

KETpic v5.1.1 コマンド一覧

for Scilab

PD プロットデータ

1 Scilab についての注意

1. KETpic の読み込みには次の 2 行を実行する.

```
Ketlib=lib( 'C:/work/ketpicsciL5/')    (C:/work/は作業フォルダ名)
Ketinit();
```

※ ThisVersion() によって、バージョン情報が得られる.

2. 行末に ; (セミコロン) をつけると結果非表示, そうでないと表示

3. ユーザー関数・変数は大文字から始めるのが慣例

4. 文字列は ' (シングル) または " (ダブル) で囲む

5. 関数などを引数とするときは, 文字列とする.

例) G=Plotdata('x^2', 'x=[0,1]');

6. \ (バックスラッシュ) は 1 つでよい.

7. Editor 画面でコマンド列を作って実行した方がよい.

8. 注釈は //

9. 行が変わっても継続することを表すには ...

10. 1 つの ' (シングル) を文字列の中におくことはできない.

※ ` (バッククオート) で代用して, Assign を使えば, ' に変換してくれる.

(Assign を参照)

※ Prime() で ' の文字列が得られる.

11. 数と文字列の変換

string(数)

msscanf(書式, 数)

evstr(文字列)

12. 異なる型のデータからリストを作るには list を用いる. (以下, Scilist という)

作成 L=list(... , ...);

取り出し A=L(i)

書き換え L(i)=...

消去 L(i)=null()

追加 L(\$+1)=...

長さ length(L)

結合 lstcat(L1,L2,...) (Mixjoin を参照)

空リスト list()

タイプは 15

2 Scilabのための追加コマンド

Member(D, L)	D が L の要素であれば true, そうでなければ false を返す. ※ L はベクトルまたは list
Flattenlist(LL)	LL を平準化した単層の list を作る
Mixdisp(list)	list の要素を画面に簡易表示
Mixjoin(L1, L2, ...)	要素を合併した list を作る 例) L1=list(G1,G2,3); L2=list(G3, 'a'); L=Mixjoin(L1,L2, [3,6]); => L は G1, G2, 3, G3, 'a', [3, 6] の list ※ [], list() は無視される
Mixsub(V,L)	添字が V である L の要素からなる sublist を作る 例) L=Mixsub(1:2,L);
Tolist(V)	V の要素からなる list を作る 例) L=Tolist(1:10);
Roundstr(X, Ni, Nf)	数 X を整数 Ni 位, 小数 Nf に丸めた文字列を返す 例) L=Roundstr(exp(1),1,5);
Op(N, Data)	Data の N 番目の要素 (Data は文字列, ベクトル, list)
Assign(式, 変数名, 値, ...)	変数名 (文字列) に値を割り当てた文字列を返す ※ 値は, 数, 文字列, 数行列, Scilablist 例) A=0.4; B=[2,1]; Fn=Assign('A*x^2+B(1)*y^2','A', A, 'B', B);
Assignset(変数名, 値, ...)	割り当て変数テーブルをセットする 例) Assignset('A', 0.4, 'B', [2,1], 'C', list(...), 'D', 'string');
Assignset('?'+変数名)	変数名の値を返す 例) Assignset('?A');
Assignset()	割り当て変数テーブルを初期化
Assignadd(変数名, 値, ...)	割り当て変数テーブルに追加する 例) Assignadd('C', 0.4, 'D', [2,1]);
Assignrep(変数名, 値, ...)	割り当て変数テーブルを置き換える 例) Assignrep('C', 0.8);
Prime(文字列)	文字列の最後に ' をつける 例) Prime('A'); 例) Prime(); (' だけを出力) ※ Assign('A'); としてもよい.
Trunc(X)	X の整数部分
Xmin(), Xmax(), Ymin(), Ymax()	ウィンドウ範囲 (デフォルト $-5 \leq x \leq 5, -5 \leq y \leq 5$)
Ptne(), Ptnw(), Ptsw(), Ptse()	フレーム枠の各頂点
Ptne(PD), Ptnw(PD), Ptsw(PD), Ptse(PD)	PD を囲む矩形の各頂点

<code>ThisVersion()</code>	Ketpic のバージョン
<code>Fracform(x{, tol})</code>	x に近い分数 (文字列) を返す ※ tol は誤差レベル (rat を参照) 例) <code>Fracform([2, 3; 5, 2]^(-1))</code>
<code>Dotprod(v1,v2)</code>	内積
<code>Crossprod(v1,v2)</code>	外積
<code>Derivative(関数文字列, 変数名, 値ベクトル { , 変数の番号 })</code>	関数の微分係数を求める. 例) <code>Derivative('x^2+y', '[x,y]', [2, 3], 1);</code> 例) <code>Assign('A', 3);</code> <code>Derivative(Assign('x^2+A*y'), '[x,y]', [2, 3], 2);</code> ※ 変数の番号を指定しないときは, 列ベクトルで返す.
<code>Integrate(関数文字列, 変数文字列, 積分区間 (数リスト) { , es, er})</code>	関数の定積分を求める. ($es=10^{-10}$, $er=10^{-8}$ は精度) ※ 区間 (リスト) は積分を分けて計算するときに指定 例) <code>deff('Y=Fn(x)', 'if x>0,Y=1,else,Y=-1,end');</code> <code>Integrate('Fn(x)', 'x', [-2,0,3]);</code>

3 設定コマンド

3.1 基本

<code>Setwindow([xmin, xmax], [ymin, ymax])</code>	ウィンドウ範囲を設定 例) <code>Setwindow([-%pi, %pi], [-1.5, 1.5]);</code> ※ <code>Xmin()</code> , <code>Xmax()</code> , <code>Ymin()</code> , <code>Ymax()</code> で値を得られる.
<code>Setwindow(PD{ またはリスト })</code>	PD の最大最小に合わせた window を自動設定
<code>Setscaling(ratio)</code>	縦の横に対する比を <i>ratio</i> に設定 (デフォルトは 1) 例) <code>Setscaling(2);</code> ※ ウィンドウも連動
<code>Setorigin(点)</code>	座標軸の原点を指定 (デフォルトは (0, 0)) ※ 空引数のとき, 現在の設定値を表示
<code>Setunitlen('単位長')</code>	単位長を指定する ※ <code>Beginpicture("")</code> とすると指定された単位長が使われる. ※ 空引数のとき, 現在の設定値を表示
<code>Ketinit()</code>	定数をデフォルトに初期化

3.2 空間 (平行投影)

<code>Setangle(θ, φ)</code>	角度 ($^{\circ}$) を指定 ※ デフォルト値は $\theta = 60$, $\varphi = 30$ ※ 空引数のとき, 現在の設定値を表示
<code>Initangle()</code>	デフォルト値に戻す

