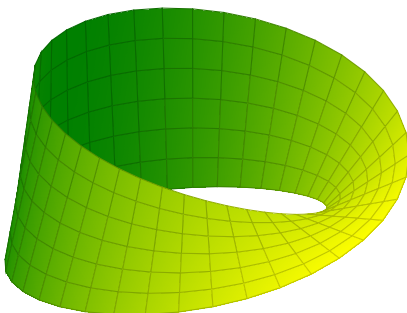


Quick Reference

PGFPLOTS

```
\begin{tikzpicture}
\begin{axis}[
  hide axis,
  view = {40}{40},
]
\addplot3[
  surf,
  colormap/greenyellow,
  shader = faceted interp,
  z buffer = sort,
  point meta = x,
  domain = 0:360,
  domain y = -0.5:0.5,
  samples = 40,
  samples y = 7,
]
({(1 + 0.5 * y * cos(x / 2))) * cos(x)},
{(1 + 0.5 * y * cos(x / 2))) * sin(x)},
{0.5 * y * sin(x/2)});
\end{axis}
\end{tikzpicture}
```



Ralph Schleicher

Contents

1	General	1	3.2	Line Plots	3
1.1	Document Structure . . .	1	3.3	Bar Plots	4
1.2	PGFPLOTS Options	1	3.4	Comb Plots	4
1.3	Key Handlers	1	3.5	Quiver Plots	5
1.4	Mathematical Expressions	1			
2	Axis Environments	1	4	Lines and Markers	5
			4.1	Line Width	5
3	Plots	2	4.2	Line Cap	5
3.1	Input Data	2	4.3	Line Join	5
3.1.1	Coordinates List	2	4.4	Dash Pattern	6
3.1.2	Table Data	2	4.5	Markers	6
3.1.3	Mathematical Expressions	3	4.6	Colors	7

Nomenclature

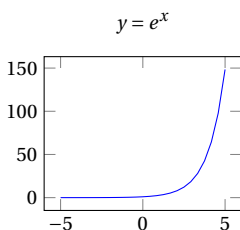
<code>\foo</code>	T _E X control sequence.
<code>foo_{env}</code>	L ^A T _E X environment <code>foo</code> .
<code>foo_{sty}</code>	PGFPLOTS style with key <code>foo</code> .
<code>foo</code>	Terminal symbol, literal text.
<code><foo></code>	Non-terminal symbol, metasyntactic variable.
<code><foo> → <bar></code>	Production rule; <code><foo></code> can be replaced by <code><bar></code> , <code><foo></code> and <code><bar></code> are implicit groups.
<code><foo> <bar></code>	Sequence; <code><foo></code> followed by <code><bar></code> .
<code><foo> <bar></code>	Choice; <code><foo></code> or <code><bar></code> .
<code><foo>*</code>	<code><foo></code> can occur zero or more times.
<code><foo>+</code>	<code><foo></code> can occur one or more times.
<code><foo>?</code>	<code><foo></code> is optional.
<code>()</code>	Explicit group.
<code>▷ <key> = <value></code>	User option, <code><key></code> and <code><value></code> are implicit groups.
<code>▷ <key></code>	User option without a value.
<code>42</code>	Default value is 42.
<code>↵</code>	Line continuation mark.
<code><empty></code>	Nothing.
<code><newline></code>	Newline character, <code>^^M</code> in T _E X.
<code><dimension></code>	A legitimate T _E X dimension.
<code><number></code>	$(-\infty, \infty) \cap \mathbb{R}$.
<code><positive number></code>	$(0, \infty) \cap \mathbb{R}$.
<code><non-negative number></code>	$[0, \infty) \cap \mathbb{R}$.
<code><integer></code>	$(-\infty, \infty) \cap \mathbb{Z}$.
<code><positive integer></code>	$(0, \infty) \cap \mathbb{Z}$.
<code><non-negative integer></code>	$[0, \infty) \cap \mathbb{Z}$.

