PGFPlots
Cheatsheet, page 1 of 2

1 Pgfplots

Syntax

```
\begin{tikzpicture}
 \begin{axis}[<ax_option>*]
  [\addplot[+|3][<pl_option>*]<plot>]+
  [\draw{...}]*
  [\node{...}]*
  [\legend[]]?
\end{tikzpicture}
```

There are a large number of <ax_option> *and* <pl_option>*s, each of which are of the following types: flags[F], styles[S], or keys[K]. All options can be set for an axis or plot as indicated, and many can be set for different scopes (optionally delimited with {} braces) using* \pgfplotsset{...}. *Assign* \empty *for non-drawn entity. Each option is described below, in the sections relevant to the graphical behavior controlled.*

Examples

Many more examples here!

```
\begin{tikzpicture}                  % tikz envmt
\begin{axis}                         % pgfplot axis
[legend cell align=left,             % <ax_optns>
legend pos=outer north east]         % ...
  \addplot coord's {(0,0) (1,1)};    % coords
  \addplot+[...] coord's ...;        % appnd opts
  \addplot { sin(x)};                % math expr
  \legend{a,fine,legend}             % legend
\end{axis}                           %
\end{tikzpicture}                    %
```

Plot-level Settings

[ultra]thick <color>

2 Addplot Command

Syntax

```
\addplot coordinates (<tuple>)*
\addplot fill between[...]
\addplot table {file.dat}
\addplot table <inline_data_row>*
\addplot {<math_expression>}
\addplot gnuplot {<gnuplot_code>}
\addplot graphics {<image_file>}
```

Coordinates

Tables

Map dataset columns to plot variables using x, y *keys. Further processing available with* x expr, y expr *keys. Inside such expressions, the following helper macros are available:* \thisrow \thisrowno \coordindex \lineno

Expressions

By default, will plot "marks" at 20 "samples" along the function described by <expression>, *connecting each with piecewise line segments. Use* smooth *for Bezier interconnections, instead.*

```
\addplot[smooth,samples=50,mark=none]
  {x^2};
```

Δ default to radians using trig format plots.

Gnuplot

```
\addplot gnuplot [id=exp,domain=0:10]
  {exp(x)};
\addplot3[contour gnuplot]{exp(0-x^2-y^2)};
```

Options exclusive to gnuplotting in pgfplots:

translate gnuplot
id
prefix
raw gnuplot
parametric
parametric/var 1d
parametric/var 2d

Fill Between

Do something like:

```
\addplot[gray] fill between
  [of=A and B,soft clip={domain=3:4},];
```

... where A and B are previously-constructed tikz or pgfplots "named paths."

Graphics Plotting

Position image with [x|y][min|max]; *process with* includegraphics, *which passes parameters to* \node[]{\includgraphics[...]{...}}. *Control default node parameters with* node/.style *key.*

```
\addplot graphics
  [xmin=0,xmax=1, ymin=0,ymax=1,
  includegraphics={trim=12 9 12 8,clip}]
  {external.file};
```

It is possible to overlay PGFPlots axes onto 3D graphics, but is more tedious, and requires mapping 3D points to 2D canvas points. See here.

Parametric Plots

Can change variable, *but must set* domain:

```
\addplot [variable=t,domain=0:2*pi]
  {(sin(t),2*cos(t))}
```

3 Axes & Legends

All of the below can be set in styles rather than set on each plot or axis. ∃ a number of pre-fab styles for axes, lines, ticks, legends, and colorbars, such as: every major tick, title style, every loglog axis, *etc. See here for a full list. Or, ideally, create your own style, appending it to the relevant elements as call for using:* \pgfplotsset{ <style_name>/.style=...}.

Axis Lines & Labels

The following options afford setting axis and plot titles, including their style and positioning.

title
extra description
axis lines
axis lines*
axis [x|y|z] line
axis [x|y|z] line*
every inner [x|y|z] line
every outer [x|y|z] line
axis line style
[x|y|z] axis line style
every boxed [x|y|z] axis
separate axis lines
axis [x|y|z] line shift
axis [x|y|z] discont'y
hide [x|y|z] axis
inner axis line style
outer axis line style
grid
domain
[x|y|z][min|max]
[x|y|z]label
[x|y|z]label shift
[x|y|z]label near ticks
[x|y|z]label absolute
axis line style
legend pos
enlargelimits