3.3 空間（一点投影）

Setpers(注視点, 視点) 一点投影の FocusPoint, EyePoint を指定
※ デフォルト値は FocusPoint=[0,0,0], EyePoint=[5,5,5]
Setpers() 現在の FocusPoint, EyePoint を表示
※ 空引数のとき, 現在の設定値を表示
SetstereoL($R, \theta, \varphi, \Delta$) 原点を注視点として, 空間極座標により定まる左目の位置を
視点にセット (Δ は目の間隔)
SetstereoR($R, \theta, \varphi, \Delta$) 原点を注視点として, 空間極座標により定まる右目の位置を
視点にセット (Δ は目の間隔)

4 プロットデータの作成

4.1 平面図形

Plotdata(関数, 範囲, オプション)
関数のグラフの PD を作成
例) G1=Plotdata('sin(x)', 'x=[-2*%pi, 2*%pi]');
※ 範囲を 'x' とすると, Xmin() から Xmax() にとる.
※ x 以外の変数を使うときは関数に使われていないかを注意.
※ オプション
 'N=...' 点の個数
 'E=[...]' 除外点のリスト
 'E=関数' 関数の 0 点は除外
 'D=...' 連続限界値 (これ以上離れたら結ばない)
※ デフォルトは N=50, D=%inf
※ 関数は function で与えてもよい.
例) G1=Plotdata('1/x', 'x', 'N=200', 'E=[0]', 'D=1');
例) G2=Plotdata('1/((x-1)*(x+2))', 'x', 'E=(x-1)*(x+2)');
例) deff('Out=Fn(x)', 'Out=sin(x)');
 G3=Plotdata(Fn, 'x');
Listplot(点のベクトルまたは列または list)
折れ線の PD を作成. ただし, 点は線分で結ぶ.
例) G2=Listplot([[3,2],[5,4]]);
Lineplot(点 A, 点 B{ , 長さ, 半直線 })
線分 AB を延長した線分の PD を作成
例) G3=Lineplot([3,2],[5,4]);
例) G4=Lineplot(A, B, '+');
 半直線 AB (B 側に延長)
※ 長さのデフォルトは片側 100
Paramplot(パラメトリック関数, 範囲, オプション)
パラメトリック関数のグラフの PD を作成
※ t 以外の変数を使うときは関数に使われていないかを注意
※ 関数は function で与えてもよい.
例) G3=Paramplot('[cos(t), sin(t)]', 't=[0, 2*%pi]');
例) G4=Paramplot(Fnx, Fny, 't=[0, 1]');

Rotatedata(PD, 角度 {, 中心 })

PD を回転した PD を作成

例) G4=Rotatedata(G1,%pi/4);

Translatedata(PD, x 方向 y 方向)

PD を平行移動した PD を作成

例) G5=Translatedata(G1, 3, -1);

Scaledata(PD, x 方向, y 方向 {, 中心 })

PD を拡大 (縮小) した PD を作成

例) G6=Scaledata(G1, 2, 1/3);

Reflectdata(PD, 点) 点対称移動した PD を作成

Reflectdata(PD, [点 1, 点 2])

線対称移動した PD を作成

例) G7=Reflectdata(G1, [0,0]);

例) G8=Reflectdata(G1, [[0,0], [0,1]]);

Pointdata(PD, ...) PD の節点の list を作成

例) G9=Pointdata(G1);

※ Drwpt(G9) などで、点のプロットができる。

Circledata(中心, 半径 {, オプション }) 円の PD を作成

Circledata(中心, 通る点 {, オプション }) 円の PD を作成

例) G10=Circledata([3,1], 2);

※ オプション

'R=...' θ の範囲

'N=...' 点の個数

例) G10a=Circledata([3,1], 2, 'R=[0, %pi/2]');

例) G10b=Circledata([3,1], 2, 'N=100');

例) G10c=Circledata([3,1], [2,4], 'N=100');

Framedata(P, dx{, dy}) 点 P を中心に $\pm dx, \pm dy$ の矩形 (dy を省略すると $dy=dx$)

例) G3=Framedata([3, 1], 0.5);

※ 引数を list でまとめて渡してもよい

Framedata([x_1, x_2], [y_1, y_2])

$x_1 \leq x \leq x_2, y_1 \leq y \leq y_2$ の矩形の PD を作成 (右上から反時計)

※ 引数が空のとき, Setwindow で指定した枠

例) G1=Framedata([-2, 3], [1, 4]);

例) G2=Framedata();

Ovaldata(P, dx, dy{, rc}) 点 P を中心に $\pm dx, \pm dy$ の四分円ボックス

※ rc は四分円の半径 (default は 0.2)

例) G3=Ovaldata([3, 1], 0.5);

Hatchdata(パターン文字 (list){, 開始点 }, (閉) 曲線の列 {,kaku{,haba} })

パターンと一致する領域を斜線塗りする PD を作成

例) G1=Hatchdata(list('io'), list(g1,g2), list(g3));

(i; 内部, o; 外部)

※ 開始点が指定されたとき

(仮想的に) その点を通る斜線から描き始める

※ kaku は斜線の傾き (def=45), haba は間隔 (def=1)

例) G2=Hatchdata(list('ii'), O, list(G1), -45, 1.5);

※ 曲線リスト内の PD は隣接の順に指定
 ※ 閉じていないとき

- (1) 方向 's','n','w','e' を指定する
- (2) 窓枠とちょうど 2 点で交わる場合、領域の点を指定
- (3) 指定しなければ端点を直線で結ぶ.

例) G3=Hatchdata(list('ii'), list(g1,'s'), list(g2, [3,0]));
 Hatchdata(領域の点 {, 開始点 }, (閉) 曲線 list の列 {,kaku{,haba} })
 点 (のどれか) が含まれる領域を斜線塗り

例) F4=Hatchdata(list(A,B,C), list(G1), list(G2,G3));
 ※ 包含パターンが点 A, B, C のどれかと一致する領域
 を斜線塗り (領域は隣接するものとする)

Enclosing(PD リスト {, 始点の近くの点 })
 PD 列の直近の交点を結んで閉曲線を作成

例) G2=Enclosing(list(G1, Invert(G2), G3), [2,1]);
 G1 と (最後の)G3 の交点のうち, [2,1] に近い点から始める
 ※ 交点が 1 個の場合は, 点を省略してよい.

Dotfilldata(パターン文字列 (リスト) {, 開始点 }, (閉) 曲線 PD リストの列 {, 濃さ })
 パターンと一致する領域を点描する PD を作成

例) Fd=Dotfilldata("ii",list(G1),list(G2),0.7);
 ※ 濃さ d は $0 < d \leq 1$ (デフォルトは 0.5)
 ※ 書き出しは, Drwpt を用いる.

