有効です。

空白の混入回避

```
%fboxsep=0pt%relax%fbox{%  
%begin{pszhayou}[ul=8mm](-1.5,2)(-1.5,2)%  
%def%A{(1,1)}%  
%begin{EMpscolor}{blue}%  
%En%O{1.414}%  
%end{EMpscolor}%  
%Put%A[ne]{%color{red}A}%  
%end{pszhayou}}%
```

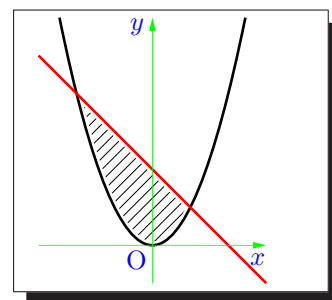


さて, 最後に少し大きな例です。



### 一、例

```
%\begin{pszahyou*}[ul=5mm](-3,3)(-1,6)%
\def\Fx{X*X}%
\def\Gx{2-X}%
\gsave
\setlinewidth{3}%
\YNurii*<syasentanmatu=1mm>\Fx\Gx{-2}{1}%
\grestore
\YGurafu*\Fx%
\begin{EMpscolor}{red}
  \YGurafu*\Gx%
\end{EMpscolor}
\begin{EMpscolor}{green}
  \Put#0{\color{blue}\drawXYaxis}%
\end{EMpscolor}%
\end{pszahyou*}%
```



### 1.10 EMpsrectbox 環境

### 1.10.1 EMpsrectbox とは

emathPh.sty で定義されている `rectbox` 環境の `emathPs` 版という位置付けです。ページをまたぐことができない、傍注もつけられないというのは、`rectbox` と同様です。

### · EMpsrectbox 環境 ·

[illegible]

あああああああああああああああああああああああああああああああああああ  
 ああああああああああ  
 じじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじ  
 じじじじじじじじじじじじじ

rectbox 環境に比して、罫線が太いことでしょうか。コーナーがきちんとつながっていることが特徴です。



`fboxsep` 罫線と中のテキストとの間隔は、`#fboxsep` で決まります。これを変更したいときは、`[fbox=..]` オプションを与えます。

```
\begin{Empsrectbox}[fboxsep=1zw]
ああああああああああああああああああああ
ああああああああああああああああああああ
ああああああああああああああああああああ
\end{Empsrectbox}
```

`hsep vsep [fboxsep=..]` オプションは、左右・上下すべてを一律に変更しますし、当該環境内のみではありますが、`\fboxsep` の値が変更されています。

$\begin{array}{c} \text{あ あ あ あ あ あ あ あ あ あ あ あ} \\ \text{い い い い い い い い い い い い} \\ \text{う う う} \end{array}$

うう

そこで、左右の空きを指定するオプション `hsep=..` と、上下の空きを指定するオプション `vsep=..` を新設しました。

$$\begin{array}{c} \text{ああああああああああああああああ} \\ \text{ああああああああああああああああ} \end{array}$$







```
\begin{EMpsrectbox}[item={~中央見出し~},itempos=c]
ああああああああああああああああああああ
ああああああああああああああああああああ
ああああああああああああああああああああ
\end{EMpsrectbox}
```

中央見出し

```
\begin{EMpsrectbox}[item={~右見出し~},itempos=r]
ああああああああああああああああ
ああああああああああああああああああ
ああああああああああああああああああ
\end{EMpsrectbox}
```

右見出し

ああああああああああああああああああああああああああああああああああ  
ああああああああああああああああ

罫線の種類を変更する方法は、rectbox 環境とは異なります。rectbox に<...>オプションをつけます。

```
\begin{Empsrectbox}<\setdash{3,2}>
あああああああああああああああ
あああああああああああああああ
あああああああああああああああ
\end{Empsrectbox}
```

ああああああああああああああああああああああああああああああ  
ああああああああああああああああ

$$\forall \text{begin}\{\text{EMpsrectbox}\}[\dots]\langle\dots\rangle$$

19



こちらも rectbox 環境とは異なります。

あああああああああああああああああああああああああああああああああ  
ああああああああああああああああ

EMpsrectbox 環境は、コーナーが直角です。これを丸く—四分円にする試みです。

[rectboxoval] オプションをつけると、角の丸い枠で囲むことができます。

-rectboxoval=5pt オプション

[rectboxoval=5pt] オプションをつけると、角の丸は半径 5pt の円となります。

### hsep=.,vsep=.



オプションを用います。

## -hsep, vsep オプション

[illegible]

¥verb+[hsep=.,vsep=..]+オプションを付加した場合は、

そちらが優先されます。

\end{EMpsrectbox}

[hsep=.,vsep=.] オプションを付加した場合は、そちらが優先されます。

注 グラフィックスは機種依存, dvi-ware 依存です。

EMpsrectbox 環境は、dvipdfm(x) には対応していません。再述べていますが、 $x < 0$ ,  $y < 0$  を無視する仕様のようで、枠の下辺と左辺が削られます。

確認の意味で、dvipsk+Distiller で作成した PDF を同梱します。

### 1.10.8 mawarikomi 環境内の囲み

mawarikomi 環境内で, ascmac.sty で定義されている itembox 環境などを用いると

mawarikomi &amp; itembox

```
\begin{mawarikomi}{}{%
```

¥unitlength=1pt¥relax

```
\begin{picture}(200,100)
```

```
¥framebox(200,100){¥Huge 図}
```

$$\text{\texttt{\$end{picture}}}$$

}

あああ

```
\begin{itembox}{みだし}
```

*J J*

\end{itembox}

[illegible][illegible][illegible][illegible]

*(A series of vertical bars representing a barcode or similar pattern.)*

¥end{mawarikomi}















別法

```
%begin{mawarikomi}[-5]{}{%  
  %unitlength=1pt%relax  
  %begin{picture}(200,100)  
    %framebox(200,100){%Huge 図}  
  %end{picture}  
}  
あああ  
%end{mawarikomi}  
  
%begin{minipage}{%EMWRlinewidth}  
%begin{itembox}{みだし}  
アアアアアアアアアアアアアアアアアアアアアアアアアアアアアアアアアアアアアアア  
%end{itembox}  
%end{minipage}  
%begin{mawarikomi*}  
じじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじ  
じじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじ  
じじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじ  
じじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじ  
じじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじ  
%end{mawarikomi*}
```

あああ

みだし

アアアアアアアアアアアア  
アアアアアアアアアアアア  
アアアアアアアアアアアア



じじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじ  
じじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじ

じじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじ  
じじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじ  
じじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじ  
じじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじ  
じじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじじ

### 1.10.9 rectbox 環境の併用

emathPs.sty をロードしていても，rectbox を用いることは可能です。

ただし，rectbox 環境の罫線は，%drawline を用いる — すなわち tpic-specials に依存します。したがって

```
%usepackage{epic,eepic}
```



をプリアンブルに宣言しておく必要があります。

rectbox 環境

```
¥begin{rectbox}
ああああああああああああああああ
ああああああああああああああああ
ああああああああああああああああ

iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii
iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii
iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii
iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii

¥end{rectbox}
```

```
ああああああああああああああああああああああああああああああああああ
ああああああああああああああああああ
iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii
iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii
```

### 1.11 EPSfilename=.. オプション

pszahyou 環境で EPS ファイルが作成されますが、作成する EPS ファイルの名前を指定できるようにしました。

さらに、作成した EPS ファイルは pszahyou 環境で読み込まれるのが原則ですが、ここでは読み込まず、別のところで読み込むことも可能としました。

下の例では、数字を三角形で囲んだもので番号付けをしています。枠の三角形を pszahyou 環境を用いて EPS ファイル化したものを用いています。

2 桁の数字は枠にかかっています。2 桁も使用したいのなら、プリアンブルの

```
¥def¥ippen{1.6em}% 1 桁数字
```

右辺値を大きくするなり、文字サイズを小さくするなりの修正が必要です。

① ああああ

② iiiiii

③ うううう

④ うううう

⑤ うううう

⑥ うううう



7 うううう

8 うううう

9 うううう

10 うううう

## 1.12 下線

### 1.12.1 `\pskasen`

下線を `eps` 画像にして処理しようという試みです。

ただし、行をまたぐことはしません。また、この文書で述べていることは暫定的なもので、仕様が変化することがありますから、ご承知おき願います。

### 1.12.2 `\underline` との比較

L<sup>A</sup>T<sub>E</sub>X 標準の `\underline` と同じことをする `\pskasen` から見て行きます。

比較

```
\underline{あいうえお}\quad
\pskasen{あいうえお}
```

あいうえお あいうえお

### 1.12.3 数式モードでは

数式モードでの使用を見ておきましょう。

数式モード

```
\pskasen{$\bunsuu{12}$}
$\pskasen{\bunsuu{12}}$
```

$\frac{1}{2} \frac{1}{2}$

別行立て数式モード

```
\[ x=\pskasen{\bunsuu{12}} \]
```

$x = \frac{1}{2}$

### 1.12.4 下線の太さ

`\pskasen` は、画像ですから、いろいろな細工が可能です。

下線の太さを変更するには `<linewidth=>` オプションです。デフォルトの右辺値は 4 としてあります。

下線の太さ

```
\pskasen<linewidth=20>{あいうえお}
```

あいうえお



#### 1.12.5 下線を破線で

下線を破線とするには、<dash=..>オプションを用います。

下線を破線で

```
¥pskasen<dash={2,2}>{あいうえお}
```

あいうえお

#### 1.12.6 二重下線

下線を二重とするには、¥pskasen に [...] オプションを与えます。[ ] 内の数値は二本線の間隔で、単位は pt です。

二重下線

```
¥pskasen[1.5]{あいうえお}
```

あいうえお

#### 1.12.7 下線に色

下線に色をつけるには、¥pskasen に<iro=..>オプションを与えます。

色付き下線

```
¥pskasen<iro=red>{あいうえお}
```

あいうえお

#### 1.12.8 下線上下の間隔

下線と下線をつけた文字列との間隔を調整するには、コマンド ¥kasenUehosei の引数に増減する数値（単位付）を与えます。下線は、正の値で下方に、負の値で上方に移動します。