1 General

1.1 Document Structure

```
\documentclass{standalone}
\usepackage{pgfplots}
\pgfplotsset{compat=1.16}

\begin{document}
\begin{tikzpicture}
\begin{axis}[title={$y = e^x$}]
\addplot+[no markers] {exp(x)};
\end{axis}
\end{tikzpicture}
\end{document}
```



1.2 PGFPlots Options

```
\pgfplotsset{<key/value list>}
<key/value list> → ((<key> = <value>),)*
```

Options are supplied as a $\langle \text{key/value list} \rangle$. The `/pgfplots/` and `/tikz/` prefixes in $\langle \text{key} \rangle$ can be omitted in the scope of PGFPlots commands. Please note that a trailing comma in $\langle \text{key/value list} \rangle$ does no harm.

1.3 Key Handlers

```
\pgfplotsset{<key>/.style = {<key/value list>}}
```

Define or replace style $\langle \text{key} \rangle$.

```
\pgfplotsset{<key>/.append style = {<key/value list>}}
```

Append to style $\langle \text{key} \rangle$.

```
\pgfplotsset{<key>/.code = {<TeX code>}}
```

Define or replace $\langle \text{key} \rangle$ that – when run – takes one argument; $\langle \text{TeX code} \rangle$ can refer to the supplied argument as $\#1$. Invoke as `\pgfplotsset{<key> = {<argument>}}`.

```
\pgfplotsset{<key>/.code 2 args = {<TeX code>}}
```

Like $\langle \text{key} \rangle /.code$ but with two arguments; $\langle \text{TeX code} \rangle$ can refer to the supplied arguments as $\#1$ and $\#2$. Invoke as `\pgfplotsset{<key> = {<first argument>}{<second argument>}}`.

```
\pgfplotsset{<key>/.cd}
```

Make $\langle \text{key} \rangle$ the default prefix.

1.4 Mathematical Expressions

See the TikZ/PGF manual for a detailed description.

Use parenthesis, (and), for grouping. Arguments and values of trigonometric functions are in degree angle.

Arithmetic Operators: +, - (also unary minus), *, /, ^ (exponentiation), ! (factorial, postfix operator), r (radian, postfix operator, see deg).

Relational Operators: ==, !=, <, <=, >, >=.

Logical Operators: ! (not, prefix operator), || (or), && (and).

Conditionals: $\langle \text{condition} \rangle ? \langle \text{true} \rangle : \langle \text{false} \rangle$.

Constants: pi, e, false, true.

Unary Functions: abs, sign, int, frac (fractional part), round, floor, ceil, factorial (see !), iseven, isodd, isprime, sqrt, exp, ln, log10, log2, sin, cos, tan, cot, sec, cosec, asin, acos, atan, deg (degree from radian), rad (radian from degree), sinh, cosh, tanh.

Binary Functions: div (integer division), mod, Mod (unsigned result), gcd, pow (see ^), atan2, vecLen (vector length in \mathbb{R}^2).

n-ary Functions: min, max.

Pseudo-Random Number Functions (Uniform Distribution): rnd ($[0, 1] \cap \mathbb{R}$), rand ($[-1, 1] \cap \mathbb{R}$), random(n) ($[1, n] \cap \mathbb{N}$), random(m, n) ($[m, n] \cap \mathbb{Z}$).

2 Axis Environments

```
\begin{axis}[<axis options>]?
<axis options> → <key/value list>
```

axis_{env} can also be $\text{semilogxaxis}_{\text{env}}$, $\text{semilogyaxis}_{\text{env}}$, or $\text{loglogaxis}_{\text{env}}$.

```
> every <type>? axis style
<type> → (linear|semilogx|semilogy|loglog)
Define default axis options.
```

▷ `xmode|ymode|zmode = normal|linear|log` option
 Customize axis scaling; `linear` is a synonym for `normal`.

▷ `log basis (x|y|z) = <empty>|<positive number>` option
 The basis for logarithmic axis scaling. Empty means to apply the natural logarithm (base e) to any input coordinate – if the axis scaling is logarithmic – and use the decadic/common logarithm (base 10) for displaying tick labels. Any non-empty value causes both, coordinates and tick labels, to use the logarithm with base $\langle number \rangle$.

3 Plots

`\addplot[<plot options>] ? <input data> <trailing TikZ path commands>;`
`\addplot` (without options) and `\addplot+[<plot options>]` utilize default options from the cycle list. `\addplot[<plot options>]` only use the manually provided options.