Arrowdata 矢印の PD を作成 (Arrowline 参照)
 ※ やじりは塗りつぶさない

Arrowheaddata やじりだけの PD を作成 (Arrowhead 参照)

Bowdata(点 A, 点 B{, 曲がり {, 切り } })
 弓形の PD を作成

- ※ 曲がり: 弧の曲がり (デフォルトは 1)
- ※ 切り: 中央に入れる切りの長さ (デフォルトは 0)
- ※ 点 A から B に反時計まわりに弧をかく

例) Bowdata([2, 1], [3, 4], 0.8, 0.5);

Bowmiddle(弧データ }) 弓形の midpoint を返す

Splinedata(点データ {, オプション })
 spline 曲線の PD を作成

- ※ 点データは PD (複数可) または PD のリスト
- ※ オプション:

- 'N=点の個数 (のベクトル)' (デフォルトは 50)
- N が数のときは, 最大値を表す
- 'C' (閉曲線でスムーズにつなぐ)
- 'C=閉曲線の添字番号'

例) Fs=Splinedata(PL, 'N=200', 'C'); (PL は点データ)
 例) Fs=Splinedata(PLL, 'N=[50,20]', 'C=[0,1]');
 (PLL は複数の PD)

Skeleton2data(平面曲線 list1, 平面曲線 list2{, 大きさ {, 遠近の閾値 } })
 list2 で隠される list1 のスケルトンデータを作成

Anglemark(A, B, C {, サイズ })

$\angle ABC$ の間の角度記号を作成
 ※ BA から BC へ反時計回りに描く
 ※ サイズのデフォルトは 0.5
 Paramark(A, B, C {, サイズ})
 $\angle ABC$ の間の角度記号 (平行四辺形) を作成
 Sumfun({ 定数項, } 一般項, 添字のリスト, 変数の範囲, { 点の個数 })
 $c + \sum_{k=a}^b f(x, k)$ の PD を作成 (Scilab に追加)
 ※ 定数項が 0 のときは省略できる.
 ※ 点の個数は 'N=個数' で指定 (デフォルトは 50)
 例) G1=Sumfun(1, 'x^n/factorial(n)', 'n=1:5', 'x=[-2, 2]');
 Implicitplot(関数, x 範囲, y 範囲 {, 分割数})
 Implicitplot(Zvalue, Xvalue, Yvalue)
 陰関数の PD を作成
 ※ 分割数のデフォルトは [50, 50]
 例) G1=Implicitplot('x^2+y^2-1', 'x=[-1,1]', 'y=[-1,1]');
 Deqplot(方程式, x 範囲, x0, y0{, 分割数})
 微分方程式の解曲線の PD を作成
 ※ y' は y' と書く.
 ※ x の範囲を省略すると, 描画範囲全体
 例) G1=Deqplot('y'=y*(1-y)', 'x', 0, 0.2, 'N=100');
 例) G2=Deqplot('y'=-0.4*y'-3*y', 'x=[0,5]', 0, [0,2]);
 例) G3=Deqplot('[x,y]'=[x*(1-y), 0.3*y*(x-1)], ...
 't=[0,20]', 0, [1,0.5], 'N=200');

4.2 空間図形

Spaceline(空間点のベクトルまたは list)
 空間点を結ぶ線分の PD3d を作成
 例) G1=Spaceline([3, 2, 1], [5, 6, 6]);
 Spacecurve(関数, 範囲, オプション)
 空間曲線の PD3d を作成
 例) G2=Spacecurve('[cos(t), sin(t), t]', 't=[0, 2*%pi]');
 Rotate3data(PD3, v1, v2 {, 中心})
 PD3 を v1 が v2 に重なるように回転した PD3d を作成
 例) G2=Rotate3data(G1, [1, 0, 0], [1, 2, 3]);
 ※ PD3 は list でもよい (この場合は list を返す)
 Rotate3data[PD3, 回転軸, 角度 {, 中心}]
 PD3 を回転軸のまわりに回転した PD3d を作成
 例) G2=Rotate3data[G1, [0, 0, 1], %pi/4];
 ※ PD3 は list でもよい (この場合は list を返す)
 Translata3data(PD3, 移動ベクトル v)
 PD3 を v だけ移動した PD3d を作成
 例) G2=Translate3data[G1, [3, 2, 1]);
 例) G2=Translate3data[G1, 3, 2, 1];
 ※ PD3 は list でもよい (この場合は list を返す)

※ 面 datalist は list(VL, FL), または, その list
 ※ 面を点で直接指定するときには VL=list() とする.
 例) Fd=list(list(),list([3,2,1],[0,0,0],[1,2,4]));
 ※ PhHiddenData() で陰線の PD を取り出せる.
 Phsrawparadata(面 datalist), Phsrawpersdata(面 datalist)
 複数の多面体の PD3d を作成 (陰線処理をしない)
 Facesdata(面 datalist { , 追加曲線 PDlist }, 射影のタイプ)
 面の辺 (と追加曲線) を面により陰線処理
 ※ 射影のタイプは 'para', 'pers', 'rawpara', 'rawpers'
 Faceremovaldata(面 datalist, 曲線 PDlist, 射影のタイプ)
 曲線を面により陰線処理

5 データの書き出し

5.1 基本コマンド

Windisp(PD 列または list {, 画面指定 })
 画面を開き, PD 列を表示 (確認のため)
 ※ 画面指定
 'c': 現在の画面を消去して表示
 'a': 現在の画面に追加して表示
 'n': 新規画面を作成して表示 (デフォルト)
 '番号 a(または c)': 番号の画面に表示
 例) Windisp(G1, G2);
 例) Windisp(list(G1,G2));
 例) Windisp(G2, 'c');
 Openfile("ファイル名"{, "単位長", "SF=ソースファイル名"})
 書き出し用ファイルを開く (デフォルトは画面)
 例) cd('C:/TeXF/');
 Openfile('fig.tex');
 Openfile('fig.tex','fig.sce');
 ※ 単位長を指定すると Beginpicture("単位長") まで書き出す
 ※ ソースが同一フォルダにあるときは, SF の指定は不要
 Beginpicture('単位長') picture 環境を始める.
 例) Beginpicture('1cm');
 例) Beginpicture('2*10/12cm');
 Endpicture(1) picture 環境を終える (座標軸をかく)
 Endpicture(0) picture 環境を終える (座標軸をかかない)
 Closefile({"1"または"0"})書き出し用ファイルを閉じる (デフォルト=画面に戻す)
 ※ "1"または"0"の文字列を指定
 Endpicture(1 または 0) を書き出す

5.2 設定コマンド

Setax(線種, 横軸名, 位置, 縦軸名, 位置, 原点名, 位置)
 座標軸を設定 (引数 7 個)

