PSTricks easy document

11 Jan 2019 3:15 a.m. Yukio Yoshida

この文書は, TEX を使っての PSTricks の簡易解説です. また, 絵画を描くというある意味での私的利用の為に, PSTricks の膨大量の command の解説では有りません.

通常, 絵画を描くときは筆で描画 line を決めていきます. また, この描画 line はあらゆる grafical software のもっとも main の顔だと思っています. 従って, この文書の目的は, 絵画の為の数種類の line-command を列挙して, PSTricks でその持つ意味を私なりの解釈での文書にしたものです.

PSTricks で TEX を使っての私的な意味は, 単に絵画を描く場合には LATEX2ε の文書清書環境 (or 出版関係者の製本用) 書式を使ってではなく, 単なるサイズ不定の画用紙を念頭に置いての, 私の勝手な独断的なものです. 人生余暇の趣味として, TEX の表現 (各人の工芸・手芸・マジック・手品・トリック) での drive をも意味すると思います (参考書; "TEX by Topic" *and* "明快 TEX" この二冊は座右の書です).

PSTricks の内部は PostScript コードであり, その育成された dvi は pstricks.con に因って obje への実態像として現われます.

この文書は, "The LATEX Graphics Companion Second Edition"を参照しながら書いた物です. また, その example をそのまま載せたり, 一部描画を変えて載せたりと TEX 使用で行いました.

TEX or pTEX での PSTricks の宣言は, 下記の如くです.

```
%– Preamble(プリアンブル) ————-
\input pstricks.tex
\input pst-grad.tex   (PSTricks の追加パッケージ. グラデーションを使うため)
\input tty-colorsdvi.tex   (私的追加の絵の具ファイル [rgb-color モデル])
\input colordvi.tex   (cmyk-color モデルリスト表示の為に)
%–Body Start ————————
\pspicture(0,0)(4,4)   (4cm 正方形の領域を宣言場所に確保; カレント位置から上向に.)
or, \pspicture(0,0)(4,-4)   (4cm 正方形の領域を宣言場所に確保; カレント位置から下向きへ.)
        この中に PSTricks のコマンドでマークアップして行く
\endpspicture      (描画領域終了,pstricks's コマンドは all-off になります.)
.....................
\bye   (TEX or pTEX での文章作成終了.)
```

— これを例えば, name.tex で save する.—

run!は (角藤版)W32TEX + Ghostscript(gs) 環境下で (windows OS cmd.exe);

```
>ptex name.tex
>ptex name.tex   (念の為, 再度 running.)
>dvips -P dl name.dvi   or >dvips -P pdf name.dvi
32bit 使用なら, >gswin32 name.ps     or >gsview32 name.ps
64bit 使用なら, >gswin64 name.ps     or >gsview64 name.ps
```

で表示印刷できます (通常は pdf にて印刷です).

```
>call ps2pdf name.ps name.pdf
```

で pdf ファイルが作成されます.

PSTricks は基本的に LATEX2ε 用の grafical software である事を忘れないでください.
(角藤版)W32TEX は, 64bit(OS) で running 行う*.exe も用意されています.

文書作成者; 吉田征夫

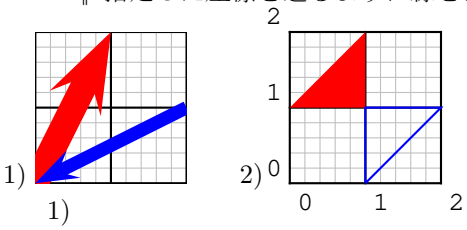
PSTricks パッケージにおいての, 特殊追加パッケージの `\input` 呼び込みは, 呼び込むパッケージ順番によって左右される事も有ります, さらに, パッケージファイルの version に因っても整合性が左右されます. TEX へ PSTRicks の特殊追加パッケージを入れたのなら, 必ず備忘録に記録しておくことを薦めます. 新規は原稿は通っても, 過去の作成原稿ファイルが通るとは保障できません. 整合性が合っていれば問題なく全て通ります.

† この文書内の color は TEX color-list-page 以外, rgb モデルの tty-colorsdvi.tex の値です.

5.6) Lines and polygons(page 231)
Lines represent a main feature in any grafical software, and PSTricks is no eception.

- **\psline*[settings]{arrows}(X0,Y0)(X1,Y1).....(Xn,Yn)**

¶ 指定した座標を通るように線を引く.



1) line の描画.
2) end の (Xn,Yn) を描画 start 座標点に繋げると, \psline は多角形 (polygones) になる.

1)
```
\pspicture*(0,0)(2,2)\psgrid
\psline[linecolor=red,linewidth=10pt]{->}(0,0)(1,2)
\psline[linecolor=blue,linewidth=5pt]{<-}(0,0)(2,1)
\endpspicture
```

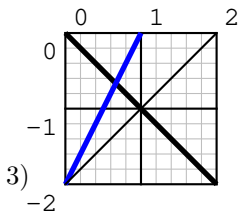
2)
```
\pspicture(0,0)(2,2)\psgrid
\psline*[linecolor=red](0,1)(1,2)(1,1)
\psline[linecolor=blue](1,0)(1,1)(2,1)(1,0)
\endpspicture
```

[settings] の key とその値, {arrows} の値に関しては tex-pstricks.pdf の方で参照してください. これらには, 半角空白での区切りや, 半角空白の挿入は絶対禁止と思った方が問題を避けます.
[setting] や {arrows} を省略するとデフォルト値が使われます.