Tick Marks

[x|y|z]tick
[x|y|z]ticklabel
[x|y|z]ticklabels
"" from table
extra [x|y|z]tick label
[x|y|z]minorticks
[x|y|z]majorticks
[x|y|z]tickmin
[x|y|z]tickmax
[x|y|z]tick pos
[x|y|z]ticklabel pos
[x|y|z]tick align
[x|y|z]tick distance
[x|y|z]ticklabel shift
[x|y|z]ticklabel style
minor [x|y|z] tick[S]
minor [x|y|z] tick num
extra [x|y|z] ticks
[x|y|z]tickten
scaled [x|y|z] ticks
max space between ticks
min space between ticks
try min ticks
tickwidth
major tick length
minor tick length
subtickwidth
[x|y|z]tick placement tolerance
hide obscured [x|y|z] ticks
sloped like [x|y|z] axis
every x tick scale label

3D Axes

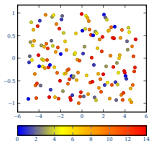
Radial Axes

Legends

Create imperatively using \addlegendentry *or* \legend, *or declaratively with option* legend entries. *Behind scenes, legends are implemented as Tikz matrices, so the styling for those apply here as well. Customize placement & appearance:*

at
anchor
legend pos
mesh legend
reverse legend
[x|y|z]bar legend
[x|y|z]bar inverval legend
legend plot pos
legend cell align
legend columns
transpose legend

Colorbars



These can be positioned, given different color scales, [dis]connected to plotting variables, moved outside an individual plot, etc. Often associated with point meta *data.*

colorbar right
colorbar top
colorbar horizontal
every colorbar
point meta [min|max]
colorbar to name
colorbar shift
colorbar style
colorbar/width
colorbar source
colorbar sampled
colorbar as legend

Axis Coordinate Systems

∃ two coordinate systems, unique to PGFPlots that allow node positioning along axis locations, namely, axis description cs *and* [x|y|z]ticklabel cs. *The former is simple, the latter affords better placement along 3D axes. Additionally, these coordinate systems enable some extra anchors for nodes:* near [x|y|z]ticklabel, near ticklabel opposite.

4 Positioning in Document

Scaling

To size and scale a plot, use either width *and* height, *or position along unit vectors as indicated with* x, y, *and* z. *Related general scaling, sizing, & aspect-ratio options include:*

width
height
x
y
z
[x|y|z] dir
scale only axis
scale mode
z mode
axis equal
axis equal image
unit vector ratio
unit vector ratio*
[x|y|z] post scale

Some pre-fab scaling styles for controlling various axis & font settings: normalsize, small, footnotesize, *and* tiny.

Additionally, low-level control is afforded through several scaling "strategies".

Alignment

Bounding Box

Layers

Grouping

5 Stylizing

Colors

Marks

line patterns

Number Formatting

Annotations & Labeling

symbolic y coords
nodes near coords

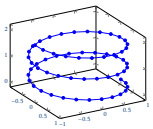
Tikz Interaction

Use disabledatascaling *opt'n to coerce tikz to pgfplots coord system. Then Tikz accordingly, eg:*

```
\draw[thick,<->] ($(\x,{sin(2 * \x)})$) -- ...
```

6 Plot Types

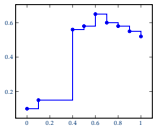
Line



Direct piecewise connections between samples are the default. Use sharp plot *to indicate explicitly, or change to* smooth *for Bezier interconnections. Create 3D line plots by passing 3d tuples into* \addplot3.

Constant

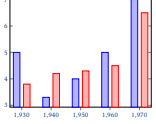
Graphs with horizontal line connections. Options control placement of segments relative to coordinates, and whether vertical segments also shown.



const plot mark left
const plot mark right
const plot mark mid

jump mark left
jump mark right
jump mark mid

Bar

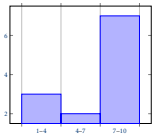


`[x|y]bar`, *indicated at axis level, affords bar charts in either x or y direction. Each* `\addplot` *contributes an additional series. Options apply at axis level as well (except those marked with *):*

xbar
ybar
bar shift auto
update limits
bar width
enlarge [x|y] limits
pattern
bar cycle list

Use `bar cycle list` *to install new styles for different series. A different graph altogether is* `[x|y]bar interval` *which, together with* `[x|y|z]ticklabel interval boundaries` *allows for bars of differing widths.*

Histogram

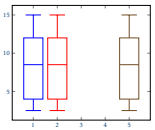


Hist's take `addplot table` *or* `<expression>` *commands. Data must be single column.*

```
\addplot+ [hist={bins=3}]
    table ...
\addplot+ [samples=200,hist] {rnd};
```