例) `Setax('a', 't', 's', 'u', 'w', 'O', 'nw');`
 ※ 線種は `d`; `line`, `a`; `arrow`
 ※ `arrow` のとき `'a0.5'` のように、矢印の大きさを指定できる。
 ※ ” とすると、現在の設定を変更しない。
 ※ 以降が ” のとき省略できる。また途中からも指定できる
 例) `Setax('a');`
 例) `Setax(6, 'O', 'se');` (6 番目から指定)
 ※ 位置は `'n', 's', 'e', 'w', 'ne', 'nw', 'se', 'sw'`
 ※ `'s2w3'` のように微小移動量を付加してよい。
 ※ 空引数のとき、現在の設定値を表示
Setpen(倍率) 線の太さを指定 (標準からの倍率で)
 例) `Setpen(1.5);`
 ※ 空引数のとき、現在の設定値を表示
Setpt(倍率) `Drwpt` の点の大きさを指定 (標準からの倍率で)
 ※ 空引数のとき、現在の設定値を表示
Setmarklen(倍率) 目盛りの大きさを指定 (標準からの倍率で)
 ※ 目盛りの大きさは微小移動量の単位としても用いられる。
 ※ 空引数のとき、現在の設定値を表示
Setarrow(鋸の大きさ {, 開き角 {, 鋸位置 {, 太さ } } {, 形と位置 })
 矢印の形状を指定する
 例) `Setarrow(0.5, 1, 1, 0.7, 'tf');`
 ※ デフォルト 大きさ 1, 角度 18°, 位置は終点
 ※ 5 以下の開き角を指定したときは、18° からの倍率とする
 ※ 形状 'l' : ライン 'f' : 塗り (デフォルト)
 ※ 位置微調整 'c' : 中央 'b' : 下 't' : トップ (〃)
 ※ 空引数のとき、現在の設定値を表示

5.3 プロットデータ

Drwline(PD 列または list{, 太さ })
 PD 列またはを实線で書き出す
 例) `Drwline(G1, G2);`
 例) `Drwline(G3, 0.5);`
Dashline(PD 列または list{, len {, gap} })
 PD 列または list を破線で書き出す (実線部から始まる)
 例) `Dashline(G1, G2);`
 例) `Dashline(G1, 1.5);` (実線部, ギャップとも 1.5 倍)
 例) `Dashline(G1, G2, 1.5, 0.5);`
 (実線部 1.5 倍, ギャップ 0.5 倍)
Invdashline(PD 列または list{, len{, gap} })
 破線を書き出す (ギャップから始まる)
Dottedline(PD 列または list{, len {, size} })
 点線を書き出す
 例) `Dottedline(G1, G2);`
 例) `Dottedline(G1, 1.5);` (間隔 1.5 倍)
 例) `Dottedline(G1, G2, 1, 0.5);` (点の大きさ 0.5 倍)

Arrowline(A, B {, 鋸の大きさ {, 開き角 {, 鋸位置 {, 太さ {}} {,
 形と位置, "Cut=切り込み率"} })
 点 A から B に向けて矢印をかくコードを書き出す
 例) Arrowline(A, B);
 例) Arrowline(A, B, 2, 10, 'l');
 例) Arrowline(A, B, 1, 18, 0.5, 2, 'lc');
 ※ AB の中点の位置に鋸をかく

Arrowhead(位置, 方向 {, 大きさ {, 角度 {, 形状と位置, "Cut=切り込み率"} })
 鋸だけを書き出す
 例) Arrowhead([0, 0], [2, 1], 'cl');

Arrowhead(P, PD{, 大きさ {, 角度 {, 形状と位置 { })
 PD 上の点 P に矢印を描く
 例) Arrowhead([1, 1], Plotdata('x^2', 'x'));
 ※ 鋸はライン

Drwpt(点の列 {, 塗り { }) 点を書き出す (大きさは Setpt で指定, 塗りのデフォルトは 1)
 例) Drwpt([2, 3], [5, 7]);

Drwxy() 座標軸を書き出す

Htickmark(座標, 方向, 数式, ...) 横軸上に目盛りをつける (方向のデフォルトは's')

Vtickmark(座標, 方向, 数式, ...) 縦軸上に目盛りをつける (方向のデフォルトは'w')

例) Htickmark(-1, -1, 1, '1', %pi, '\pi');
 例) Vtickmark(-1, 'e', -1, 1, 'ne', '1');
 例) Htickmark([2, 1], 'a');
 ※ 数式を省略すると目盛りだけをつける

Htickmark('m..n..r..') 横軸全体に目盛りをつける

Vtickmark('m..n..r..') 縦軸全体に目盛りをつける
 ※ m (目盛りの間隔), n (文字を何目盛り毎に), r (数の倍率)
 例) Htickmark('mn') (目盛りと数を 1 間隔でつける)
 例) Vtickmark('m1n2r1.5') (1.5 倍した数を 1 つとびに)

Shade(PD 列または list{, 濃さ { }) 閉曲線の内部を塗りつぶし 濃さ; 0 ~ 1 (デフォルトは 1)

5.4 文字の書き入れ

Letter(点, 方向, 文字列, ...) 点の位置の「方向」に文字列をかく (複数可)
 例) Letter([4, 3], 'n', '文字');
 ※ 位置は'n', 's', 'e', 'w', 'ne', 'nw', 'se', 'sw', 'c'
 ※ 'n1' n の方向にさらに 1 目盛長だけ離す

Expr(点, 方向, 数式, ...) 点の位置の「方向」に数式をかく (\$\$は不要)
 例) Expr([4, 3], 's', 'y=f(x)');

Letterrot(点, 方向 {, 接線方向 {, 法線方向 { }, 文字列)

Exprrot(点, 方向 {, 接線方向 {, 法線方向 { }, 文字列)
 「点」の位置に「方向」を右横方向にできるように文字を傾けて書く

※ 「接線方向」, 「法線方向」 はそれぞれの微少移動量
 ※ graphicx パッケージが必要
 Fontsize(' 記号') 文字サイズの変更コマンドを書き出す
 'n', 's', 'f', 'ss', 't',
 'la', 'La', 'LA', 'h', 'H' (" のとき 'n')
 例) Fontsize('s');

Bowname(弓形, 数式 {, 方向 })
 弓形 PD の「方向」に式を書き入れる
 ※ 方向のデフォルトは "c"
 例) Gb=Bowdata(A, B, 1, 0.5);
 Bowname(Gb, 'd');

Bownamerot(弓形 {, 接線方向 {, 法線方向 }}, 数式 {, 向き })
 弓形 PD の中央に式を傾けて書き入れる
 ※ graphicx パッケージが必要
 ※ 向きに -1 を指定すると向きが反対になる

Xyzaxparaname(軸データ {, 各軸のラベル名 } {, 離れ })
 平行投影で, 各軸のラベルを書き入れる
 例) Gax=Xyzax3data('x=[0,1]', 'y=[0,1]', 'z=[0,2]');
 Xyzaxparaname(Gax);
 ※ '¥sin x' など文字列で指定することもできる