また, PSTricks の描画空間は TEX の空間ではありません. TEXrun メモリー上では認識されていません. 上図例では参照点が左下ですが, 例えば (上図で)(0,0)(1,-2) とすれば参照点 (0,0) は左上側になります. もちろん, 空間確保の上か下かで座標 (x,y) 点が変わってきます (いずれでも, TEX の参照点は図形の下側にあります).

- **\qline(X1,Y1)(X2,Y2)**

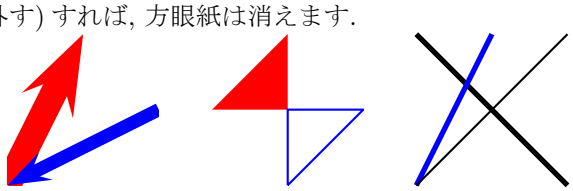
¶ \psline の単純形, key setting を持たず, それらは {\psset{ }} でおこなう.



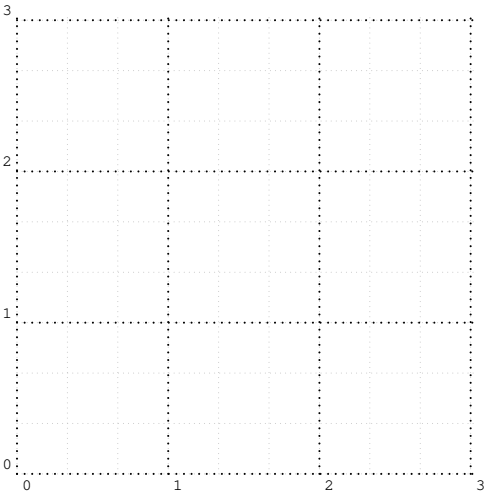

```
\pspicture(0,0)(2,-2)\psgrid
      {
        \psset{linewidth=2pt}\qline(0,0)(2,-2)
        \psset{linecolor=blue}\qline(0,-2)(1,0)
      }
    \qline(0,-2)(2,0)
\endpspicture
```

この場合, \psset{} は必ず { } で囲った中で行うこと. \psset{ } はグローバル・コマンドで pspicture 内の全体を支配します (上図の例; 3 番目の \qline(0,-2)(2,0) はデフォルト値 [black,0.8pt] に戻って線を描画しています).

1), 2), 3) の組版位置が TEX 側の参照点 (baseline) です.
PSTricks の描画領域宣言 \pspicture* の"*"は領域のはみ出し部分を消します (例 1), 2), 3) 参照).
\psline* の"*"は描画終点を始点と自動的に結び, その内部を指定色で塗りつぶします (例 2) 参照).
\psgrid をコメントアウト (外す) すれば, 方眼紙は消えます.



PSTricks の方眼紙 unit は 1cm(デフォルト) ですが, この unit は \psset で任意に設定可能ですし, subgrid も同じく可能です.



¶ **PSTricks の unit(単位) の変更**

```
\psset{unit=2cm}
\newpsobject{mygrid}{psgrid}{subgriddiv=3,%
griddots=20,subgriddots=10,gridlabels=6pt}
\pspicture(0,0)(3,3)\mygrid
\endpspicture
```

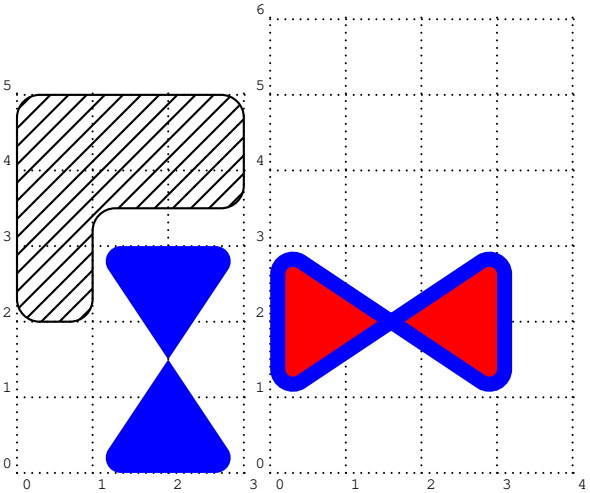
\psset{xunit=.5cm,yunit=2cm} 等々と x 軸,y 軸個別設定も可能です.
%は以後を PSTricks は無視します.
griddots=20 を省略すると, grid はデフォルト描線です.

\newpsobject 式は, デフォルト \psgrid のコマンド定義へ変更設定を入れ, それを \mygird(コマンド) と命名にカスタマイズをしたものです (新規コマンド名前は任意命名です).
unit を \pspicture の外側で指定しているので, これ以後に呼び出す \pspicture の全てに作用します (但し, 文章記述が完全に TEX 側に戻ってしまえば PSTricks の今までのパラメターはクリアー (default) されます. 例ですと再度, \mygrid を記述しなければなりません). \pspicture の 内側で宣言すれば作用はその領域内だけです.