▷ `every axis plot (no n) ?` style
 Define $\langle plot options \rangle$ for all plots or for the n^{th} plot of every axis. Plot numbers are zero-based.

3.1 Input Data

▷ `empty line = auto|none|scanline|jump` option
 How to handle empty lines in $\langle coordinates list \rangle$, `none` means to do nothing, `jump` means to insert a discontinuity.

3.1.1 Coordinates List

$\langle input data \rangle \rightarrow coordinates \{ \langle coordinates list \rangle \}$
 $\langle coordinates list \rangle \rightarrow \langle coordinates \rangle^*$
 $\langle coordinates \rangle \rightarrow (x, y, z) (+- (u, v, w))^? ([\langle meta data \rangle])^?$

Read input data from a sequence of coordinates. x , y , and z are the point coordinates. u , v , and w are the error coordinates (reliability bounds) for error bar plots. Coordinate z and w are only mandatory for 3D plots. Empty lines in the $\langle coordinates list \rangle$ indicate discontinuities; use `\\` when gathering coordinates in a \TeX macro.

▷ `plot coordinates/math parser = true|false` option
 Whether or not to enable mathematical expressions in every coordinate inside of a $\langle coordinates list \rangle$.

3.1.2 Table Data

$\langle input data \rangle \rightarrow table [\langle table options \rangle]^? \{ \langle table data \rangle \}$
 $\langle table data \rangle \rightarrow \langle file name \rangle | \langle inline table \rangle$

Read input data from table columns.

▷ `table/<coordinate> = <column name>` option
 ▷ `table/<coordinate> index = <column index>` option
 ▷ `table/<coordinate> expr = <expression>` option
 $\langle coordinate \rangle \rightarrow x|y|z|(x|y|z) \text{ error } (\text{plus}|\text{minus})^?|meta$

Column names are case sensitive and have to exist. Use $\{ \langle column name \rangle \}$ to quote non-trivial column names. The first column has index zero. Within $\langle expression \rangle$ `\thisrow{<column name>}` and `\thisrowno<column index>` yields the cell value of the specified column. Likewise, `\coordindex` yields the index of the current set of coordinates and `\lineno` yields the total line number. Both numbers start counting at zero.

▷ `table/header = true|false` option
 Whether or not to check $\langle table data \rangle$ for column names. If enabled, the first non-comment line is checked for column names. That means if any element is not a number, all entries are treated as column names.

▷ `table/skip first n = 0|<non-negative integer>` option
 Don't process the first n lines in $\langle table data \rangle$.

▷ `table/ignore chars = { }|<comma-separated list>` option
 ▷ `table/white space chars = { }|<comma-separated list>` option
 ▷ `table/comment chars = { }|<comma-separated list>` option

Extra characters to be ignored, treated like a whitespace character (beside space and tab), or treated like a comment start character (beside `#` and `%`).

▷ `table/row sep = <newline>|\\` option
 Use `\\` as the row separator if you experience problems with $\langle newline \rangle$, for example with inline table data or when gathering table data in a \TeX macro.

▷ `table/col sep = space|tab|comma|semicolon|colon|
|braces|&|ampersand` option

A space column separator means one or more space or tab characters. With braces, every table cell looks like $\{\langle contents \rangle\}$ and whitespace characters between adjacent table cells is ignored. A `&` column separator implies `'table/trim cells = true'`.

▷ `table/read completely = auto|true|false` option
Whether or not to read the whole table into memory. Use with care!

▷ `table/search path = {}|<comma-separated list>` option

▷ `table/search path/implicit . = true|false` option
Search path for input files, `.` means to use the standard \TeX procedure.

```
\pgfplotstableread{<file name>}\foo
\addplot table [<table options>] {\foo};
```

Read table data once so that you can use it multiple times; `\foo` is a user-defined command sequence.