Xyzaxpersname(軸データ {, 各軸のラベル名 } {, 離れ })
 一点投影で, 各軸のラベルを書き入れる
 例) Xyzaxpersname(Gax, ", ", 'w');

Execmd(文字列の list) コマンド文字列を順次実行
 例) StrL=list(Assign('Beginpicture('1cm)'), 'Endpicture(0)');
 Execmd(StrL);

6 プロットデータの操作

6.1 平面

Joingraphics(PD1, PD2, ... {, 'L' })
 複数の PD を 1 つの PD に合併
 G11=Joingraphics(F9, G10);
 ※ 'L' を指定したときは, 結果をリストで返す

Dividegraphics(PD)
 PD を要素に分けた list を作成
 例) FL=Dividegraphics(G1);
 例) G1=Op(1, FL);

Joincrvs(PD 列)
 複数の曲線をつなげた PD を作成 (2D, 3D 共通)
 例) G3=Joincrvs(G1, Invert(G2));
 ※ 曲線は隣接する順番で指定する

Invert(PD)
 PD の点列を逆順にした PD を作成 (2D, 3D 共通)

Partcrv(s1, s2, PD)
 曲線 PD 上のパラメータ値 s1, s2 を両端とする PD を作成
 ※ s1 > s2 の場合
 s2 から終点, 始点から s1 までの PD のリストを出力
 PD が閉曲線のときは上の 2 つの PD をつなげる.

Partcrv(A, B, PD) 曲線 PD 上の点 A, B の間の部分曲線の PD を作成
 ※ A, B の順序が逆転しているとき, B から終点, 始点から A までの PD のリスト (閉のとき接続) を出力
 例) G1=Plotdata('x^2', 'x=[Xmin(), Xmax()]');
 G2=Partcrv([0,0], [1,1], G1);
 G3=Partcrv([1,1], [0,0], G1);

Intersectcrvs(PD1, PD2 {, 限界値 })
 2 曲線 PD1, PD2 の交点のリストを作成
 例) G1=Paramplot('cos(t), sin(t)', 't=[0, 2*pi]');
 G2=Plotdata('x+1/2', 'x');
 PL=Intersectcrvs(G1,G2);
 ※ 限界値のデフォルト 10⁻⁴

IntersectcrvsPp(PD1, PD2 {, 限界値 })
 2 曲線 PD1, PD2 の交点とパラメータのリストを作成

Intersectlines(L1, L2) 2 直線の交点を返す
 例) L1=Lineplot(A, B)
 L2=Lineplot(C, D);
 P=Intersectcrvs(L1,L2);

Nearestpt(P, PD) 点 P に最も近い曲線 PD 上の点とパラメータ値のリストを返す
 例) Pp=Nearestpt([0, 1], G1);
 A=Op(1,Pp);

Nearestpt(PD1, PD2) PD1 の節点のうち, PD2 に最も近い点データのリストを返す
 例) Pp=Nearestpt(G1, G2);
 A=Op(1,Pp);

Ptstart(PD) 曲線 PD の始点を返す

Ptend(PD) 曲線 PD の終点を返す

Numptcrv(PD) 曲線 PD の節点データの個数を返す

Ptcrv(n, PD) 曲線 PD の n 番目の節点を返す

Pointoncrv(s, PD) PD 上の点でパラメータ値 s をもつ点を返す
 例) Pointoncrv(5.3, G1);
 (5 番目の線分上で 0.3 の位置にある点)

Paramoncrv(P {, n }, PD) PD (の n 番目の線分) 上にある点 P のパラメータを返す
 例) Paramoncrv([3, 2], G1);
 例) Paramoncrv([2, 4], 5, G1);

Droppoint(PD {, 限界値 }) 限界値以下の隣接点を落として点を少なくする.
 ※ 限界値のデフォルト値は 0.02

Powersum(係数リスト (list) , 変数値 x {, 中心 c})

$$\sum_{k=0}^n a_n(x-c)^k$$
 の値を返す. (Plotdata で使用)
 ※ 係数リストは list(添字リスト, 係数リスト) の形
 例) Tmp1=0:20;
 Tmp2=ones(length(Tmp1))./factorial(Tmp1);
 CL=list(Tmp1, Tmp2);
 G1=Plotdata('Powersum(CL, x)', 'x');

FouriercoeffL(関数 F, 周期 T, 項数 N)
 F の N 項までのフーリエ係数リスト (list) を求める.

※ list(定数項, 余弦係数 (行), 正弦係数 (行), 周期) の形
 ※ Fouriersum で用いる.

Fouriersum(フーリエ係数リスト, 変数値)

フーリエ級数の値を返す. (Plotdata で使用)

例) deff('Out=Fun(x)', 'Out=abs(x)');
 CL=FouriercoeffL(Fun, 2, 30);
 G2=Plotdata('Fouriersum(CL, x)', 'x', 'N=300');

6.2 空間

Partcrv3(S1, S2, PD) 曲線 PD 上のパラメータ値 S1,S2 を両端とする PD を作成

Rotate3pt(点, V1, V2{, C})

回転移動した点を返す (Rotate3data 参照)

Rotate3pt[点, 軸, 角度 {, C}]

回転移動した点を返す (Rotate3data 参照)

Parapt(点), Perspt(点)

空間の点を投影した点を返す

Zparapt(点), Zperspt(点)

投影した平面を X, Y としたときの Z 座標

Invparapt(P, PD3d), Invperspt(P, PD3d)

PD3d を投影した PD 上の点 P に対応する PD3d 上の点

※ Pd3d が線分のときは, 延長線上の点でもよい.

Invperspt(s, PD2d, PD3d), Invperspt(s, PD2d, PD3d)

PD2d 上のパラメータ値 s の点に対応する PD3d 上の点

Cancoordpara(投影座標) 平行投影で「投影座標」で表される点の標準座標

Cancoordpers(投影座標) 一点投影で「投影座標」で表される点の標準座標

Viewfrom(Vec, 曲線 3D {, 非表示オプション })

一時的に Vec 方向からみた射影データを返す

例) Out1=Viewfrom([0,0,1], G1); (表示してデータを作成)

例) Out1=Viewfrom([0,0,1], G1, 0); (データのみを作成)

7 曲面の描画

7.1 関数データリストの定義

(1) $z = f(x, y)$ ($a \leq x \leq b, c \leq y \leq d$) のとき

list(関数, x の範囲, y の範囲)

Fd=list('z=x^2+y', 'x=[-1,2]', 'y=[-2,1]');

※ 変数名 x, y が関数名にある文字と重ならないようにする.