New コマンドを作成する書式は
\newpsobject{new-name}{old-command}{key=value,key=value, ...} です.
(ここで old-command とは, existing(or default) command のことです)

- **\pspolygon*[setting](X0,Y0)(X1,Y1).....(Xn,Yn)**

  ¶ 指定した座標を通るように線を引き. そのオブジェクトを塗りつぶせるようにパス (自動で最終点と開始点) を閉じる (*で塗りつぶす).




```
\newpsobject{mygrid}{psgrid}{subgriddiv=1,%
griddots=10,gridlabels=6pt}
\pspicture(0,0)(3,5)\mygrid
%%\pspolygon[linewidth=1.5pt](0,2)(1,0)
\pspolygon[linearc=0.2,linestyle=none,fillstyle=solid,%
fillcolor=blue,swapaxes=true](0,1)(0,3)(3,1)(3,3)
\pspolygon[fillstyle=hlines,linearc=0.3]%
(0,2)(0,5)(3,5)(3,3.5)(1,3.5)(1,2)
\endpspicture
\quad
\pspicture(0,0)(4,6)\mygrid
\pspolygon[linearc=0.2,linewidth=.2,linecolor=blue,%
fillcolor=red,fillstyle=solid]%
(0.1,1)(0.1,3)(3.1,1)(3.1,3)
\endpspicture
```

二つの方眼紙を繋いでいる空白 \quad は TEX のコマンドです.

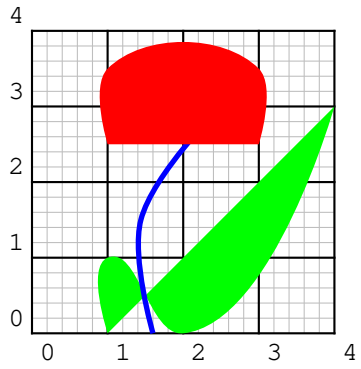
linearc=0.2 は角度です.
linestyle=none は line 描線が無色です. swapaxes=true は座標の値が x 軸と y 軸の反転値になります (塗りつぶした図形の座標点を参照).
fillstyle=hlines は塗りつぶす内容 (style) の値は hlines の意味です.
fillstyle=solid は塗りつぶす設定です.
fillcolor=blue は塗るつぶす色は blue です.
例えば, [linewidth=0.8pt] と [linewidth=0.8] は違います, 前者 (pt) は絶対値ですが, 後者 (0.8) は current の linewidth を 0.8 倍せよのことです.
座標点 (0,0) に限っては省略可能です. 多くの解説参考本の example の例では, この (0,0) が省略されたソースコードの記述を目にすると思います.

\pspolygon* や \psline* 等々で塗りつぶすときは, その時 (current) の linecolor の値です.
[linestyle=none,fillstyle=solid,fillcolor=red] は \pspolygon* や \psline* 等々の"*"を用いずに範囲内部を fillcolor の値で塗りつぶします.

描画はどのようであれ, たえず current 描画,current 描画, と上描きして行きます. 多くは, この特性を利用して絵にしていきます.

この特性を利用するとは, 絵のターゲットに対して, 最後まで残る部分は繊細さを要旨しますが, それ以外はかなりいい加減な描画で OK. このかなりいい加減な描画部分は, いずれは繊細な上書で消されてしまうからです.

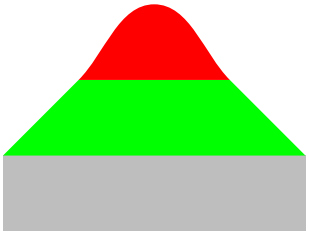
- \pscurve*[settings]{arrows}(X1,Y2)(X2,Y2).....(Xn,Yn)
  ¶ ポイント間に曲線を描く ("*" で自動的に閉じ内部を塗りつぶす).
- \psecurve*[settings{arrows}(X1,Y2)(X2,Y2).....(Xn,Yn)
  ¶ 開いた曲線を引くが, 最初と最後の点を省略する ("*" で自動的に閉じ内部を塗りつぶす.)
- \psccurve*[settings]{arrows}(X1,Y2)(X2,Y2).....(Xn,Yn)
  ¶ ポイント間に閉じた曲線を描く ("*" で内部を塗りつぶす.)



\pscurve の example
```
\pspicture(0,0)(4,4)\psgrid
\pscurve*[linecolor=green](1,0)(1,1)(2,0)(4,3)
\pscurve[linecolor=blue,linewidth=2pt](1.6,0)(1.44,1.5)(2.5,3)
\pscurve[linestyle=none,fillstyle=solid,fillcolor=red]%
(1,2.5)(1,3.5)(3,3.5)(3,2.5)
\endpspicture
```
curve を自動で閉じると (最終点と開始点を結ばないと),
閉じと開始は直線で結ばれます.
blue 線の top は, とにかく赤の中に入っている高さで OK ですが,
描線が curve なので, 座標の取り方で curve が変化します.