3.1.3 Mathematical Expressions

$\langle input\ data \rangle \rightarrow expression^? \{ \langle expression \rangle \}$

$\langle input\ data \rangle \rightarrow (\langle x-expression \rangle, \langle y-expression \rangle, \langle z-expression \rangle)$

Create input data by sampling a mathematical expression over an argument domain. The second form can be used to create parametric plots. Say $\{ \langle x-expression \rangle \}$ if $\langle x-expression \rangle$ contains parenthesis or commas. The $\langle z-expression \rangle$ is only mandatory for 3D plots.

▷ `domain = -5:5| $\langle x_1 \rangle : \langle x_2 \rangle$` option

▷ `domain y = <empty>| $\langle y_1 \rangle : \langle y_2 \rangle$` option

Define the argument domain for the x-axis to the closed interval $[x_1, x_2]$. Likewise for the y-axis for 3D plots. If `domain y` is empty, use the value of `domain`.

▷ `samples = 25|<non-negative integer>` option

▷ `samples y = <empty>|<non-negative integer>` option

The number of samples to be generated. Samples are equally spaced over the corresponding argument domain. If `'samples y'` is empty, use the value of `samples`.

▷ `samples at = {}|<comma-separated list of numbers>` option

Explicit argument values for sampling $\langle expression \rangle$. This option always overrides the `domain` and `samples` options.

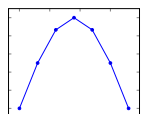
$\langle comma-separated\ list\ of\ numbers \rangle$ can contain `...` expressions, for example `'{-2, -1.8, ..., 2}'`.

▷ `variable = x|<variable name>` option

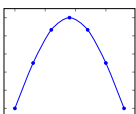
▷ `variable y = y|<variable name>` option

The variable name containing the argument value when evaluating $\langle expression \rangle$.

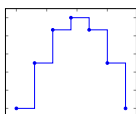
3.2 Line Plots



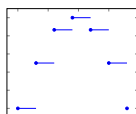
sharp plot



smooth



const plot



jump mark left

▷ `/tikz/sharp plot` option

Connect points by straight lines. This is the default.

▷ `/tikz/smooth` option

▷ `/tikz/tension = 0.55|<number>` option

Connect points by a smooth curve. For best results, points should be equidistant and the bending angles should be less than about 30° . The `tension` option controls the sharpness of the corners; 0 yields sharp corners and 1 yields a circle if the path is a square.

▷ `/tikz/const plot` option

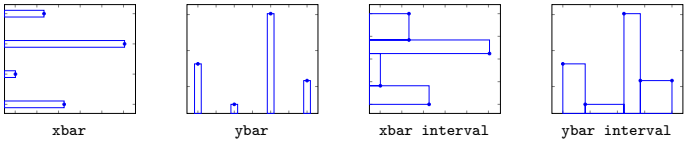
▷ `/tikz/const plot mark (left|mid|right)` option

Connect points with horizontal and vertical line segments. `'const plot'` is an alias for `'const plot mark left'`. Markers are placed on the left corner, in the middle, or on the right corner of the horizontal line segments. Use `'const plot, no markers'` to omit the markers.

▷ `/tikz/jump mark (left|mid|right)` option

Like `'const plot'` but omit the vertical line segments.

3.3 Bar Plots



- ▷ `/tikz/xbar` option
- ▷ `/tikz/ybar` option

Render coordinates as horizontal or vertical bars respectively.

- ▷ `/pgf/bar width = 10pt | <dimension> | <number>` option

Width of a single bar. $\langle dimension \rangle$ is a $\text{T}_{\text{E}}\text{X}$ dimension and $\langle number \rangle$ is in axis units. Value can be a mathematical expression. The fully computed value is then available in `\pgfplotbarwidth`.

- ▷ `/pgf/bar shift = 0pt | <dimension> | <number>` option

Off-center distance for the bars. $\langle dimension \rangle$ is a $\text{T}_{\text{E}}\text{X}$ dimension and $\langle number \rangle$ is in axis units. Value can be a mathematical expression. The fully computed value is then available in `\pgfplotbarshift`.

- ▷ `xbar` style

- ▷ `xbar(= 2pt | <dimension> | <number>)?` option

- ▷ `ybar` style

- ▷ `ybar(= 2pt | <dimension> | <number>)?` option

Predefined axis style for bar plots; implies `/tikz/xbar` or `/tikz/ybar` respectively, `bar shift autosty`, and `bar cycle liststy`. The default handler takes one optional argument which is passed on to `bar shift autosty`.