¥kasenUehosei

```
¥kasenUehosei{-6pt}%
```

```
¥pskasen<iro=red>[2]{あいうえお}
```

あいうえお

下線とその下の行との間隔を調整するコマンドが ¥kasenSitahosei です。

¥kasenSiahosei

```
¥pskasen{あいうえお}
```

oooooooooooooooooooooooo

oooooooooooooooooooooooo

oooooooooooooooooooooooo

```
¥kasenSitahosei{10pt}%
```

```
¥pskasen{あいうえお}
```

oooooooooooooooooooooooo

oooooooooooooooooooooooo

oooooooooooooooooooooooo

あいうえお oooooooooooo

oooooooooooooooooooooooo

oooooooooooooooooooooooo

oooooooooooooooooooooooo

oooooooooooo

あいうえお oooooooooooo

oooooooooooooooooooooooo

oooooooooooooooooooooooo

oooooooooooooooooooooooo

oooooooooooo

第 1 段落が標準の間隔です。第 2 段落は下線の下を 10pt 増やすように指示されています。



### 1.12.9 左右のアキ

下線をつけた部分と、その前後の文章とのアキについては、`%underline` のそれと同様に少しあけることとしました。

前後のアキ

`%cmd{underline}` の場合から見ましょう。

あいうえお `%underline{かきくけこ}` さしすせそ

次に `%cmd{pskasen}` の場合です。

あいうえお `%pskasen{かきくけこ}` さしすせそ

`%unerline` の場合から見ましょう。

あいうえお かきくけこ さしすせそ

次に `%pskasen` の場合です。

あいうえお かきくけこ さしすせそ

### 1.12.10 下線近辺に文字配置

下線の左端など、下線の近辺に文字列を配置したいことがあります。そのために、`%pskasen` (あるいは `%psnamikasen`) に `'....'` オプションを用意しました。`'.....'` 内に、下線を引く `pszahyou*` 環境内に記述するコマンドを書くことができます。

`'.....'` オプション

`%verb+'.....'` 内に記述したものは、下線を描画する `%textsf{pszahyou}` 環境に置かれます。この環境の原点は、下線の左端です。

`%pskasen%`

`'%Put%0(0,0)[c]{$%bullet$}'%`

`{あいうえお}`

下線の右端は `%cmd{XMAX}` で、その座標は `%verb+(%xmax,0)+` となっています。

`%pskasen%`

`'%Put%XMAX(0,0)[c]{$%bullet$}'%`

`{あいうえお}`

`'.....'` 内に記述したものは、下線を描画する `pszahyou` 環境に置かれます。この環境の原点は、下線の左端です。あいうえお

下線の右端は `%XMAX` で、その座標は `(%xmax,0)` となっています。あいうえお

下線に番号を振って区別する例です。



下線に番号

```
あいう
¥pskasen<kasenSitahosei=5pt>%
  '¥Put¥0(0,-2pt)[r]{¥scriptsize (1)}'¥
  {かきくけこ}
さしすせそ
¥pskasen%
  '¥Put¥0(0,-2pt)[r]{¥scriptsize (2)}'¥
  {なにぬねの}
¥begin{enumerate}[(1)]
  ¥item 下線部 (1) について...
  ¥item 下線部 (2) について...
¥end{enumerate}
```

あいう かきくけこ さしすせ  
(1)

そ なにぬねの  
(2)

(1) 下線部 (1) について...

(2) 下線部 (2) について...



### 1.12.11 %pskasen の書式

%pskasen の書式です。%psnamikasen も同様です。

%pskasen<#1>[#2]’#3’#4  
#1 : key=val の形式  
有効な key は  
    linewidth  
    dash  
    iro  
    kasenUehosei (%kasenUehosei コマンドは有効範囲内すべてに効きます)  
    kasenSitahosei  
    kasenFunc (%psnamikasen に対してのみ)  
で、いずれも効果は局所的です。  
#2 : 下線を二重にするとき、二重線の間隔 (無名数で単位は pt がつきます。)  
#3 : 下線を引く pszahyou 環境内にそのまま配置されます。  
その pszahyou 環境について  
    %unitlength は 1pt  
    原点 (%0) は、下線の左端  
    右端が %XMAX, 座標は (%xmax,0)  
#4 : 下線を引く対象

### 1.13 波下線

波下線を引くコマンドが %psnamikasen です。

#### 1.13.1 比較

新設した %psnamikasen と %uwave, %namikasen とを比較してみます。

比較

%uwave{あいうえお}

%namikasen{あいうえお}

%psnamikasen{あいうえお}

あいうえお  
あいうえお  
あいうえお

#### 1.13.2 各種 option

%pskasen に対する各種オプションなどは、%psnamikasen に対しても有効です。



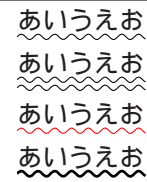
#### オプション

```
¥psnamikasen{あいうえお}

¥psnamikasen[3]{あいうえお}

¥psnamikasen<iro=red>{あいうえお}

¥psnamikasen<linewidth=10>{あいうえお}
```



また, ¥kasenUehosei, ¥kasenSitahosei も使用できます。

なお, これらはそれぞれ ¥namikasenUehosei, ¥namikasenSitahosei と同値なコマンドです。

### 1.13.3 波線の形状

波線は pszahyou 環境で

```
¥YGurafu*{sin(X)}
```

で描画されています。この部分を変更するオプションが

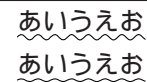
```
<kasenFunc=...>
```

オプションです。使用例をいくつかご覧ください。

#### 波の高さ

```
¥psnamikasen{あいうえお}

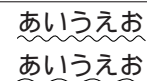
¥psnamikasen<%
  kasenFunc={¥YGurafu*{.5*sin(X)}}%
>{あいうえお}
```



#### 波の周期

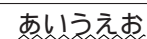
```
¥psnamikasen{あいうえお}

¥psnamikasen<%
  kasenFunc={¥YGurafu*{sin(X/2)}}%
>{あいうえお}
```



#### ギザギザ波

```
¥psnamikasen<%
  kasenFunc={¥YGurafu*{%
    4*min(.2*X-int(.2*X), 1-.2*X+int(.2*X))}}
>{あいうえお}
```





雑

```
%namikasenUehosei{2pt}%
%psnamikasen<%
  kasenFunc={%YGurafu*{abs(2*sin(X/2))}}
>{あいうえお}

%psnamikasen<kasenFunc={%
  %YNurii[1]{abs(2*sin(X/2))}{0}%xmin%xmax}
>{あいうえお}

%psnamikasen<kasenFunc={%
  %YNurii[nuriiro=red]{4*min(.2*X-int(.2*X),1-.2*X+int(.2*X))}{0}%xmin%xmax}
>{あいうえお}
```

あいうえお  
あいうえお  
あいうえお

## 1.14 %ovalbox

ovalbox の枠線を eps 画像にして処理しようという試みです。

### 1.14.1 fancybox.sty & eepic.sty

fancybox.sty で定義されている %ovalbox を eepic.sty を読み込んでいるソースファイルでタイプセットすると枠線が乱れます。

```
%ovalbox & eepic.sty
%ovalbox{$t=1$}
```

$t = 1$

右上の四分円と上罫線がつながらなくなります (右下も同様)。

### 1.14.2 %emovalbox

emath.sty では, %emovalbox を定義して対応することになっています。

```
%emovalbox
%ovalbox{$t=1$}
%emovalbox{$t=1$}
```

$t = 1$   $t = 1$

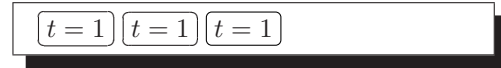


### 1.14.3 `\psovalbox`

いずれ、罫線の太さを変えたいとか、四分円の半径を変えたい、といった要求が出てくる予感がありますので、枠線を eps 画像にして、細かい変更が可能にしましたのが `\psovalbox` です。

`\psovalbox` —

```
\ovalbox{$t=1$}  
\emovalbox{$t=1$}  
\psovalbox{$t=1$}
```



なお、このコマンド `\psovalbox` は「perl との連携機能」を必要としません。

罫線の太さ 枠は eps 画像ですから、印刷環境によって太さは異なる可能性があります。太くしてみましょう。emathPs.sty においては、線の太さの変更は `linewidth=..` によって行われるのが原則です。

`linewidth=..` オプション —

```
\psovalbox{$t=1$}~  
\psovalbox<linewidth=10>{$t=1$}
```



`dash=..` オプションも有効ではあります。

`dash=..` オプション —

```
\psovalbox{$t=1$}~  
\psovalbox<dash={1,2}>{$t=1$}
```



枠線を描画する pszahyou\*環境の `\unitlength` は 1pt となっています。それを前提に `dash=...` の右辺値を与えなければなりません。

`ovalsep=..` オプション 中のテキストと枠線との間は `\fboxsep` だけ空くことになっていますが、これを変更するオプションです。

`ovalsep=..` オプション —

```
\psovalbox{$t=1$}~  
\psovalbox<ovalsep=6pt>{$t=1$}
```



`ovalradius=..` オプション コーナーの四分円の半径は、デフォルトでは 2pt となっていますが、これを変更するオプションです。

`ovalradius=..` オプション —

```
\psovalbox{$t=1$}~  
\psovalbox<ovalradius=5pt>{$t=1$}
```



枠線をカラーで 枠罫線に色をつけるオプションが `<iro=..>` です。

`iro=..` オプション —

```
\psovalbox<iro=red>{$t=1$}
```





塗りつぶし `\psovalbox` 内の背景を塗りつぶすコマンドが `\psovalbox*` です。

`\psovalbox*` —

`\psovalbox*{$t=1$}`

`t = 1`

塗りつぶしの濃度は [...] オプションで指定するのは `\Nuritubusi` と同様です。

濃度調整 —

`\psovalbox*[0.2]{$t=1$}`

`t = 1`

背景をグレーではなく、色を指定するオプションが `[nuriiro=...]` です。

背景色 —

`\psovalbox*[nuriiro=yellow]{$t=1$}`

`t = 1`

枠線にも色を指定したりするには `<...>` オプションを併用します。

枠線にも色 —

`\psovalbox*[nuriiro=yellow]%  
<iro=red,ovalsep=6pt,ovalradius=5pt>%  
{$t=1$}`

`t = 1`

中のテキストにも色をつけるのは、`\color` または `\textcolor` を用います。

テキストにも色 —

`\psovalbox*[nuriiro=yellow]%  
<iro=red,ovalsep=6pt,ovalradius=5pt>%  
{\textcolor{green}{$t=1$}}%`