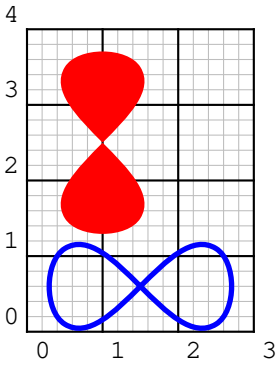
curve は最低 3 個の座標点 (x1,y1)(x2,y2)(x3,y3) を必要とします.
弧線の角度は結ぶべき座標点の位置に因り自動で描画されます.

絵画を描くときには, 方眼紙を表示し座標点を参照しながら描いて行く. 人の頭脳で座標点を思いながらとは違います. 目線と頭脳は対象物を, まず, 方眼紙で捉え, 方眼紙のどの地点に対象物を置くかで ..... これが決まれば後は出来 (描け) たも同じです. 描く際に PSTricks との格闘があるだけで, 格闘は力でねじ伏せるのではなく, 技で逃げきった方がすっきりします.



\psecurve の example
```
\pspicture(0,0)(4,4)%\psgrid
\psline*[linecolor=gray](0,0)(0,1)(4,1)(4,0)(0,0)
\psline*[linecolor=green](0,1)(1,2)(3,2)(4,1)(0,1)
\psecurve*[linecolor=red](0,1)(1,2)(2,3)(3,2)(4,1)
\endpspicture
```
\psecurve の開始点 (0,1) と最終点 (4,1) を結んで
この領分は塗りつぶしていません. 開始座標と終了座標 (領分) は外しています.
領分 (1,2)(2,3)(3,2) の範囲内を描画 (塗りつぶし) しています.

この意味は, 特性の違った描線を smooth に繋いで描いて行くことが出来る, つまり, 直前の特性の違った領分からな沿って自己領分へと描き入っていけます.



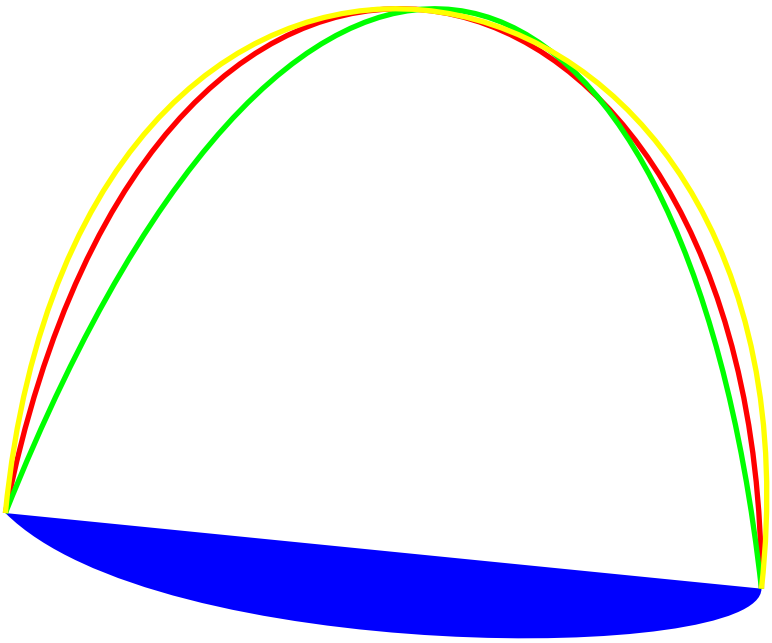
\psccurve の example

```
\pspicture(0,0)(3,4)\psgrid
\psccurve*[linecolor=red](.5,1.5)(1.5,1.5)(.5,3.5)(1.5,3.5)
\psccurve[linecolor=blue,linewidth=2pt]%
(.5,0.1)(.5,1.1)(2.5,0.1)(2.5,1.1)
\endpspicture
```

\psccurve は開始点と終了点を, curveline の属性を維持 (継続) したまま自動で閉じます.

- **\psbezier*[settings]{arrows}(X0,Y0)(X1,Y1)(X2,Y2)(X3,Y3)**

  ¶ 4 つの制御ポイントでベジェ(Bézier) 曲線を描く.



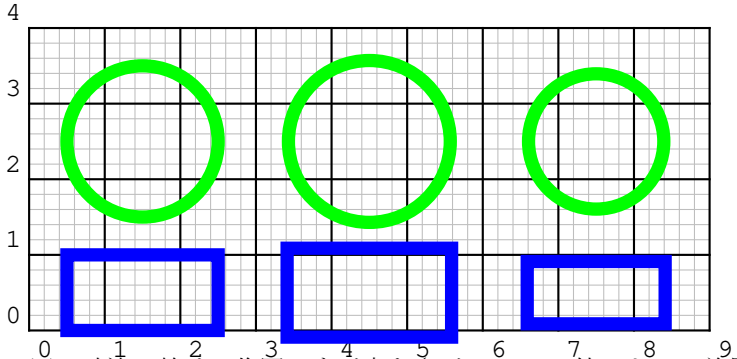
```
\pspicture(0,0)(14,10)%\psgrid
\psbezier[linecolor=red,linewidth=2pt](2,2)(4,12)(12,10)(12,1)
\psbezier[linestyle=none,fillstyle=solid,fillcolor=blue]%
(2,2)(4,0)(12,0)(12,1)
\psbezier[linecolor=green,linewidth=2pt](2,2)(6,12)(11,10)(12,1)
\psbezier[linecolor=yellow,linewidth=2pt](2,2)(3,12)(13,10)(12,1)
\endpspicture
```

\psbezier が円弧の描線は, 本当に美しい (それだけで芸術的) と思いませんか? この円弧は, 描線にしろ塗りつぶしにしろ他の line-commands では描画できない.
この curve の弧線は 4 個の座標 (X0,Y0)(X1,Y1)(X2,Y2)(X3,Y3) で与えなければなりません. 塗りつぶし (自動的に開始点と終点での閉じ) は, 直線で結合されています.