(2) $z = f(x, y), x = x(u, v), y = y(u, v)$ ($a \leq u \leq b, c \leq v \leq d$) のとき

list(関数 z, 関数 x, 関数 y, u の範囲, v の範囲 {, 境界の指定 })

例) Fd=list('z=sin(2*sqrt(abs(x^2+y^2))', 'x=R*cos(T)',

'y=R*sin(T)', 'R=[0,4]', 'T=[0,2*pi]', 'e');

※ 'e' は $r = 1, 0 \leq t \leq 2\pi$ で定まる閉曲線を表す.

※ sqrt の中は abs をつける (計算誤差のため).

(3) $x = x(u, v)$, $y = y(u, v)$, $z = z(u, v)$ ($a \leq u \leq b$, $c \leq v \leq d$) のとき
`list('p', 関数 x 関数 y, 関数 z, U の範囲, V の範囲 {, 境界の指定 })`
 例) `Fd=list('p','x=sin(U)*cos(V)','y=sin(U)*sin(V)',
 'z=cos(U)','U=[0,%pi]','V=[0,2*%pi]', '')`;

7.2 曲面のワイヤフレーム図

`Sf3data(FD, {, 曲線の点の個数 (リスト){, 横方向の分割数, 縦方向の分割数 })`
 曲面データリスト FD の 3 次元ワイヤフレームデータを生成
 ※ デフォルトは 点の個数 = [50, 50], 分割数 = 25
 ※ 点の個数を数 n で指定したときは 2 方向とも n になる
 例) `Out=Sf3data(Fd);`
`Sfparadata(〃)` FD のワイヤフレームデータを平行投影した 2 次元データ
`Sfpersdata(〃)` FD のワイヤフレームデータを平行投影した 2 次元データ

7.3 輪郭線データの作成

`Sfbdparadata(関数データ {, 各方向の分割数, 限界値 1 ベクトル, 限界値 2})`
`Sfbdpersdata(関数データ {, 各方向の分割数, 限界値 1, 限界値 2})`
 各方向の分割数 曲線, 陰関数のグリッド数
 デフォルトは 50
 リストか単独の数で与える
 限界値 1 ベクトル 同一点と見なす限界値と投影の垂直方向のずれ限界倍率
 デフォルトは [0.05, 1]
 限界値 2 交点の余裕幅 (デフォルトは 0.2)
 例) `Out1=Sfbdparadata(Fd, [50,50]);`
`Windisp(Projpara(Out1))`
 ※ 曲面上の 3 次元データとして得られる.
 ※ `BorderHiddenData()` で陰線の 3 次元データが得られる.
 ※ `CuspData()` で輪郭線の 3 次元データが得られる.
 ※ `CuspPt()` で cusp 点の 2 次元データが得られる.
 ※ `BorderPt()` で輪郭線の交点の 2 次元データが得られる.

7.4 ワイヤデータの作成

`Wireparadata(輪郭のリスト, 関数データ, 横の線数, 縦の線数 {, 分割数, 限界値 1, 限界値 2})`
`Wirepersdata(輪郭, 関数データ {, 横の線数, 縦の線数 {, 分割数, 限界値 1, 限界値 2})`
 輪郭 `Sfbdpersdata(Sfbdparadata)` で作成したデータ
 線 個数か値のリストで与える
 限界値 1 デフォルトは 0.05
 限界値 2 デフォルトは 0.2
 例) `Out2=Wireparadata(Out1,Fd1,5,5);`
`Windisp(Projpara(Out1,Out2))`
 ※ `WireHiddenData()` で陰線の 3 次元データが得られる.
 ※ `WirePt()` で輪郭線との交点の 2 次元データが得られる.

7.5 曲面と曲線

7.5.1 パラメータ平面上の曲線

Crvonsfparadata(平面上の曲線 2D データ, 輪郭, 関数データ {, オプション})

Crvonsfpersdata(平面上の曲線 2D データ, 輪郭, 関数データ {, オプション})

例) `Fg=Parametricplot(' [T, 0]', 'T=[0,2*%pi]');`

`Out4=Crvonsfpersdata(Fg,Out1,Fd);`

※ CrvonsfHiddenData() で陰線の 3 次元データが得られる。

7.5.2 曲面上の曲線

Crv3onsfparadata(曲面上の曲線 3D データ, 輪郭, 関数データ {, オプション})

Crv3onsfpersdata(曲面上の曲線 3D データ, 輪郭, 関数データ {, オプション})

例) `Out5=Crv3onsfparadata(Fg,Out1,Fd);`

※ Crv3onsfHiddenData() で陰線の 3 次元データが得られる。

7.5.3 曲面外の曲線

Crvsfparadata(曲線 3D, 輪郭, 関数データ {, オプション})

Crvsfpersdata(曲線 3D, 輪郭, 関数データ {, オプション})

例) `G2=Xyzax3data('x=[-5,5]', 'y=[-5,5]', ...`

`'z=[-5,5]');`

`Out2=Crvsfparadata(G2,Out1,Fd);`

※ CrvssfHiddenData() で陰線の 3 次元データが得られる。

※ 曲線と曲面が交わらないときは、引数の最後に -1 をつける。

7.5.4 その他のコマンド

Intersectcrvsf(曲線 3D, 関数データ {, 分割数 {, 限界値}})

曲線と曲面との交点 (MD) を求める。

7.6 曲面の切断

Sfcutdata(曲面データ, 切断面データ {, 分割数})

曲面を切ったときの切り口の 3d データを作成

※ 切断面は, x , y , z などの方程式で与える。

(平面の場合は, Phcutdata のように与えてもよい)

例) `Fd=list('p', 'x=R*cos(T)', 'y=R*sin(T)', 'z=2*(1-R)',`

`'R=[0,1]', 'T=[0,2*%pi]', 'se')`

`Out=Sfcutdata(Fd, 'x^2+(z-1/2)^2=1/4');`

Sfcutoffparadata(関数データ, 切断面, 符号 {, 分割数})

Sfcutoffrawparadata(関数データ, 切断面, 符号 {, 分割数})

Sfcutoffpersdata(関数データ, 切断面, 符号 {, 分割数})

Sfcutoffrawpersdata(関数データ, 切断面, 符号 {, 分割数})

切断面で切った曲面をリッジライン法で返す


```
例) Fd=list('z=2*(1-sqrt(abs(x^2+y^2)))','x=R*cos(T)','y=R*sin(T)','R=[0,1]','T=[0,2*pi]','e');
      Out=Sfcutoffparadata(Fd,'z=1+2*x','-');
```

8 その他

8.1 データの読みこみなど

Readtextdata(ファイル名, { 開始位置 {, オプション } })
 ファイルからコンマ, スペース, タブ区切りのテキストを読み込み,
 データリストを返す
 ※ オプション:
 '%書式' (デフォルトは%lf)
 'N=点の個数' (デフォルトはすべて)
 'R=列の数'
 'D=PD の区切りの限界値' (デフォルトは -%inf)
 (最初の値が D 以下のとき %inf で置き換える)
 例) DL=Readtextdata('data.txt', [2, 1], 'N=1000', 'R=2')
 例) DL=Readtextdata('data.txt', 'D=-1')
 Writetextdata(ファイル名, 行列 {'D=行区切り (数) '})
 ファイルにコンマ区切りのデータを書き出す.