`t = 1`

#### 1.14.4 `\psovalbox` の書式

`\psovalbox` の書式です。

実は、`\psovalbox` の正式コマンド名は `\EMPsovalbox` です。 `pstrick.sty` に `\psovalbox` が既に定義されていますから、このスタイルを使用する際は、コマンド名の衝突が起きます。その場合は、`\psovalbox` は `pstrick.sty` のものとし、`emathPs.sty` のコマンドは `\EMPsovalbox` としてください。



```

%EMpsovalbox<#1>#2
  #1 : key=val の形式
      有効なキー
          linewidth
          dash
          ovalsep
          ovalradius
          iro
  #2 : 枠で囲む文字列

%EMpsovalbox* [#1]<#2>#3
  #1 : 0~1 の間の数値でグレーの濃度を指定 (0 で白, 1 で黒)
      または
      nuriiro=.. で背景色を指定
  #2 : %psovalbox の<#1>と同じ
  #3 : 枠で囲む文字列

```

## 2 対応コマンド

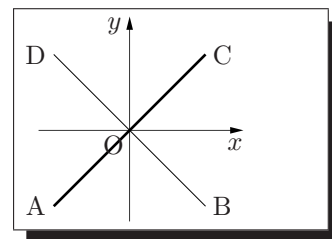
従来のコマンドのうち, pszahyou 環境でも使用可能なものです。ただし, オプション引数については少し異なるものもあります。

### 2.1 %Drawline

```

%Drawline
%begin{pszahyou}[ul=10mm](-1.2,1.5)(-1.2,1.5)
  %tenretu{A(-1,-1)w;B(1,-1)e;C(1,1)e;D(-1,1)w}
  %Drawline{%A%C}
  %setlinewidth{3}
  %Drawline{%B%D}
%end{pszahyou}

```



デフォルトの線幅は tpic specials のそれと比べると太くなっています (上図の線分 AC)。これを変更するコマンドが新設の

```
%setlinewidth
```

で, tpic のデフォルトは %setlinewidth{3}としたものと大体同じです。(線分 BD)

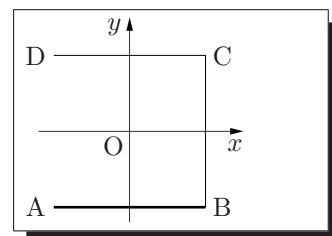


### 2.1.1 <linewidth=..>オプション

`%setlinewidth` コマンドによって、線の太さを変えられますが、この変更は以降すべてに効いてきます。グルーピングは無効です。

`%setlinewidth` による変更

```
%begin{pszahyou}[ul=10mm](-1.2,1.5)(-1.2,1.5)
  %tenretu{A(-1,-1)w;B(1,-1)e;C(1,1)e;D(-1,1)w}
  %Drawline{%A%B}
  {%
    %setlinewidth{3}
    %Drawline{%B%C}
  }%
  %Drawline{%C%D}
%end{pszahyou}
```

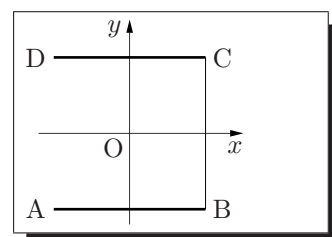


この図で、線分 AB はデフォルトの太さで引かれます。線分 BC は `setlinewidth{3}` により、細めになりますが、そのグルーピングが終わっても線分の太さはデフォルトには戻りません。

そこで、`%Drawline` のオプション引数で、局所的な線分の太さを変更する機能を附加しました。

`linewidth=..` オプションによる変更

```
%begin{pszahyou}[ul=10mm](-1.2,1.5)(-1.2,1.5)
  %tenretu{A(-1,-1)w;B(1,-1)e;C(1,1)e;D(-1,1)w}
  %Drawline{%A%B}
  %Drawline<linewidth=3>{%B%C}
  %Drawline{%C%D}
%end{pszahyou}
```



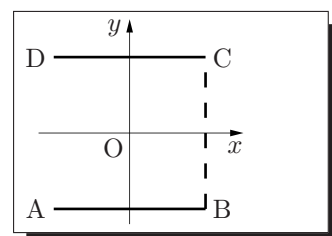
今度は、線分 BC のみが線幅を変更され、次の `%Drawline{%C%D}` によって引かれる線分 CD の線幅はデフォルトに戻っています。

### 2.1.2 <dash=..>オプション

破線にするコマンド `%setdash` も、以降すべてに効いてきます。局所的な変更は `%Drawline` に `dash=` オプションをつけてください。

`dash=..` オプションによる変更

```
%begin{pszahyou}[ul=10mm](-1.2,1.5)(-1.2,1.5)
  %tenretu{A(-1,-1)w;B(1,-1)e;C(1,1)e;D(-1,1)w}
  %Drawline{%A%B}
  %Drawline<dash={0.2,0.2}>{%B%C}
  %Drawline{%C%D}
%end{pszahyou}
```

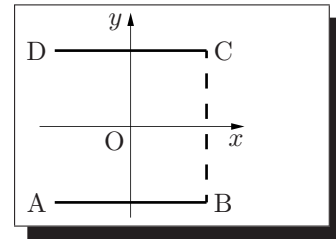


オフセットを附加したいときは



dash=.. オプション (オフセット付)

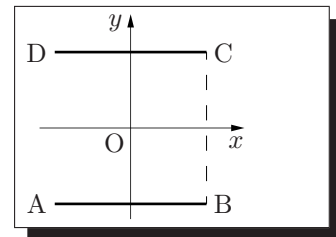
```
%begin{pszahyou}[ul=10mm](-1.2,1.5)(-1.2,1.5)
%tenretu{A(-1,-1)w;B(1,-1)e;C(1,1)e;D(-1,1)w}
%Drawline{%A%B}
%Drawline<dash=[.1]{0.2,0.2}>{%B%C}
%Drawline{%C%D}
%end{pszahyou}
```



線幅指定オプションとの併用も可能です。

dash, linewidth オプション併用

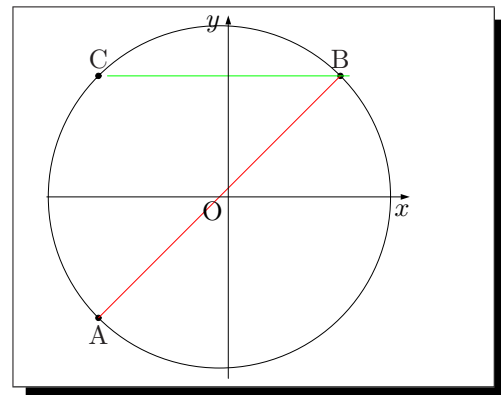
```
%begin{pszahyou}[ul=10mm](-1.2,1.5)(-1.2,1.5)
%tenretu{A(-1,-1)w;B(1,-1)e;C(1,1)e;D(-1,1)w}
%Drawline{%A%B}
%Drawline<linewidth=3,dash=[.1]{0.2,0.2}>{%B%C}
%Drawline{%C%D}
%end{pszahyou}
```



### 2.1.3 <iro=..>オプション

%color コマンドを発行すると、いわゆる「空白の混入」が起きやすくなります。一例です。  
空白の混入

```
%begin{zahyou}[ul=8mm]%
(-3,3)(-3,3)
%tenretu{%
A(-2,-2)s;B(2,2)n;C(-2,2)n}
%kuromaru{%A;%B;%C}
%En%0{2.828}
{%color{red}%Drawline{%A%B}}
{%color{green}%Drawline{%B%C}}
%end{zahyou}
```



3 点 A, B, C, 円および線分 AB (赤) は正しい位置に描画されていますが、線分 BC (緑) は、半角空白分右にずれています。(座標軸、また然り。)

これは、%color の責任というよりは、%color による色変更を局所的にするためのグルーピング終了記号 '}' の後に '%' を補うことを怠っているために発生しているのです。

行末に '%' を補ってみましょう。

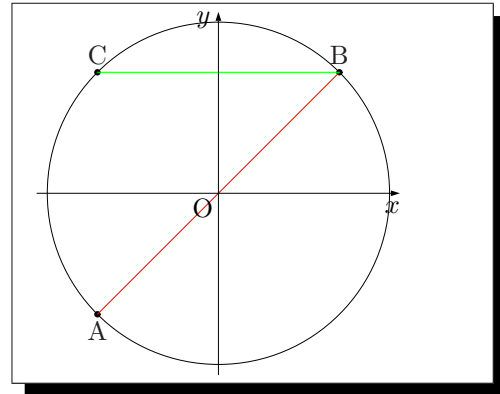


行末に ‘%’

```

\begin{zahyou}[ul=8mm]%
  (-3,3)(-3,3)
\tenretu{%
  A(-2,-2)s;B(2,2)n;C(-2,2)n}
\kuromaru{\A;\B;\C}
\En{0{2.828}}
{\color{red}\Drawline{\A\B}}%
{\color{green}\Drawline{\B\C}}%
\end{zahyou}

```



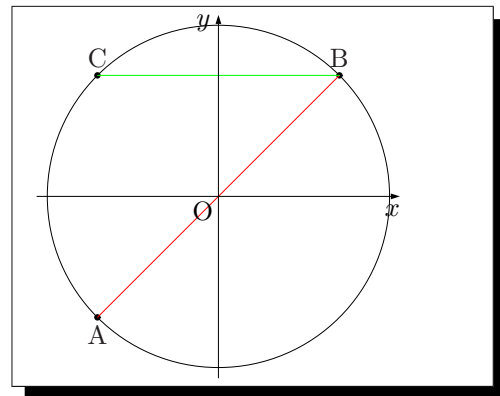
ずれは解消しましたが，うっかり忘れそうですから，別の回避法を紹介しておきます。

iro=red オプション

```

\begin{zahyou}[ul=8mm]
  (-3,3)(-3,3)
\tenretu{
  A(-2,-2)s;B(2,2)n;C(-2,2)n}
\kuromaru{\A;\B;\C}
\En{0{2.828}}
\Drawline<iro=red>{\A\B}
\Drawline<iro=green>{\B\C}
\end{zahyou}