この美しさに見方を変えれば, 描かれる弧線は重力 (座標点) に因ってその直進性が曲げられた美しさ! 個人的な見方でしょうが. 円弧の定規では引けない内面性が感じ取れて仕方がないのです (私的には).

製図描画に関連して.

作図において, \psframe , \pscircle , \psellipes ,\pswedge に関しては,
描画する時, 三つの値 (dimen) を持っています. 一つはデフォルト値の middle,
あとの二つが inner と outer です. 次にそれらを使った描画サンプルです.



同じ寸法に於ける作図ですが大きさが dimen の値によって差異があります.

```
\pscircle[linecolor=green,linewidth=5pt,dimen=middle](1.5,2.5){1}
\psframe[linecolor=blue,linewidth=5pt,dimen=middle](.5,0)(2.5,1)
\pscircle[linecolor=green,linewidth=5pt,dimen=inner](4.5,2.5){1}
\psframe[linecolor=blue,linewidth=5pt,dimen=inner](3.5,0)(5.5,1)
\pscircle[linecolor=green,linewidth=5pt,dimen=outer](7.5,2.5){1}
\psframe[linecolor=blue,linewidth=5pt,dimen=outer](6.5,0)(8.5,1)
```

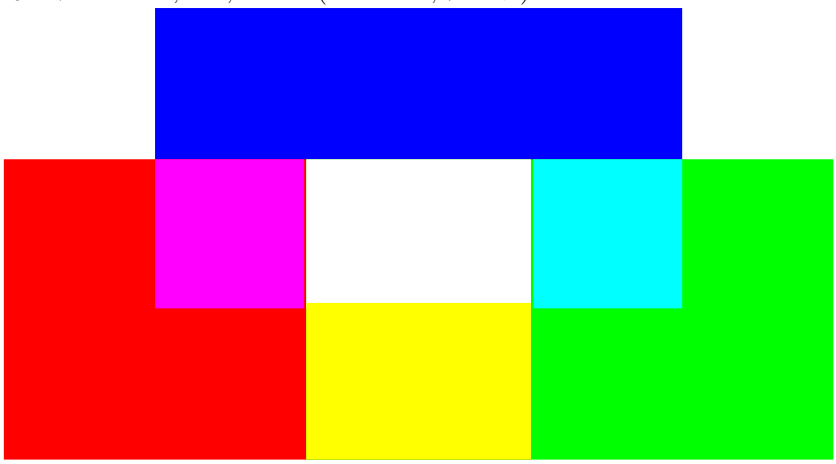
6.7.8) Shapes and color gradients(page 448)
Shapes and gradient are available for characters(text) or any other graphical objects.

About Light and color

光と闇は古代より物質世界と人間世界の両面を表してきました. 要するに, 光あるところに闇が存在し, 闇ある所に一条の光が射し込む. 人の感性を文学的に今日でも人間界にある. 20 世紀の量子力学によって, 光は粒子でもあり波でもあるということで文学的な意味に決着をつけました.

(一節の) 光は色に分析され, three primary colors(光の三原色) に纏められ, この赤 (red)・ 緑 (green)・青 (blue) の (光を) 任意の強さで混ぜ合わせることによって, 様々色 (光) を作り出すことができます. つまり, 光の三原色と様々な色名の誕生です.

光の三原色表示–Blue,Red,Green–(PSTricks; 製図描)

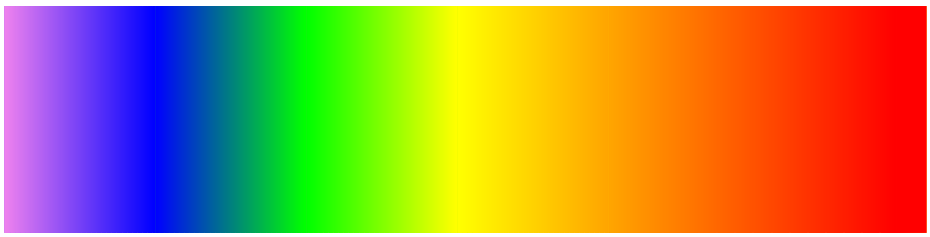


Blue+Red+Green=White, Blue+Red=Magenta, Blue+Green=Cyan Red+Green=Yellow

スペクトル

color − *name*	*color*	波長 $[nm]$	振動数 $[Hz]$
紫 (*violet*/*purple*)		$380 - 430$	$7.0 - 7.9 \times 10^{14}$
青 (*blue*)		$430 - 490$	$6.1 - 7.0 \times 10^{14}$
緑 (*green*)		$490 - 550$	$5.5 - 6.1 \times 10^{14}$
黄 (*yellow*)		$550 - 590$	$5.1 - 5.5 \times 10^{14}$
橙 (*orange*)		$590 - 640$	$4.7 - 5.1 \times 10^{14}$
赤 (*red*)		$640 - 770$	$3.9 - 4.7 \times 10^{14}$

プリズムなどで分解される光は, 紫から赤まで色が順番に並んでいます. 色に分解された光をスペクトル (spectrum) と呼んでいます.