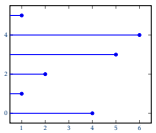
data
data min
data max
bins
symbolic coords
data coord trafo
intervals
cumulative
density
handler
data filter

Box



Can adjust orientation, box thickness, outlier distance, quartile thresholds, "whisker" styling, etc, with the following:

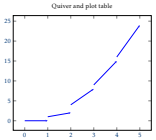
Comb



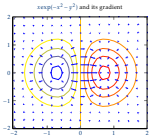
Similar to bar chart, but with lines rather than bars. As with bar, indicate with a set of coordinates:

```
\addplot+[xcomb] coordinates ...
```

Quiver



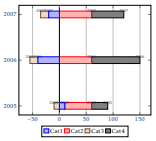
Requires a vector input (u,v) *that will calculate respective* (x,y) *components of each "quiver. Works readily with parametrics (using* `"variable=t`*) & tables of predefined* (x,y,u,v) *vectors.*



Often used to visualize tangent & gradient fields (using 3D quivers). Set `w` *for 3rd dimension.*

u
v
w
colored
scale arrows
update limits
every arrow
before arrow
after arrow
quiver legend

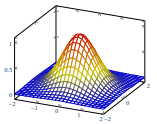
Stacked



Initiate basic stacking with `stack plots=y|x`*, or stacked bars with, eg,* `ybar stacked`*. Given coordinates for each* `\addplot` *line up positionally to comprise each of the plot's "series'." Can also stack in negative direction.*

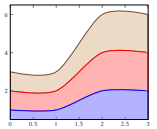
ybar stacked
xbar stacked
stack dir
stack negative
reverse stacked plots
stacked ignores zero
xbar interval stacked
ybar interval stacked
bar shift auto

Mesh



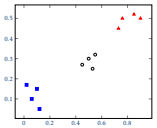
Available as 2D or, as is more customary, 3D. Invoke with `mesh`*. Default mapping from y to colormap. Change this using* `point meta`*.*

Area



For simple areas use stacking in addition to `area style`*. Can also use* `\addplot fill between` *in other plot types to achieve a fill affect, or use* `\closedcycle` *at end of* `\addplot` *to close a drawn path. Further restrict areas by setting* `soft clip` *key to a previously-defined "clip."*

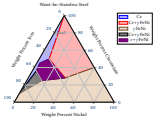
Scatter



For simple scatter plots, use `only marks` *For more control over marks, including styling (eg by color), and "classing" into groups, use* `scatter`*. "Classes" allow styling by ordinal metadata set using* `scatter src` *or* `point meta`*. Easily extended to 3D axes, which consume 3d vectors.*

scatter src
classes
use mapped color
scatter position

Ternary Diagram & Tieline

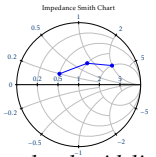


Requires "ternaryaxis" package. Plots to a "Barycentric" coordinate system, requiring that relative coordinates that are correctly specified.

"Tielines" are binodal curves in a ternary coordinate plane.

every ternary axis
tieline
ternary limits relative
tieline style
curve style

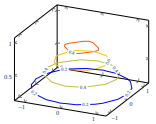
Smith



Requires external "smithchart" library. Can plot coords or lines onto this coordinate system, can invert the chart into a "admittance" chart, and can manipulate size and ticks. Low-level grid-line control with `x|ygrid each nth passes y` *and* `x|ygrid stop at x|y`*.*

smithchart mirrored
show origin
ytick label in circle
few smithchart ticks
every smithchart axis
ytick label ar'nd circle
ytick label ar'nd circle*
many smithchart ticks
dense smithchart ticks
default s.c. x|ytick

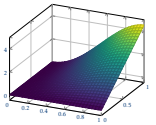
Contour



3D contours, like topo charts. Use `contour prepared` *or* `contour gnuplot` *for contour input, the former which takes matrix-style input, and the latter which creates these matrices and passes them to the former.*

number
levels
draw color
labels
contour external
contour gnuplot
contour filled
contour prep'd format
contour dir
contour label style
label distance
labels over line
label node code

Surface



Visualize 3D surfaces, which are–unlike with meshes–filled. Easiest to create with `<expression>` *but can also create with* `table` *or* `coordinates`*. Choosing from various "shaders" affords control over color gradients on each patch.* `Colormaps` *are inherited from the* `axis`*, but can be overridden. For lower-level control of color interpolations, set* `mesh/color input`*. This works with* `mesh`*,* `patch` *or* `surf`*.*

Patch

Regression