```



<iro=red>オプションによる色変更は，当該コマンドのみに働きます。その使用例を追加しておきます。

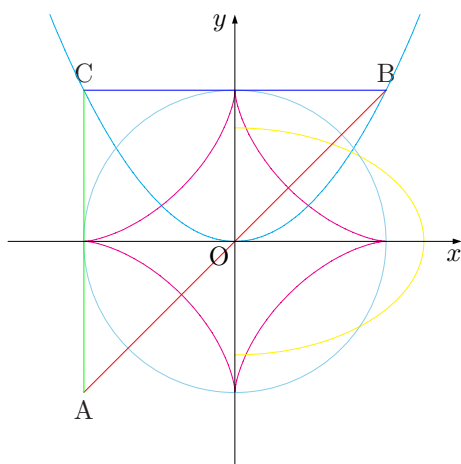
```

\def\byouga{%
  \def\Fx{X*X/2}
  \def\ASTx{2*cos(T)**3}
  \def\ASTy{2*sin(T)**3}
  \tenretu{A(-2,-2)s;B(2,2)n;C(-2,2)n}
  \YGurafu*[iro=cyan]\Fx
  \BGurafu[iro=magenta]\ASTx\ASTy{0}{2*\pi}
  \Put{0}{\Daenko<iro=yellow>{2.5}{1.5}{-90}{90}}
  \En<iro=skyblue>{0{2}}
  \Drawline<iro=red>{\A\B}
  \Drawline<iro=blue>{\B\C}
  \Drawline<iro=green>{\C\A}
}

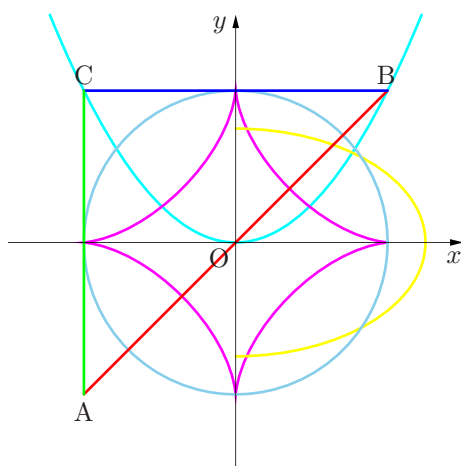
```

を zahyou 環境内で実行すると，





次いで，pszahyou 環境内では



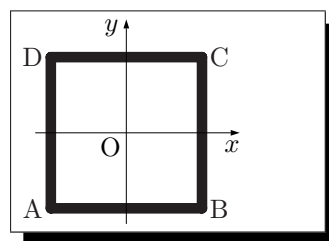
## 2.2 ¥Takakkei

これは新設のコマンドです。いままで多角形を描画するには¥Drawline を用いてきました。これは，pszahyou 環境でももちろん使用可能です。

以下の図では，問題点を強調するため，線幅を太くしています。

従来の多角形描画

```
¥begin{zahyou}[ul=10mm](-1.2,1.5)(-1.2,1.5)
  ¥tenretu{A(-1,-1)w;B(1,-1)e;C(1,1)e;D(-1,1)w}
  {¥allinethickness{4pt}¥Drawline{¥A¥B¥C¥D¥A}}¥%
¥end{zahyou}
```



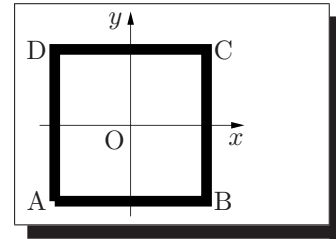


pszahyou 環境での多角形描画

```

\begin{pszahyou}[ul=10mm](-1.2,1.5)(-1.2,1.5)
  \tenretu{A(-1,-1)w;B(1,-1)e;C(1,1)e;D(-1,1)w}
  \setlinewidth{40}%
  \Drawline{\A\B\C\D\A}%
\end{pszahyou}

```



上の2つの図を比較してください。特に頂点 B, C, D のところです。

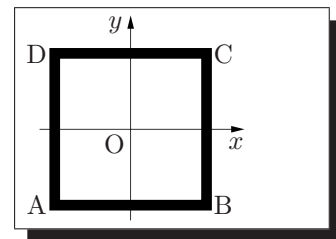
pszahyou 環境を使用したほうが、縦横の線のつながり具合が良くなっていることに気が付きませんか。ただし、A のところはいけませんね。これを改良したのが、今回新設した多角形コマンドです。

\Takakkei

```

\begin{pszahyou}[ul=10mm](-1.2,1.5)(-1.2,1.5)
  \tenretu{A(-1,-1)w;B(1,-1)e;C(1,1)e;D(-1,1)w}
  \setlinewidth{40}%
  \Takakkei{\A\B\C\D}%
\end{pszahyou}

```



さらに、四角形 ABCD を描画するのに、

\Drawline では \Drawline{\A\B\C\D\A}  
 としなければならなかったのですが、  
 \Takakkei では \Takakkei{\A\B\C\D}  
 で済むのも嬉しいことです。

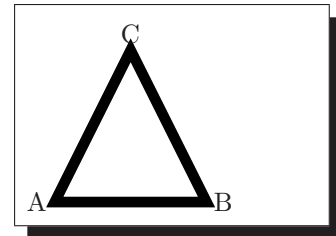
三角形ではどうでしょう。

\三角形

```

\begin{pszahyou*}[ul=10mm](-1.2,1.5)(-1.2,1.5)
  \tenretu{A(-1,-1)w;B(1,-1)e;C(0,1)n}
  \setlinewidth{40}%
  \Takakkei{\A\B\C}%
\end{pszahyou*}

```



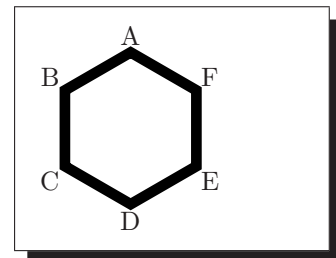
最後に正六角形でのコーナリングを見てみましょう。

\Takakkei

```

\begin{pszahyou*}[ul=10mm](-1.2,1.5)(-1.5,1.5)
  \rtenretu{A(1,90)n;B(1,150)nw;C(1,-150)sw;%
    D(1,-90)s;E(1,-30)se;F(1,30)ne}
  \setlinewidth{40}%
  \Takakkei{\A\B\C\D\E\F}%
\end{pszahyou*}

```



## 2.3 \zahyouMemori

\zahyouMemori の線種・線幅の変更はいままでと異なる方法をとります。

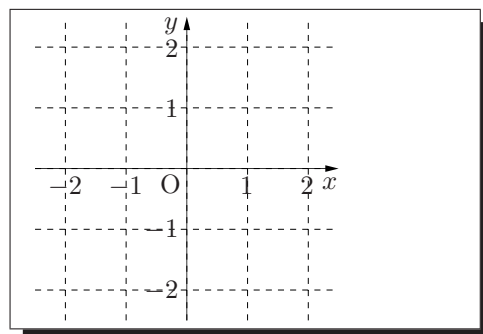


まず、デフォルトの確認です。

```

%zahyouMemori
\begin{pszahyou}[ul=8mm]%
  (-2.5,2.5)(-2.5,2.5)
  %zahyouMemori[g]
\end{pszahyou}

```



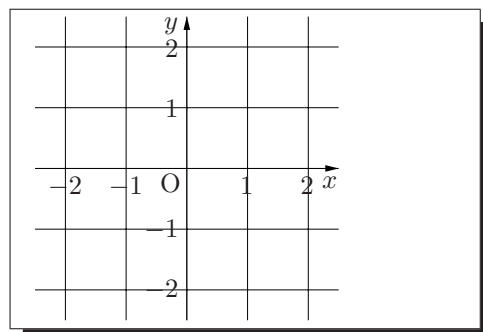
従来、線種を変更するには<sensyu=..>, 線幅を変更するには<allinethickness=..>オプションを用いてきましたが, pszahyou 環境では, %Drawline と同形式のオプションを用います。

線種を実線にしてみましょう。

```

線種変更
\begin{pszahyou}[ul=8mm]%
  (-2.5,2.5)(-2.5,2.5)
  %zahyouMemori[g]<dash={}>
\end{pszahyou}

```

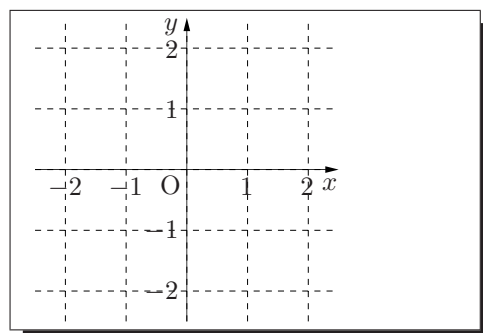


線幅を座標軸と同じにします。

```

線幅変更
\begin{pszahyou}[ul=8mm]%
  (-2.5,2.5)(-2.5,2.5)
  %zahyouMemori[g]<linewidth=3>
\end{pszahyou}

```



## 2.4 %Put の syaei=.. オプション

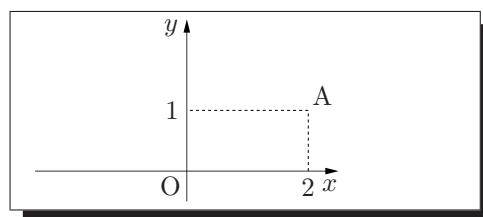
%Put に [syaei=..] オプションをつける場合の, 線種, 線幅の変更も前節と同様です。

まず、デフォルトです：

```

syaei=.. オプション
\begin{pszahyou}[ul=8mm]%
  (-2.5,2.5)(-.5,2.5)
  %tenretu{A(2,1)ne}
  %Put%A[syaei=xy]{ }
\end{pszahyou}

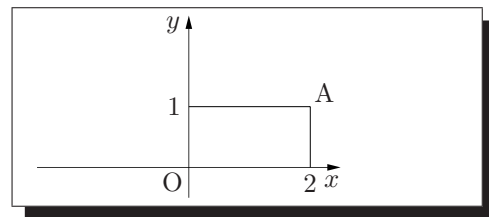
```





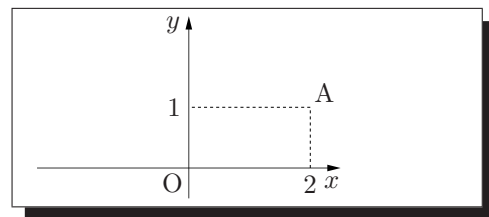
実線にしましょう。

```
syaei=.. オプション
\begin{pszahyou}[ul=8mm]%
  (-2.5,2.5)(-.5,2.5)
  \tenretu{A(2,1)ne}
  \Put%A[syaei=xy,dash={}]{ }
\end{pszahyou}
```



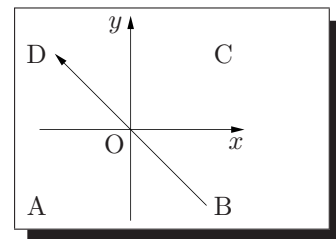
線幅変更です。

```
syaei=.. オプション
\begin{pszahyou}[ul=8mm]%
  (-2.5,2.5)(-.5,2.5)
  \tenretu{A(2,1)ne}
  \Put%A[syaei=xy,linewidth=3]{ }
\end{pszahyou}
```