Spectrum(PSTricks; 製図描)

PSTricks のグラデーションでの塗りつぶし描線数は `[gradlines=300]` がデフォルトです, 線数を多く設定すればグラデーションも微細になりますが, 結果の印刷速度も即して非常に遅くなります.
PSTrick の `[setting]` での color の値は rgb モデルです.

また, "グラデーション"を行うには, TEX に於いては, プリアンブルで `\input pst-grad.tex` の宣言が必要になります, 通常は使用直前での追加パッケージ呼び込むではなく, ファイルの Preamble(プリアンブル) 部分で呼び込みを宣言したり, `\def` に於けるマクロ記述を書いておきます.

TEX で表現した, 代表的な物質色名 68 色のリスト [cmyk モデル]

Pre-defined in TEX

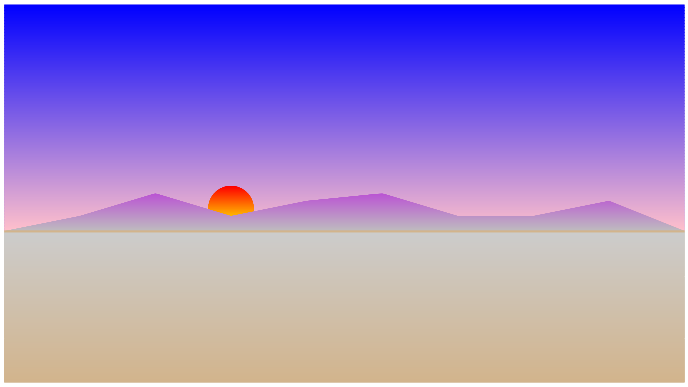
GreenYellow 0.15, 0.00, 0.69, 0.00	**Yellow** 0.00, 0.00, 1.00, 0.00	Goldenrod 0.00, 0.10, 0.84, 0.00
Dandelion 0.00, 0.29, 0.84, 0.00	Apricot 0.00, 0.32, 0.52, 0.00	Peach 0.00, 0.50, 0.70, 0.00
Melon 0.00, 0.46, 0.50, 0.00	YellowOrange 0.00, 0.42, 1.00, 0.00	Orange 0.00, 0.61, 0.87, 0.00
BurntOrange 0.00, 0.51, 1.00, 0.00	Bittersweet 0.00 0.75, 1.00, 0.24	RedOrange 0.00, 0.77, 0.87, 0.00
Mahogany 0.00, 0.85, 0.87, 0.35	Maroon 0.00, 0.87, 0.68, 0.32	BrickRed 0.00, 0.89, 0.94, 0.28
Red 0.00, 1.00, 1.00, 0.00	OrangeRed 0.00, 1.00, 0.50, 0.00	RubineRed 0.00, 1.00, 0.13, 0.00
WildStrawberry 0.00, 0.96, 0.39, 0.00	Salmon 0.00, 0.53, 0.38, 0.00	CarnationPink 0.00, 0.63, 0.00, 0.00
Magenta 0.00, 1.00, 0.00, 0.00	violetRed 0.00, 0.81, 0.00, 0.00	Rhodamine 0.00, 0.82, 0.00, 0.00
Mulberry 0.34, 0.90, 0.00, 0.02	RedViolet 0.07, 0.90, 0.00, 0.34	Fuchsia 0.47, 0.91, 0.00, 0.08
Lavender 0.00, 0.48, 0.00, 0.00	Thistle 0.12, 0.59, 0.00, 0.00	Orchid 0.32, 0.64, 0.00, 0.00
DarkOrchid 0.40, 0.80, 0.20, 0.00	Purple 0.45, 0.86, 0.00, 0.00	Plum 0.50, 1.00, 0.00, 0.00
Violet 0.79, 0.88, 0.00, 0.00	RoyalPurple 0.75, 0.90, 0.00, 0.00	BlueViolet 0.86, 0.91, 0.00, 0.04
Periwinkle 0.57, 0.55, 0.00, 0.00	CadetBlue 0.62, 0.57, 0.23, 0.00	CornflowerBlue 0.65, 0.13, 0.00, 0.00
MidnightBlue 0.98, 0.13, 0.00, 0.43	NavyBlue 0.94, 0.54, 0.00, 0.00	RoyalBlue 1.00, 0.50, 0.00, 0.00
Blue 1.00, 1.00, 0.00, 0.00	Cerulean 0.94, 0.11, 0.00, 0.00	**Cyan** 1.00, 0.00, 0.00, 0.00
ProcessBlue 0.96, 0.00, 0.00, 0.00	SkyBlue 0.62, 0.00, 0.12, 0.00	Turquoise 0.85, 0.00, 0.20, 0.00
TealBlue 0.86, 0.00, 0.34, 0.02	Aquamarine 0.82, 0.00, 0.30, 0.00	BlueGreen 0.85, 0.00, 0.33, 0.00
Emerald 1.00, 0.00, 0.50, 0.00	JungleGreen 0.99, 0.00, 0.52, 0.00	SeaGreen 0.69, 0.00, 0.50, 0.00
Green 1.00, 0.00, 1.00, 0.00	ForestGreen 0.91, 0.00, 0.88, 0.12	PineGreen 0.92, 0.00, 0.59, 0.25
LimeGreen 0.50, 0.00, 1.00, 0.00	YellowGreen 0.44, 0.00, 0.74, 0.00	SpringGreen 0.26, 0.00, 0.76, 0.00
OliveGreen 0.64, 0.00, 0.95, 0.40	RawSienna 0.00, 0.72, 1.00, 0.45	Sepia 0.00, 0.83, 1.00, 0.70
Brown 0.00, 0.81, 1.00, 0.60	Tan 0.14, 0.42, 0.56, 0.00	Gray 0.00, 0.00, 0.00, 0.50
Black 0.00, 0.00, 0.00, 1.00	White 0.00, 0.00, 0.00, 0.00	