- ▷ `bar shift auto` style

- ▷ `bar shift auto = 2pt | <dimension> | <number>` option

Predefined axis style setting `/pgf/bar shift` to the correct value based on the current plot number and the total number of plots. Argument is the distance between adjacent bars of a group.

When n bar plots are added to an axis, the total width for a group of bars is $n \times \langle bar width \rangle + (n - 1) \times \langle bar shift auto \rangle$.

- ▷ `bar cycle list` style

Predefined axis style installing a cycle list for bar plots.

- ▷ `bar direction = auto|x|y` option

Explicitly set the bar plot direction. Not needed if you say, for example ‘`ybar`, `bar width = 1`’, because the direction is clear from the context.

- ▷ `/tikz/xbar interval` option

- ▷ `/tikz/ybar interval` option

Like `/tikz/xbar` or `/tikz/ybar` respectively, but draw the bar width as an interval from this point to the next point. You need one extra point to define the interval for the last bar.

- ▷ `xbar interval` style

- ▷ `xbar interval(= 1 | <relative width>)?` option

- ▷ `ybar interval` style

- ▷ `ybar interval(= 1 | <relative width>)?` option

Predefined axis style for interval bar plots; implies `/tikz/xbar interval` or `/tikz/ybar interval` respectively and `bar cycle liststy`. The default handler takes one optional argument to scale the intervals.

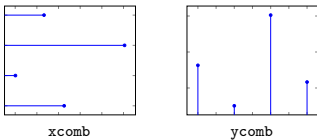
- ▷ `xticklabel interval boundaries` style

- ▷ `yticklabel interval boundaries` style

- ▷ `zticklabel interval boundaries` style

Axis style to display the interval bounds in the tick labels.

3.4 Comb Plots



- ▷ `/tikz/xcomb` option
- ▷ `/tikz/ycomb` option

Render coordinates as horizontal or vertical lines respectively.

3.5 Quiver Plots

▷ `quiver = {<quiver options>}` option

Render coordinates as small arrows. The origin of the arrow is at the final point coordinates (x, y, z) and the direction and length of the arrow is defined by the direction coordinates (u, v, w) .

The `quiver/` prefix can be omitted within `<quiver options>`.

▷ `quiver/(u|v|w) = 0|<expression>` option

The direction coordinates of the arrows. Within `<expression>`, x , y , and z are bound to the final point coordinates.

For parametric plots use `'variable = t'` and `'quiver/u = f(t)'` and `'quiver/v = g(t)'` to access the parameter.

```
\addplot[
  variable = t,
  quiver = {u = {-sin(t)}, v = {cos(t)}},
]
({cos(t)}, {sin(t)});
```

▷ `quiver/(u|v|w) value = 0|<number>` option

Like `quiver/u`, `quiver/v`, and `quiver/w` respectively but without parsing mathematical expressions. However, `\thisrow{<column name>}` and similar code works.

▷ `quiver/colored` option

▷ `quiver/colored = mapped color|<color>` option

Set a different color for each arrow. `quiver/colored` is an alias for `'quiver/colored = mapped color'`. Please note that `'<color>, quiver = ...'` is more efficient if `<color>` is constant.

▷ `quiver/scale arrows = 1|<number>` option

Scale all arrows by a constant factor.

▷ `quiver/update limits = true|false` option

Whether or not the coordinates of the arrow heads shall be considered when determining the axis limits.

▷ `quiver/every arrow` style

Style to customize arrows individually at visualization time.

▷ `quiver/before arrow` code

▷ `quiver/after arrow` code

Run `<TeX code>` before and after drawing a single arrow. Empty by default.

▷ `quiver/quiver legend` style

Style that redefines `legend image` code in order to produce a suitable legend for quiver plots.

4 Lines and Markers

4.1 Line Width

▷ <code>/tikz/ultra thin</code>		style
▷ <code>/tikz/very thin</code>		style
▷ <code>/tikz/thin</code>		style
▷ <code>/tikz/semithick</code>		style
▷ <code>/tikz/thick</code>		style
▷ <code>/tikz/very thick</code>		style
▷ <code>/tikz/ultra thick</code>		style

Predefined line widths.