## 2.5 \ArrowLine

```
\ArrowLine
\begin{pszahyou}[ul=10mm](-1.2,1.5)(-1.2,1.5)
  \tenretu{A(-1,-1)w;B(1,-1)e;C(1,1)e;D(-1,1)w}
  \ArrowLine%B%D
\end{pszahyou}
```

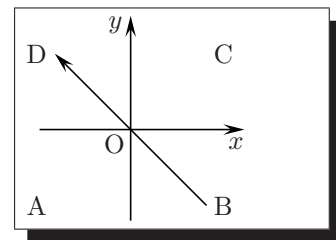


\ArrowLine において、鋸の形状を変更するコマンドは、従来の

\ArrowHeadSize

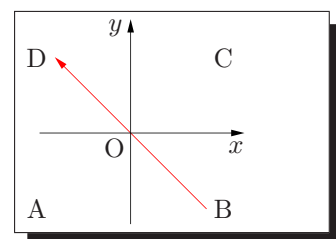
とは別のコマンド \setarrowsize を使用します。その際、<.>オプションをつけると、窪みをつけることができます。

```
\setarrowsize
\begin{pszahyou}[ul=10mm](-1.2,1.5)(-1.2,1.5)
  \tenretu{A(-1,-1)w;B(1,-1)e;C(1,1)e;D(-1,1)w}
  \setarrowsize<.33>{6}{40}{80}
  \ArrowLine%B%D
\end{pszahyou}
```



矢線に色をつけるには、<iro=>オプションを用います。

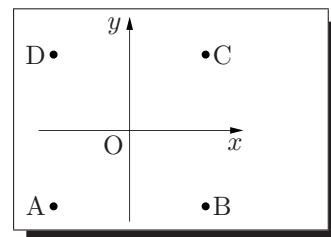
```
\ArrowLine に色付け
\begin{pszahyou}[ul=10mm](-1.2,1.5)(-1.2,1.5)
  \tenretu{A(-1,-1)w;B(1,-1)e;C(1,1)e;D(-1,1)w}
  \ArrowLine<iro=red>%B%D
\end{pszahyou}
```





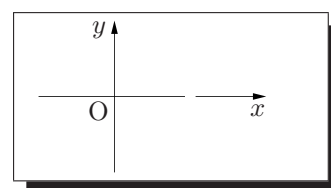
## 2.6 ¥Kuromaru, ¥Siromaru

```
¥kuromaru
¥begin{pszahyou}[ul=10mm](-1.2,1.5)(-1.2,1.5)
  ¥tenretu{A(-1,-1)w;B(1,-1)e;C(1,1)e;D(-1,1)w}
  ¥kuromaru{¥A;¥B;¥C;¥D}
¥end{pszahyou}
```



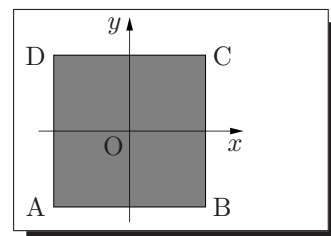
上では, ¥kuromaru を使っていますが, ¥Kuromaru も使用可能です。次は ¥Siromaru の使用例です。座標軸上に白丸を打つには, 座標軸の描画タイミングも問題であることは sampleP.tex でも触れました。

```
¥Siromaru
¥begin{pszahyou*}[ul=10mm](-1,2)(-1,1)
  ¥def¥A{(1,0)}
  ¥drawXYaxis
  ¥setlinewidth{3}
  ¥Siromaru[2pt]¥A
¥end{pszahyou*}
```



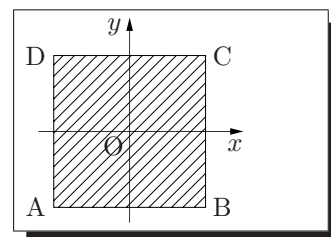
## 2.7 ¥Nuritubusi

```
¥Nuritubusi
¥begin{pszahyou}[ul=10mm](-1.2,1.5)(-1.2,1.5)
  ¥tenretu{A(-1,-1)w;B(1,-1)e;C(1,1)e;D(-1,1)w}
  ¥Nuritubusi{¥A¥B¥C¥D}
  ¥setlinewidth{3}
  ¥Takakkei{¥A¥B¥C¥D}
¥end{pszahyou}
```



斜線塗りも可能です。

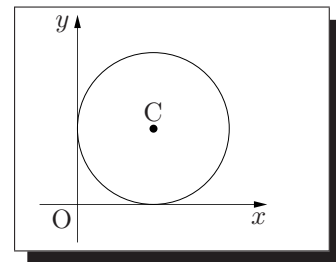
```
¥Nuritubusi*
¥begin{pszahyou}[ul=10mm](-1.2,1.5)(-1.2,1.5)
  ¥tenretu{A(-1,-1)w;B(1,-1)e;C(1,1)e;D(-1,1)w}
  ¥setlinewidth{3}
  ¥Nuritubusi*{¥A¥B¥C¥D}
  ¥Takakkei{¥A¥B¥C¥D}
¥end{pszahyou}
```





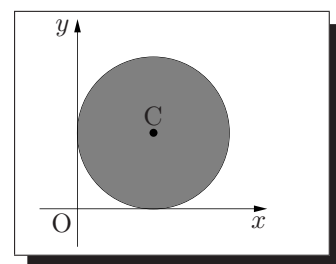
## 2.8 ¥En

```
¥En
¥begin{pszahyou}[ul=10mm](-.5,2.5)(-.5,2.5)
  ¥tenretu{C(1,1)n}
  ¥Kuromaru¥C
  ¥setlinewidth{3}
  ¥En{¥C}{1}
¥end{pszahyou}
```



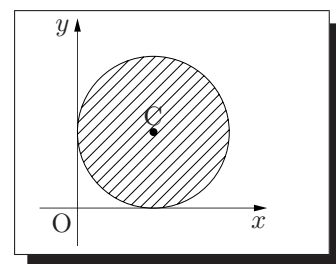
塗りつぶしも可能です。

```
¥En*
¥begin{pszahyou}[ul=10mm](-.5,2.5)(-.5,2.5)
  ¥tenretu{C(1,1)n}
  ¥setlinewidth{3}
  ¥En{¥C}{1}
  ¥En*{¥C}{1}
  ¥Kuromaru¥C
¥end{pszahyou}
```



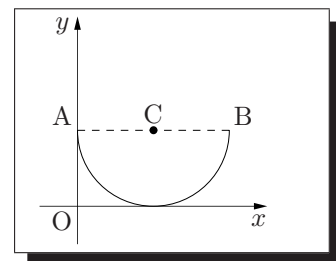
斜線塗りです。

```
¥En**
¥begin{pszahyou}[ul=10mm](-.5,2.5)(-.5,2.5)
  ¥tenretu{C(1,1)n}
  ¥setlinewidth{3}
  ¥En{¥C}{1}
  ¥En**{¥C}{1}
  ¥Kuromaru¥C
¥end{pszahyou}
```



## 2.9 ¥Enko

```
¥Enko
¥begin{pszahyou}[ul=10mm](-.5,2.5)(-.5,2.5)
  ¥tenretu{A(0,1)nw;C(1,1)n;B(2,1)ne}
  ¥Kuromaru¥C
  ¥setlinewidth{3}
  ¥Enko{¥C}{1}{-180}{0}
  ¥Hasen{¥A¥B}
¥end{pszahyou}
```

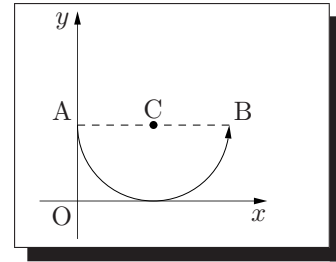


端点に矢印をつけてみましょう。



¥Enko の yazirusi=. オプション

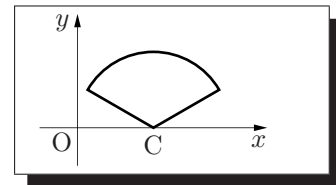
```
¥begin{pszahyou}[ul=10mm](-.5,2.5)(-.5,2.5)
  ¥tenretu{A(0,1)nw;C(1,1)n;B(2,1)ne}
  ¥Kuromaru¥C
  ¥setlinewidth{3}
  ¥Enko<yazirusi=a>{¥C}{1}{-180}{0}
  ¥Hasen{¥A¥B}
¥end{pszahyou}
```



## 2.10 ¥ougigata

¥ougigata

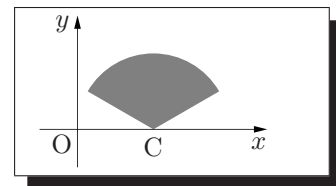
```
¥begin{pszahyou}[ul=10mm](-.5,2.5)(-.5,1.5)
  ¥tenretu{C(1,0)s}
  ¥Put¥C{¥ougigata{1}{30}{150}}
¥end{pszahyou}
```



塗りつぶし，斜線塗りも可能です。

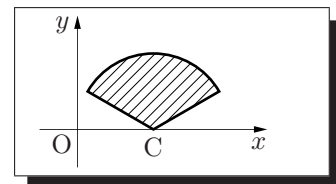
¥ougigata\*

```
¥begin{pszahyou}[ul=10mm](-.5,2.5)(-.5,1.5)
  ¥tenretu{C(1,0)s}
  ¥Put¥C{¥ougigata*{1}{30}{150}}
¥end{pszahyou}
```



¥ougigata\*\*

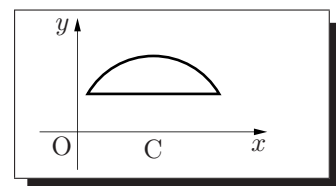
```
¥begin{pszahyou}[ul=10mm](-.5,2.5)(-.5,1.5)
  ¥tenretu{C(1,0)s}
  ¥Put¥C{¥ougigata{1}{30}{150}}
  ¥setlinewidth{1}
  ¥Put¥C{¥ougigata**{1}{30}{150}}
¥end{pszahyou}
```



## 2.11 ¥yumigata

¥yumigata

```
¥begin{pszahyou}[ul=10mm](-.5,2.5)(-.5,1.5)
  ¥tenretu{C(1,0)s}
  ¥Put¥C{¥yumigata{1}{30}{150}}
¥end{pszahyou}
```