任意色の設定 (モデル cmyk = Cyin, Magenta, Yellow, Black), 数値は {0~ 1} の間.
\definecolor{ 新色名 }{cmyk}{c 数値,m 数値,y 数値,k 数値 }

TEX で (文章に) 使用するには, プリアンブルで `\input colordvi.tex` の呼び込む設定をする事.

color 表現には様々な表現モデルが生み出されています, TV 界ではまた違った基準でのモデル採用です. PSTricks は基本的には rgb モデル表現で, パソコンの画面 (ディスプレイ) は rgb モデル表示です. 印刷業界はこの cmyk モデル採用です.

PSTricks でのグラデーション

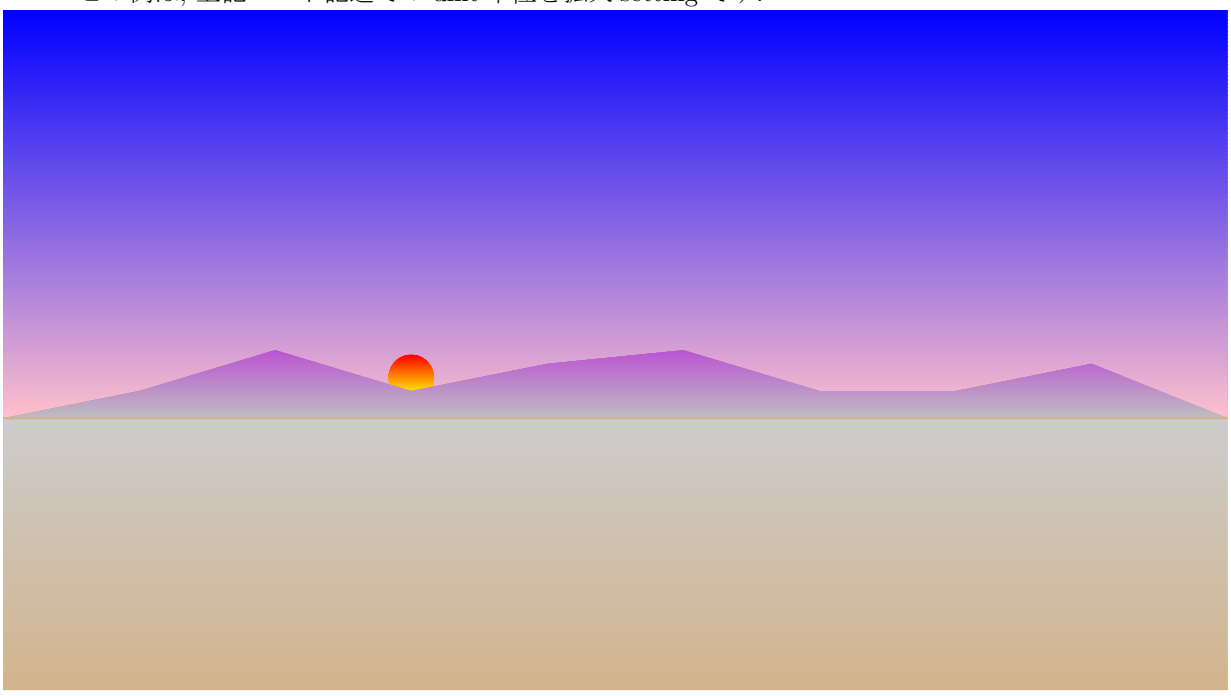


デフォルト [gradlines=300] で描写.

```
\pspicture(0,0)(9,5)
%%% The Sky
\psframe[linestyle=none,linewidth=0pt,fillstyle=gradient,%
gradbegin=blue,gradend=pink,gradmidpoint=1](0,2)(9,5)
%%% The Sun
\pscircle[linestyle=none,linewidth=0pt,fillstyle=gradient,%
gradangle=0,gradbegin=red,gradend=yellow](3,2.3){0.3}
%%% A Mountain ridge
\pspolygon[linestyle=none,linewidth=0pt,fillstyle=gradient,%
gradbegin=mediumorchid,gradend=gray,gradmidpoint=1]%
(0,2)(1,2.2)(2,2.5)(3,2.2)(4,2.4)(5,2.5)(6,2.2)(7,2.2)(8,2.4)(9,2)
%%% The Horizon
\psframe[linestyle=none,linewidth=0pt,fillstyle=gradient,%
gradbegin=grey80,gradend=tan,gradmidpoint=1](0,0)(9,2)
\psline[linecolor=tan](0,2)(9,2)
\endpspicture
```