8.2 作表

Tabledata({ 大きさ, } 縦線相対幅, 横線相対高さ)
 表のデータ list を返す
 戻り値: PD, 縦線添字, 横線添字, 枠縦 PD, 枠横 PD, 外枠 PD
 大きさは次のベクトル
 横, 縦 (, 左 margin, 右 margin (, 上 margin, 下 margin))
 ※ 横 (縦) を -1 としたときは, 縦 (横) 線のデータから
 自動的に計算される (デフォルト)
 縦線相対位置は左の罫線からの幅 list (縦方向の開始横番号, 終了横番号)
 横線相対位置は上の罫線からの幅 list (横方向の開始縦番号, 終了縦番号)
 ※ 描画領域は自動的に設定される
 例) Tmp1=list(20, 30,[30,1,2], 40);
 Tmp2=list([10,2,4,5,6],5);
 Tb=Tabledata([150,20],Tmp1,Tmp2);
 Windisp(Tb(1))
 例) V=Mixsub(Tb(2),Out(1));
 Windisp(V)
 例) Tmp1=list(20, 30,[30,1,2], 40, 30);
 Tmp2=list([10,2,4],5,5);
 Tb=Tabledata(Tmp1,Tmp2);
 Dividetable(表データ) 枠, 縦罫線, 横罫線を成分とするリストを返す
 例) G=Dividetable(Tb); (G(1),G(2),G(3) が枠, 縦, 横)

Partframe(表データ, 開始位置, 終了位置)

枠の一部の PD

※ 位置はそれぞれ, [列番号, 行番号]

※ 開始位置から終了位置までの反時計回りの PD

例) G=Parframe(Tb, [4,1],[1,2])

Findcell(表データ, 列番号, 行番号)

セルの情報 list (中心, 横幅/2, 縦幅/2) を返す

※ 番号は左上の位置

例) Out=Findcell(Out,2,1);

※ 番号がベクトルのときは, その範囲のセル

例) Out=Findcell(Out,[2,4],1);

※ 番号がベクトルのときは, その範囲のセル

例) Out=Findcell(Out,[2,4],1);

Findcell(表データ, 左セル, { 右セル })

例) Out=Findcell(Out,'A2');

Diagcelldata(表データ, 列番号, 行番号)

セルの対角線 PD のリストを返す

Putcell(表データ, 列番号, 行番号, 位置, 文字データ)

セルに文字列を入れるコードを出力

※ 位置は 'c', 'r', 'l', 'u', 'd', 'b' (微小移動量を付加できる)

例) Putcell(Out,2,1,'c','221');

例) Putcell(Out,'B3','l','\$ab\$');

Putrow(表データ, 行番号, 文字位置, 文字データの列)

1 行に順に文字を書き入れる

例) Putrow(TbL, 2, 'r', 'a', 'b', 'c');

※ 複数列にわたるときは, 列数を list 内で指定

※ 1 つの列の文字位置を変えるには, list で指定

例) Putrow(TbL, 2, 'c', list('r','a'), list(2, 'b'), 'c');

(r の位置に a, 2 列とって b をおく)

Putrowexpr(表データ, 行番号, 文字位置, 文字データの列)

1 行に順に数式を書き入れる

PutcoL(表データ, 列番号 (名前), 文字位置, 文字データの列)

1 列に順に文字を書き入れる

例) PutcoL(TbL, 'C', 'c', 'a', 'b', 'c');

PutcoLexpr(表データ, 列番号 (名前), 文字位置, 文字データの列)

1 列に順に数式を書き入れる

Putrowstr(表データ, 行番号, 文字位置, 文字列)

1 行に文字列の文字を 1 つずつ書き入れる

例) Putrowstr(TbL, 1, 'c', 'xyz');

PutcoLstr(表データ, 行番号, 文字位置, 文字列)

1 列に文字列の文字を 1 つずつ書き入れる

8.3 T_EX のコマンド書き出し (メタコマンド)

Texcom(' コマンド') T_EX コマンドのコードを書き出す

例) Texcom('\begin{minipage}{3cm}')

※ 'newline' のとき, 空白行を挿入

Openphr(ユーザーコマンド名), Closephr()

\def のコマンド定義を始める

例) Openphr('\p');

Texcom('\begin{array}{cc}');

Texcom('5 & 3\\');

Texcom('8 & 7');

Texcom('\end{array}\$');

Closephr();

Openpar(ユーザーコマンド名, 幅 {, 位置 }), Closepar()

minipage 環境を含む \def コマンド定義を始める

※ 位置のデフォルトは 'c'

例) Openpar('\s','5cm');

Texcom('\input{rei}');

Closepar();

Letter([2, 3], 'se', '\s');

Texletter(点 (list 形式), 方向, 文字列, ...)

点の位置の「方向」に文字列をかく (複数可)

例) Texletter(list(4, '#1'), 'n', '文字');

※ 位置は 'n', 's', 'e', 'w', 'ne', 'nw', 'se', 'sw', 'c'

※ 点の位置はリスト形式で, T_EX の形式で渡すことができる.

Texnewctr(番号または番号のベクトル)

K_εTpic で使うカウンタ (ketpictra,...) を定義

Texctr(番号またはカウンタ名)

番号のカウンタ名またはカウンタ名を返す

Texthctr(番号) \the+カウンタ名の文字列を返す

Texvalctr(番号) \value{ カウンタ名 } の文字列を返す

Texsetctr(番号, 文字列) カウンタに値をセットする T_EX コマンド列を出力

例) Texsetctr(2, '1*2/3');

例) Texsetctr(2, '(-#1)+2');

Texletter(位置 (list), 方向, 文字列)

位置 list で表される点に文字列をかく T_EX コマンド列を出力

例) Texletter(list(10, '-'+Texvctr(2)), 'ne', '\content');

例) Texletter(list(0, '#1'), 'c', 'A');

Texnewcmd(コマンド名, 引数の個数, オプション値)

\newcommand を始める T_EX コマンドを出力

Texrenewcmd(コマンド名, 引数の個数, オプション値)

\renewcommand を始める T_EX コマンドを出力

Texend() T_EX のコマンド定義を終わる T_EX コマンドを出力

Texfor(カウンタ番号, 初期値, 終了値)

T_EX のループ構造を始める.

※ 初期値, 終了値は文字列で与える.