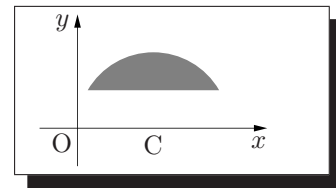
塗りつぶし，斜線塗りも可能です。



```

¥yumigata*
¥begin{pszahyou}[ul=10mm](-.5,2.5)(-.5,1.5)
  ¥tenretu{C(1,0)s}
  ¥Put¥C{¥yumigata*{1}{30}{150}}
¥end{pszahyou}

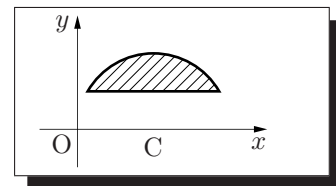
```



```

¥yumigata**
¥begin{pszahyou}[ul=10mm](-.5,2.5)(-.5,1.5)
  ¥tenretu{C(1,0)s}
  ¥Put¥C{¥yumigata{1}{30}{150}}
  ¥setlinewidth{1}
  ¥Put¥C{¥yumigata**{1}{30}{150}}
¥end{pszahyou}

```

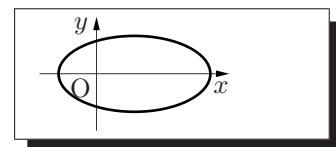


## 2.12 ¥Daen

```

¥Daen
¥begin{pszahyou}[ul=5mm](-1.5,3.5)(-1.5,1.5)
  ¥def¥A{(1,0)}
  ¥Daen¥A{2}{1}
¥end{pszahyou}

```

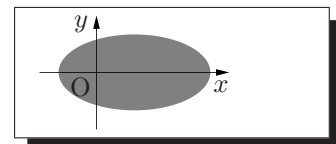


塗りです。

```

¥Daen*
¥begin{pszahyou}[ul=5mm](-1.5,3.5)(-1.5,1.5)
  ¥def¥A{(1,0)}
  ¥Daen*¥A{2}{1}
¥end{pszahyou}

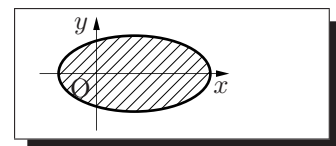
```



```

¥Daen**
¥begin{pszahyou}[ul=5mm](-1.5,3.5)(-1.5,1.5)
  ¥def¥A{(1,0)}
  ¥Daen¥A{2}{1}
  ¥setlinewidth{3}
  ¥Daen**¥A{2}{1}
¥end{pszahyou}

```



点線描画は，¥setdash を利用します。

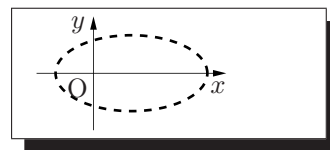


### 点線描画

```

\begin{pszahyou}[ul=5mm](-1.5,3.5)(-1.5,1.5)
  \def\A{(1,0)}
  \setdash{0.2,0.2}
  \Daen\A{2}{1}
  \setdash{}
\end{pszahyou}

```



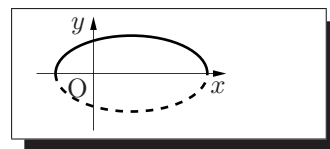
### 2.13 \Daenko

#### \Daenko

```

\begin{pszahyou}[ul=5mm](-1.5,3.5)(-1.5,1.5)
  \def\A{(1,0)}
  \Put\A{\Daenko{2}{1}{0}{180}}
  \setdash{0.2,0.2}
  \Put\A{\Daenko{2}{1}{-180}{0}}
  \setdash{}
\end{pszahyou}

```



### 2.14 \YGurafu

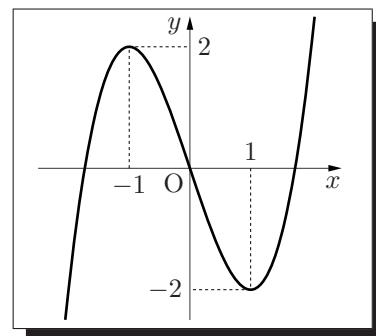
$y = f(x)$  のグラフを描画するコマンド `\YGurafu` も `pszahyou` 環境で使うことができます。一例として,  $y = x^3 - 3x$  のグラフをかいてみましょう。

#### \YGurafu

```

\begin{pszahyou}[ul=8mm](-2.5,2.5)(-2.5,2.5)
  \tenretu*{A(-1,2);B(1,-2)}
  \def\Fx{X*(X*X-3)}
  \YGurafu*\Fx
  \setlinewidth{1}
  \Put\A[syaei=xy]{}
  \Put\B[syaei=xy]{}
\end{pszahyou}

```

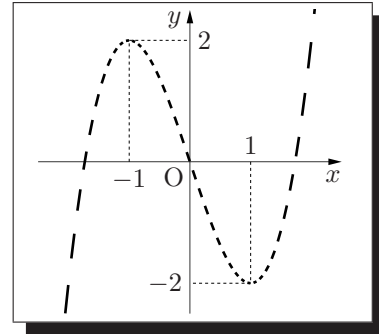


グラフの破線描画オプションも使用可能です。



### ¥YGurafu の破線描画

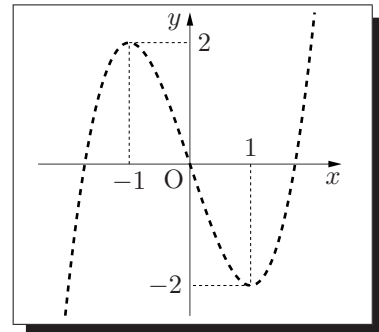
```
¥begin{pszhayou}[ul=8mm](-2.5,2.5)(-2.5,2.5)
  ¥tenretu*{A(-1,2);B(1,-2)}
  ¥def¥Fx{X*(X*X-3)}
  ¥YGurafu*(.1)(.05)¥Fx
  ¥setlinewidth{1}
  ¥Put¥A[syaei=xy]{}
  ¥Put¥B[syaei=xy]{}
¥end{pszhayou}
```



しかし, pszhayou 環境では, ¥setdash を利用した方が良いでしょう。

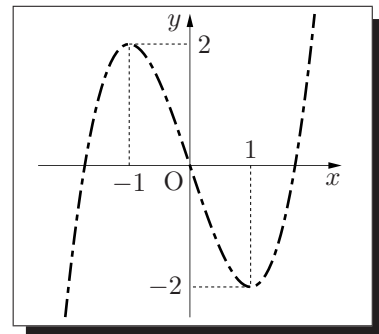
### ¥setdash による破線描画

```
¥begin{pszhayou}[ul=8mm](-2.5,2.5)(-2.5,2.5)
  ¥tenretu*{A(-1,2);B(1,-2)}
  ¥def¥Fx{X*(X*X-3)}
  ¥setdash{0.1,0.1}
  ¥YGurafu*¥Fx
  ¥setlinewidth{1}
  ¥Put¥A[syaei=xy]{}
  ¥Put¥B[syaei=xy]{}
¥end{pszhayou}
```



### ¥setdash による鎖線描画

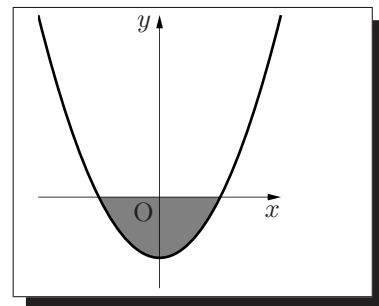
```
¥begin{pszhayou}[ul=8mm](-2.5,2.5)(-2.5,2.5)
  ¥tenretu*{A(-1,2);B(1,-2)}
  ¥def¥Fx{X*(X*X-3)}
  ¥setdash{0.3,0.1,0.1,0.1}
  ¥YGurafu*¥Fx
  ¥setlinewidth{1}
  ¥Put¥A[syaei=xy]{}
  ¥Put¥B[syaei=xy]{}
¥end{pszhayou}
```



## 2.15 ¥YNuri

### ¥YNuri

```
¥begin{pszhayou}[ul=8mm](-2,2)(-1.5,3)
  ¥def¥Fx{X*X-1}
  ¥YNuri¥Fx{-1}{1}
  ¥YGurafu*¥Fx
¥end{pszhayou}
```



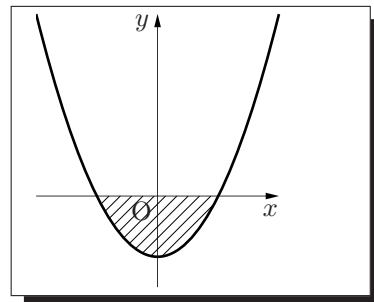


斜線塗りです。

```

%YNuri*
\begin{pszahyou}[ul=8mm](-2,2)(-1.5,3)
  \def\Fx{X*X-1}
  \YGurafu*\Fx
  \setlinewidth{3}
  %YNuri*\Fx{-1}{1}
\end{pszahyou}

```

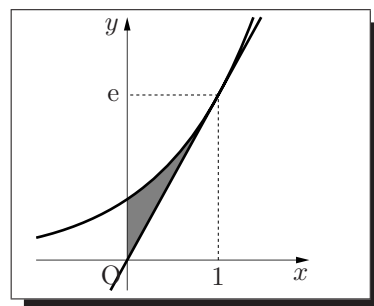


## 2.16 %YNurii

```

%YNurii
\begin{pszahyou}[ul=8mm,xscale=1.5](-1,2)(-.5,4)
  \tenretu*{T(1,%Napier)}
  \def\Fx{exp(X)}
  \def\Gx{%Napier*X}
  %YNurii*\Fx\Gx{0}{1}
  \YGurafu*\Fx
  \YGurafu*\Gx
  \setlinewidth{3}
  \Put\T[syaei=xy,ylabel=$e$]{}
\end{pszahyou}

```



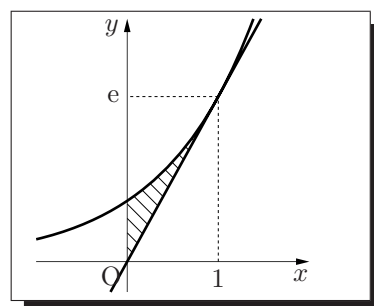
pszahyou 環境のもとになっている zahyou 環境は, emathPxy.sty のそれですから, xscale, yscale が有効 (のはず) です。

斜線塗りです。

```

%YNurii*
\begin{pszahyou}[ul=8mm,xscale=1.5](-1,2)(-.5,4)
  \tenretu*{T(1,%Napier)}
  \def\Fx{exp(X)}
  \def\Gx{%Napier*X}
  \YGurafu*\Fx
  \YGurafu*\Gx
  \setlinewidth{3}
  %YNurii*[-45]\Fx\Gx{0}{1}
  \Put\T[syaei=xy,ylabel=$e$]{}
\end{pszahyou}

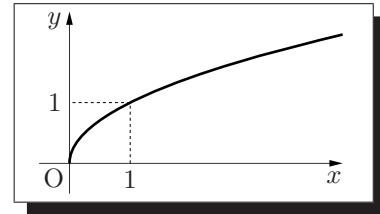
```