[settings] の key とその値は tex-pstricksdoc.pdf の方で参照してください. 例えば, [gradlines=600] と設定すると, 1cm 幅を 600 本の線描で埋めます. その代わり印刷速度は応じて非常に遅くなります.

\psset{xunit=1.8cm,yunit=1.8cm} を用いて, 上記絵の拡大 (縮小可) 例です.
この例は, 上記コード記述での unit 単位を拡大 setting です.



PSTricks で任意の color は
\newrgbcolor{ 新色名 }{r 数値 g 数値 b 数値 } で数値は 0 ～ 1 の間でお互いの区切りは半角空白です.

r は red, g は green, b は blue. これを 0 ～ 1 の間で表現するモデルを rgb と呼んでいます.
`\newrgbcolor{red}{1 0 0}` , `\newrgbcolor{green}{0 1 0}` , `\newrgbcolor{blue}{0 0 1}` の値で,
`\newrgbcolor{white}{1 1 1}` , `\newrgbcolor{black}{0 0 0}` です.

`\newrgbcolor{white}{1 1 1}` = 自然界の全ての light(color) を混ぜ合わせると透明になり, 一節の visual light をプリズムを通すと人に理解 (感知) できる色に分解できる.
`\newrgbcolor{black}{0 0 0}` = 光を全て遮断すると, それは”闇”にになる.

私達は, 絵画にしろ画像にしろ映像にしろ写真にしろ, この囲いの中のコンセプトで (無意識に) 言葉にして対象物からの印象を語ったり述べたりします. また, *light and dark* の壮絶な美しさは, `http://hubblesite.org/gallery/album/entire` を参照願います. これらは, 数十億年前に発した光と闇の光景です. 残念ながら, 私達はこれらの現象の結果の current の現在形を見ることは不可能であり, さらに, 数十億年後の後世の人々だけが観る事ができる. しかし, 悲観する必要は無い, 人間 (頭脳) は有限数を正確に計算できる生命体なのだから, これら自然系の現象を非自然科学系であるコンピューターがシュミレーション (再現) した, 数十億年前の光と闇の現象の今日の curret 結果の映像を再現することが可能になるからです.

color モデルである, rgb のモデルは他に RGB と言うモデルもあります. 両者は同色名でも設定数値が違います, `Red{255 0 0}` , `Blue{0 0 255}` , `Green{0 255 0}` , `Whit{255 255 255}` , `Black{0 0 0}` です. これらは, 身近なところでは editor の emacs や xcolor(?) の RGB の数値です (color 名の大文字小文字は別にして).
(RGB[0 ～ 255] を 255 で除した値 $\Longrightarrow$ rgb) 参照, The LaTeX Graphics Copanion Second Edition page716)

PSTricks の画布である TeX のキャンバス (画布) ですが, LaTeX2ε や (plain)TeX の設定環境を離れれば裸の TeX の持っている値の限界付近は TeXBook によると, 縦横約 5.7583m 付近とあります. 従って `A\hfill B` とすれば, ”A”から約 5.75 メートル離れた位置に”B”が打たれる, また, 最小単位は **1sp(scaled point;65536sp=1pt)** とあります (人が見ることの出来る光の大きさが 100sp). これらのキャンバスや絵筆, あるいは印刷装置, ディスプレイ装置等々は現在では, まだ開発も試作も研究もされていません. 駅のプラットホームの看板でも 1 枚でこれだけの細かさと大きさ (内容) を貼った絵柄はないです. いずれは可能になる時代を迎える日が来るのでしょうが, 今日ではまだまだ TeX 全貌の野望は高望みです.

PSTrics で絵画を描くには, そんなに多くのコマンドを把握する必要はなく, むしろ, `[settings]` の key とその値に精通して, 描く対象物に対処することでイメージを絵画に出来る. もちろん, 自己の自由度に多くの制約は付きますが, これは画家の世界でも制約と言う意味では理解できる事だと私の割り切りです.

終了.

参照した”The LaTeX Graphics Companion Second Edition”に載っている examples のソースコードの全てが, `CTAN/info/examples/lgc2` にリソースされています. PSTricks 関連は,5-1-1.ltx ～ 6-7-56.ltx までと, color 関連が 11-2-1.ltx ～ 11-4-37.ltx です. すべて LaTeX2ε で run します. ftp , http の download は深夜か日曜日等々の時間帯で行う心積もりでお願いします. 特に大学系の CTAN サイトはなおさらご配慮お願いします.