例) Texfor(1, '1', '#1');

Texendfor(カウンタ番号) T_EX のループ構造を終える.

例) Texendfor(1);

Texforinit() T_EX のループ構造を初期化

Texif(数値条件 { ,1 })	<p>T_EX の if 構造を始める. (ifnum または ifdim)</p> <p>※ 条件は文字列で与える.</p> <p>※ 1 を追加したときは ifdim</p> <p>例) Texif('Texctr(1)<#2');</p>
Texelse()	T _E X の else ブロック.
Texendif()	T _E X の if 構造を終える.

8.4 obj ファイル作成

Openobj(ファイル名)	初期化して書き出し用ファイルを開く.
Closeobj()	ファイルを閉じる.
Setobjscale(scale)	図のスケール (単位長) を設定 (デフォルト 1)
Setobjdecimals(書式)	点座標の書き出し書式 (デフォルト "%7.4f")
Objsurf(関数, 変数 1 範囲, 変数 2 範囲, 分割数 1, 分割数 2, 向き)	関数で表される曲面の 3D データを書き出す
Objsurf(関数, 変数 1 列, 変数 2 列, 向き)	<p>関数</p> <p>関数: 位置ベクトルを与える関数</p> <p>変数範囲, 分割数: [A, B] (文字列でも可), N</p> <p>または, 変数の値列</p> <p>向き: 表 ("+") または裏 ("-")</p> <p>戻り値は次のリスト</p> <p>変数 1 の値列, 変数 2 の値列,</p> <p>変数 1 下限の点リスト (w), 変数 1 上限の点リスト (e),</p> <p>変数 2 下限の点リスト (s), 変数 2 上限の点リスト (n)</p> <p>例) <code>function Out=Fun(U,V);Out=[U,V,U^2+V^2];endfunction</code></p> <p><code>Dt=Objsurf(Fun,[-1,1],[-1,1],20,30,"+");</code></p>
Objthicksurf(関数, <変数範囲・分割>, 法線ベクトル関数, 厚み 1, 厚み 2, 向きと縁)	関数で表される曲面の厚みのある 3D データを書き出す
	<p>関数</p> <p>関数, <変数範囲・分割> は objsurf と同じ</p> <p>法線ベクトル関数</p> <p>両方向の厚み</p> <p>向きと縁: "+", "-", "w", "e", "s", "n"</p> <p>例) "+n-s+" (全体は+, 縁 n は-, 縁 s は+)</p> <p>戻り値は次のリスト</p> <p>厚み 1 についての objsurf のデータ,</p> <p>厚み 2 についての objsurf のデータ,</p> <p>指定された縁についての objrecs のデータ</p>
Objrecs(点リスト 1, 初期法線ベクトル, 向き)	
Objrecs(点リスト 1, 点リスト 2, 向き)	<p>点リスト 1 と初期法線ベクトルから順次できる長方形列</p> <p>または, 点リスト 1 と点リスト 2 からの長方形列を書き出す</p> <p>戻り値は次のリスト</p>

点リスト 1, 点リスト 2
例) `Dt=Objcurve(PtL1,[0,0,1],"+");`

8.5 カラー設定

`SetColor(色 {, 濃さ })` 色を設定

※ color パッケージ必要

色は, 次の文字列または `[c,m,y,k]` のベクトル

`'greenyellow'`[0.15,0,0.69,0], `'yellow'`[0,0,1,0], `'goldenrod'`[0,0.1,0.84,0], `'dandelion'`[0,0.29,0.84,0]
`'apricot'`[0,0.32,0.52,0], `'peach'`[0,0.5,0.7,0], `'melon'`[0,0.46,0.5,0], `'yelloworange'`[0,0.42,1,0]
`'orange'`[0,0.61,0.87,0], `'burntorange'`[0,0.51,1,0], `'bittersweet'`[0,0.75,1,0.24],
`'redorange'`[0,0.77,0.87,0]
`'mahogany'`[0,0.85,0.87,0.35], `'maroon'`[0,0.87,0.68,0.32], `'brickred'`[0,0.89,0.94,0.28], `'red'`[0,1,1,0]
`'orangered'`[0,1,0.5,0], `'rubinered'`[0,1,0.13,0], `'wildstrawberry'`[0,0.96,0.39,0],
`'salmon'`[0,0.53,0.38,0]
`'carnationpink'`[0,0.63,0,0], `'magenta'`[0,1,0,0], `'violetred'`[0,0.81,0,0], `'rhodamine'`[0,0.82,0,0]
`'mulberry'`[0.34,0.9,0,0.02], `'redviolet'`[0.07,0.9,0,0.34], `'fuchsia'`[0.47,0.91,0,0.08],
`'lavender'`[0,0.48,0,0]
`'thistle'`[0.12,0.59,0,0], `'orchid'`[0.32,0.64,0,0], `'darkorchid'`[0.4,0.8,0.2,0], `'purple'`[0.45,0.86,0,0]
`'plum'`[0.5,1,0,0], `'violet'`[0.79,0.88,0,0], `'royalpurple'`[0.75,0.9,0,0], `'blueviolet'`[0.86,0.91,0,0.04]
`'periwinkle'`[0.57,0.55,0,0], `'cadetblue'`[0.62,0.57,0.23,0], `'cornflowerblue'`[0.65,0.13,0,0],
`'midnightblue'`[0.98,0.13,0,0.43]
`'navyblue'`[0.94,0.54,0,0], `'royalblue'`[1,0.5,0,0], `'blue'`[1,1,0,0], `'cerulean'`[0.94,0.11,0,0]
`'cyan'`[1,0,0,0], `'processblue'`[0.96,0,0,0], `'skyblue'`[0.62,0,0.12,0], `'turquoise'`[0.85,0,0.2,0]
`'tealblue'`[0.86,0,0.34,0.02], `'aquamarine'`[0.82,0,0.3,0], `'bluegreen'`[0.85,0,0.33,0],
`'emerald'`[1,0,0.5,0]
`'junglegreen'`[0.99,0,0.52,0], `'seagreen'`[0.69,0,0.5,0], `'green'`[1,0,1,0], `'forestgreen'`[0.91,0,0.88,0.12]
`'pinegreen'`[0.92,0,0.59,0.25], `'limegreen'`[0.5,0,1,0], `'yellowgreen'`[0.44,0,0.74,0],
`'springgreen'`[0.26,0,0.76,0]
`'olivegreen'`[0.64,0,0.95,0.4], `'rawsienna'`[0,0.72,1,0.45], `'sepia'`[0,0.83,1,0.7], `'brown'`[0,0.81,1,0.6]
`'tan'`[0.14,0.42,0.56,0], `'gray'`[0,0,0,0.5], `'black'`[0,0,0,1], `'white'`[0,0,0,0]