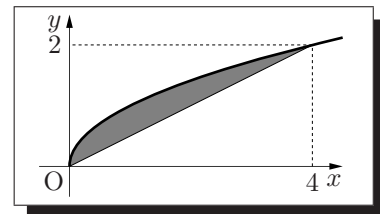
## 2.17 $\text{\textbackslash XGurafu}$

```
 $\text{\textbackslash XGurafu}$ 
 $\text{\textbackslash begin{pszahyou}[ul=8mm](-.5,4.5)(-.5,2.5)}$ 
 $\text{\textbackslash tenretu*{A(1,1)}}$ 
 $\text{\textbackslash def\Fy{Y*Y}}$ 
 $\text{\textbackslash XGurafu*[sitay=0]\Fy}$ 
 $\text{\textbackslash setlinewidth{1}}$ 
 $\text{\textbackslash Put\A[syaei=xy]\{}}$ 
 $\text{\textbackslash end{pszahyou}}$ 
```

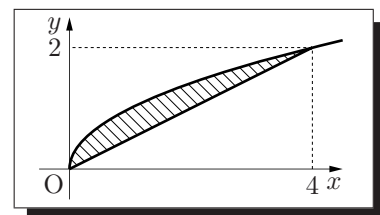


## 2.18 $\text{\textbackslash XNuri}$

```
 $\text{\textbackslash XNuri}$ 
 $\text{\textbackslash begin{pszahyou}[ul=8mm](-.5,4.5)(-.5,2.5)}$ 
 $\text{\textbackslash tenretu*{A(4,2)}}$ 
 $\text{\textbackslash def\Fy{Y*Y}}$ 
 $\text{\textbackslash XNuri\Fy{0}\{2}}$ 
 $\text{\textbackslash XGurafu*[sitay=0]\Fy}$ 
 $\text{\textbackslash setlinewidth{3}}$ 
 $\text{\textbackslash Drawline{\O\A}}$ 
 $\text{\textbackslash Put\A[syaei=xy]\{}}$ 
 $\text{\textbackslash end{pszahyou}}$ 
```



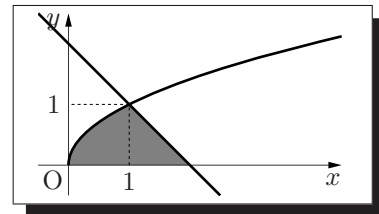
```
 $\text{\textbackslash XNuri*}$ 
 $\text{\textbackslash begin{pszahyou}[ul=8mm](-.5,4.5)(-.5,2.5)}$ 
 $\text{\textbackslash tenretu*{A(4,2)}}$ 
 $\text{\textbackslash def\Fy{Y*Y}}$ 
 $\text{\textbackslash XGurafu*[sitay=0]\Fy}$ 
 $\text{\textbackslash Drawline{\O\A}}$ 
 $\text{\textbackslash setlinewidth{3}}$ 
 $\text{\textbackslash XNuri*[-45]\Fy{0}\{2}}$ 
 $\text{\textbackslash Put\A[syaei=xy]\{}}$ 
 $\text{\textbackslash end{pszahyou}}$ 
```



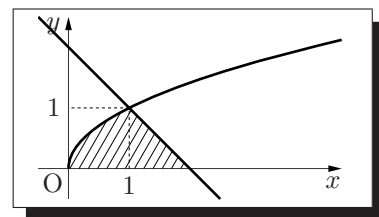


## 2.19 ¥XNurii

```
¥XNuri
¥begin{pszahyou}[ul=8mm](-.5,4.5)(-.5,2.5)
  ¥tenretu*{A(1,1)}
  ¥def¥Fy{Y*Y}
  ¥def¥Gy{2-Y}
  ¥XNurii¥Fy¥Gy{0}{1}
  ¥XGurafu*[sitay=0]¥Fy
  ¥XGurafu*¥Gy
  ¥setlinewidth{3}
  ¥Put¥A[syaei=xy]{ }
¥end{pszahyou}
```

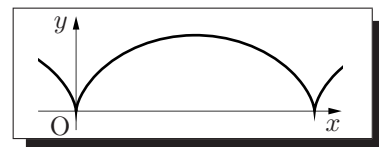


```
¥XNurii*
¥begin{pszahyou}[ul=8mm](-.5,4.5)(-.5,2.5)
  ¥tenretu*{A(1,1)}
  ¥def¥Fy{Y*Y}
  ¥def¥Gy{2-Y}
  ¥XGurafu*[sitay=0]¥Fy
  ¥XGurafu*¥Gy
  ¥setlinewidth{3}
  ¥XNurii*[60]¥Fy¥Gy{0}{1}
  ¥Put¥A[syaei=xy]{ }
¥end{pszahyou}
```



## 2.20 ¥BGurafu

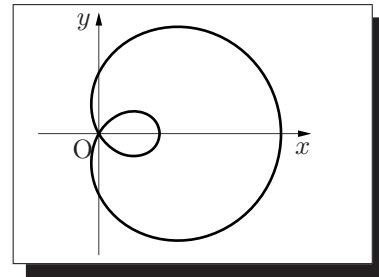
```
¥BGurafu
¥begin{pszahyou}[ul=5mm](-1,7)(-.5,2.5)
  ¥def¥Ft{T-sin(T)}
  ¥def¥Gt{1-cos(T)}
  ¥BGurafu¥Ft¥Gt{-¥pi}{3*¥pi}
¥end{pszahyou}
```





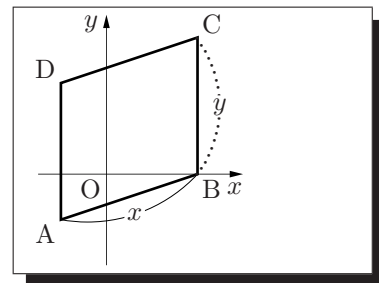
## 2.21 ¥RGurafu

```
¥RGurafu
¥begin{pszahyou}[ul=8mm](-1,3.5)(-2,2)
  ¥def¥Ft{1+2*cos(T)}
  ¥RGurafu¥Ft{0}{2*¥pi}
¥end{pszahyou}
```



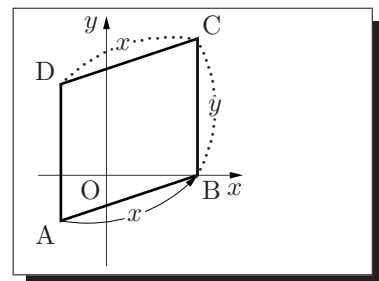
## 2.22 ¥HenKo

```
¥Hen_ko
¥begin{pszahyou}[ul=6mm](-1.5,3)(-2,3.5)%
  ¥tenretu{A(-1,-1)sw;B(2,0)se;
    C(2,3)ne;D(-1,2)nw}
  ¥Takakkei{¥A¥B¥C¥D}
  ¥setlinewidth{2}
  ¥HenKo¥A¥B{¥x¥}
  ¥HenKo[20]<.5>¥B¥C{¥y¥}
¥end{pszahyou}
```



弧の端に矢印をつけるオプションも使用可能です。  
矢印付きの ¥HenKo

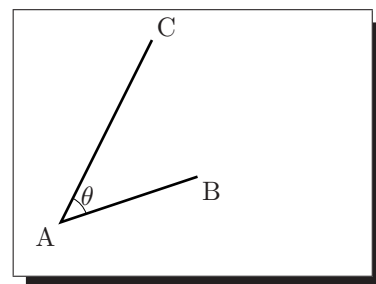
```
¥begin{pszahyou}[ul=6mm](-1.5,3)(-2,3.5)%
  ¥tenretu{A(-1,-1)sw;B(2,0)se;
    C(2,3)ne;D(-1,2)nw}
  ¥Takakkei{¥A¥B¥C¥D}
  ¥setlinewidth{2}
  ¥HenKo<yazirusi=a>¥A¥B{¥x¥}
  ¥HenKo[20]<yazirusi=r>¥B¥C{¥y¥}
  ¥HenKo[20]<yazirusi=b>¥C¥D{¥x¥}
¥end{pszahyou}
```





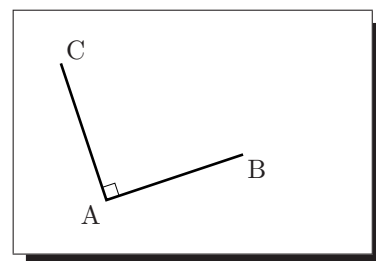
## 2.23 ¥Kakukigou

```
¥Kakukigou
¥begin{pszahyou*}[ul=6mm](-1.5,3)(-2,3.5)%
  ¥tenretu{A(-1,-1)sw;B(2,0)se;C(1,3)ne}
  ¥setlinewidth{3}
  ¥Kakukigou¥B¥A¥C{¥¥theta$}
  ¥setlinewidth{10}
  ¥Drawline{¥B¥A¥C}
¥end{pszahyou*}
```



## 2.24 ¥Tyokkakukigou

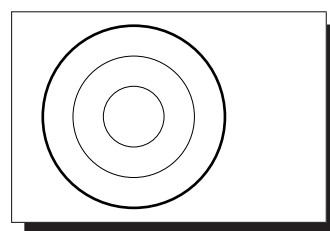
```
¥Tyokkakukigou
¥begin{pszahyou*}[ul=6mm](-2.5,3)(-2,3)%
  ¥tenretu{A(-1,-1)sw;B(2,0)se;C(-2,2)ne}
  ¥setlinewidth{3}
  ¥Tyokkakukigou¥B¥A¥C
  ¥setlinewidth{10}
  ¥Drawline{¥B¥A¥C}
¥end{pszahyou*}
```



## 2.25 emathPs における線幅，線種の変更

emathPs.sty における線幅の変更をする ¥setlinewidth, あるいは描画線を破線にする ¥setdash はグローバルに影響し，グルーピングは無効です。そのあたりを確認しておきます。

```
¥setlinewidth
¥begin{pszahyou*}[ul=4mm](-3.2,3.2)(-3.2,3.2)
  ¥En¥0{3}
  {¥setlinewidth{1}¥En¥0{2}}%
  ¥En¥0{1}
¥end{pszahyou*}
```



同心円が3個描かれています。一番外側の円（半径3）がデフォルトの線幅です。真中の円（半径2）を ¥setlinewidth{1} として，細線で描画し，その部分をグルーピングして，一番中の円（半径1）はデフォルトの線幅に戻るか，と思いきや，細線のままでグルーピングの効はありませんでした。

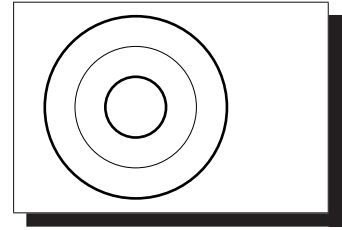
これは，emathPs.sty においては，描画コマンドは T<sub>E</sub>X ではなく，PostScript に翻訳される関係で，T<sub>E</sub>X の管理外となり，グルーピングは無効となってしまう，とご理解願います。

では，どうするかというと，¥gsave コマンドで描画条件をセーブし，¥grestore でそれを復元する，という手順を踏むことになります。



¥gsave と ¥grestore

```
¥begin{pszahyou*}[ul=4mm](-3.2,3.2)(-3.2,3.2)
  ¥En¥0{3}
  ¥gsave
    ¥setlinewidth{1}¥En¥0{2}
  ¥grestore
  ¥En¥0{1}
¥end{pszahyou*}
```



今度は、真中の円だけが細線で描画されました。

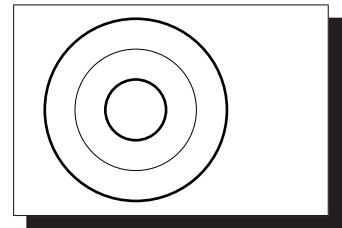
しかし、局所的な変更の場合には、描画コマンド（ここでは、¥En）に対するオプション引数を与えることで処理できないだろうか、というのが今回のバージョンアップの目論見です。

### 2.25.1 ¥En などの場合

<linewidth=..>オプション ¥En に対して、オプション引数<linewidth=1>を与えることで、局所的な変更を実現しました。

¥En への<linewidth=..>オプション

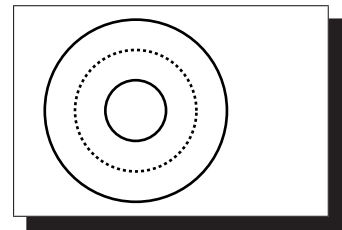
```
¥begin{pszahyou*}[ul=4mm](-3.2,3.2)(-3.2,3.2)
  ¥En¥0{3}
  ¥En<linewidth=1>¥0{2}
  ¥En¥0{1}
¥end{pszahyou*}
```



<dash=..>オプション 線種を変更するオプション<dash={.1,.1}>も局所的に効きます。

¥En への<dash=..>オプション

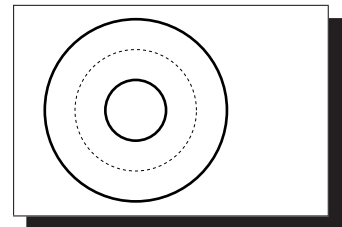
```
¥begin{pszahyou*}[ul=4mm](-3.2,3.2)(-3.2,3.2)
  ¥En¥0{3}
  ¥En<dash={.1,.1}>¥0{2}
  ¥En¥0{1}
¥end{pszahyou*}
```



両者の併用も可能です。

併用

```
¥begin{pszahyou*}[ul=4mm](-3.2,3.2)(-3.2,3.2)
  ¥En¥0{3}
  ¥En<linewidth=1,dash={.1,.1}>¥0{2}
  ¥En¥0{1}
¥end{pszahyou*}
```



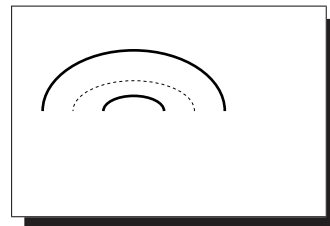
このオプションは、¥En だけではなく、¥Enko、¥Daen、¥Daenko に対しても有効です。¥Daenko に適用した例です。



```

%Daneko
\begin{pszhayou}[ul=4mm](-3.2,3.2)(-3.2,3.2)
  \Put%0{\%Daenko{3}{2}{0}{180}}
  \Put%0{\%Daenko<linewidth=1,dash={.1,.1}>%
    {2}{1}{0}{180}}
  \Put%0{\%Daenko{1}{.5}{0}{180}}
\end{pszhayou}

```



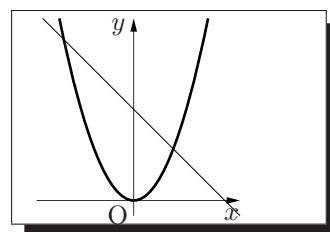
## 2.25.2 ¥YGurafu などの場合

グラフ描画コマンド ¥YGurafu などの局所的な変更をする [linewidth=...] オプションも有効としました。

```

¥YGurafu の線幅変更
\begin{pszhayou}[ul=4mm](-3.2,3.5)(-.5,6)
  \def¥Fx{X*X}
  \def¥Gx{3-X}
  ¥YGurafu*[linewidth=1]¥Gx
  ¥YGurafu*¥Fx
\end{pszhayou}

```

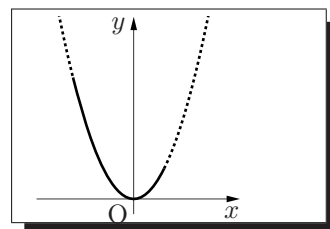


[dash=...] オプションも有効です。

```

¥YGurafu の線種変更
\begin{pszhayou}[ul=4mm](-3.2,3.5)(-.5,6)
  \def¥Fx{X*X}
  ¥YGurafu[dash={.1,.1}]¥Fx¥xmin{-2}
  ¥YGurafu¥Fx{-2}{1}
  ¥YGurafu[dash={.1,.1}]¥Fx{1}¥xmax
\end{pszhayou}

```

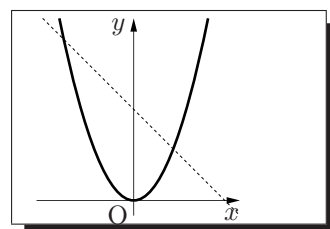


併用も可能です。

```

併用
\begin{pszhayou}[ul=4mm](-3.2,3.5)(-.5,6)
  \def¥Fx{X*X}
  \def¥Gx{3-X}
  ¥YGurafu*[linewidth=1,dash={.1,.1}]¥Gx
  ¥YGurafu*¥Fx
\end{pszhayou}

```

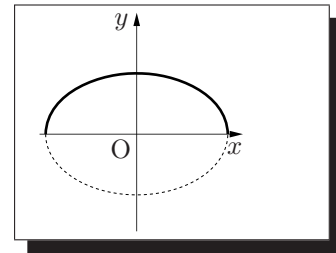


¥XGurafu, ¥BGurafu, ¥RGurafu にも適用されます。



¥BGurafu の線種変更

```
¥begin{pszahyou}[ul=4mm](-3.2,3.5)(-3.2,4)
  ¥def¥Ft{3*cos(T)}
  ¥def¥Gt{2*sin(T)}
  ¥BGurafu[linewidth=1,dash={.1,.1}]%
    ¥Ft¥Gt{-¥pi}{0}
  ¥BGurafu¥Ft¥Gt{0}{¥pi}
¥end{pszahyou}
```

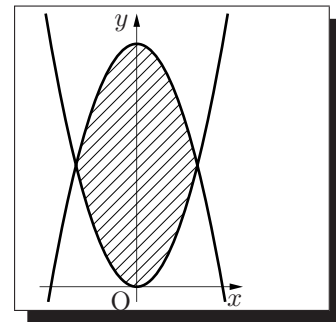


### 2.25.3 斜線塗りの場合

¥YNurii\*など、斜線塗りにおける斜線の太さ、線種の局所的な変更を<linewidth=...,dash=...>オプションで可能としました。

斜線塗りの線種変更

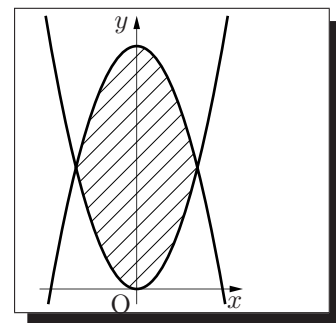
```
¥begin{pszahyou}[ul=4mm](-3.2,3.5)(-.5,9)
  ¥def¥Fx{X*X}
  ¥def¥Gx{8-X*X}
  ¥YNurii*<linewidth=1>¥Fx¥Gx{-2}{2}
  ¥YGurafu*¥Fx
  ¥YGurafu*¥Gx
¥end{pszahyou}
```



斜線塗りに対する<...>オプションは、本来、斜線の間隔を調整するためのものでした。その機能を併用するときは、<syaurikankaku=...>オプションを用います。デフォルト値は 0.125 で、¥unitlength に依存します。

斜線間隔の変更

```
¥begin{pszahyou}[ul=4mm](-3.2,3.5)(-.5,9)
  ¥def¥Fx{X*X}
  ¥def¥Gx{8-X*X}
  ¥YNurii*<linewidth=1,syaurikankaku=.4>%
    ¥Fx¥Gx{-2}{2}
  ¥YGurafu*¥Fx
  ¥YGurafu*¥Gx
¥end{pszahyou}
```



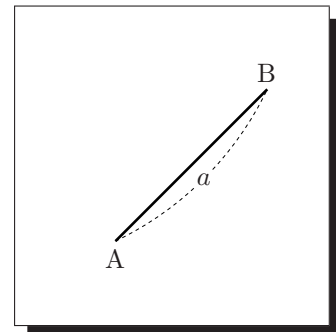
### 2.25.4 ¥HenKo の場合

¥HenKo の線幅・線種を変更するオプションも用意しました。



#### ¥Hen\_ko の線種変更

```
¥begin{pszahyou*}[ul=10mm](0,4)(0,4)
  ¥tenretu{A(1,1)s;B(3,3)n}
  ¥HenKo<linewidth=1,dash={.05,.05}>¥A¥B{¥a$}
  ¥Drawline{¥A¥B}
¥end{pszahyou*}
```



元来、このオプション<#2>は、線分と弧の離れ具合を指定するものでした。それも使用したければ <henkohi=..> オプションを用います。右辺値が1より大きければ、弧は線分より遠ざかります。あるいは<henkoH=..>オプションを用いるのもあります。

#### ¥HenKo の線種変更

```
¥begin{pszahyou*}[ul=10mm](0,4)(0,4)
  ¥tenretu{A(1,1)s;B(3,3)n}
  ¥HenKo<linewidth=1,dash={.05,.05},henkohi=2>%
    ¥A¥B{¥a$}
  ¥Drawline{¥A¥B}
¥end{pszahyou*}
```