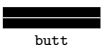
▷ `/tikz/line width = 0.4pt|<dimension>` option

Set the line width.

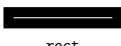
4.2 Line Cap

▷ `/tikz/line cap = butt|rect|round` option

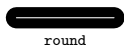
Set the line cap style.



butt



rect



round

4.3 Line Join

▷ `/tikz/line join = miter|bevel|round` option

Set the line join style.



miter



bevel
















round

▷ `/tikz/miter limit = 10|<number>` option

When the ratio of the miter length to the line width is greater than `<number>`, the miter join is replaced by a bevel. A miter limit $\ell = 1/\sin(\alpha/2)$ for $\alpha \in (0^\circ, 180^\circ]$ will create a bevel join for angles less than $\alpha = 2 \cdot \arcsin(1/\ell)$.

4.4 Dash Pattern

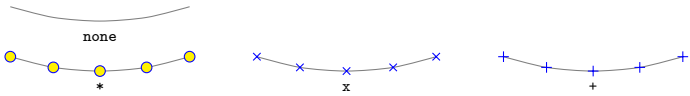
▷ <code>/tikz/solid</code>		style
▷ <code>/tikz/dashed</code>		style
▷ <code>/tikz/dotted</code>		style
▷ <code>/tikz/dashdotted</code>		style
▷ <code>/tikz/dashdotdotted</code>		style
▷ <code>/tikz/densely dashed</code>		style
▷ <code>/tikz/densely dotted</code>		style
▷ <code>/tikz/densely dashdotted</code>		style
▷ <code>/tikz/densely dashdotdotted</code>		style
▷ <code>/tikz/loosely dashed</code>		style
▷ <code>/tikz/loosely dotted</code>		style
▷ <code>/tikz/loosely dashdotted</code>		style
▷ <code>/tikz/loosely dashdotdotted</code>		style

Predefined line styles.

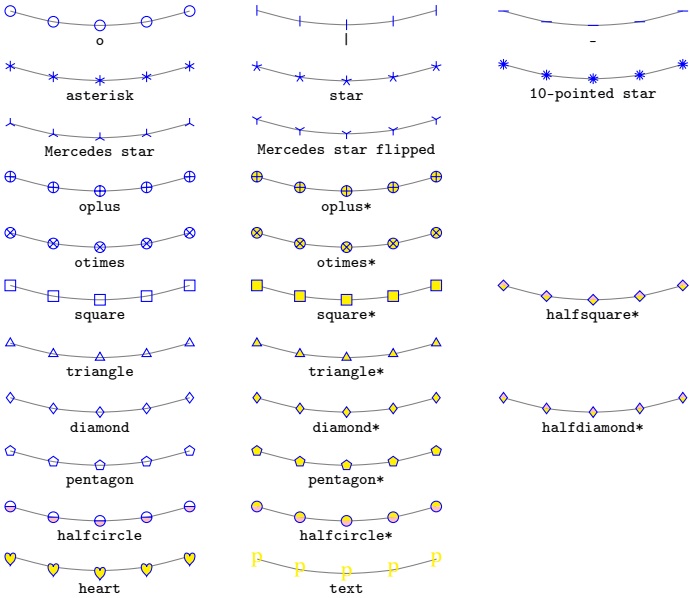
- ▷ `/tikz/dash pattern = ((on|off) <dimension>)+` option
Set the dash pattern (line style) for drawing lines, e.g.,
'dash pattern = on 3.5mm off 0.7mm'.
- ▷ `/tikz/dash phase = 0pt|<dimension>` option
Start the dash pattern at offset <dimension>.

4.5 Markers

Standard markers:



With `\usetikzlibrary{plotmarks}`:



All markers plotted with

'mark options = {draw = blue, fill = yellow}' and
'mark color = pink'. You can rotate makers with, e.g.,
'mark options = {rotate = 90}'.

- ▷ `/tikz/mark = *|<marker>` option
Use <marker>.
- ▷ `/tikz/mark size = 2pt|<dimension>` option
Marker size, <dimension> is either the radius or about half the width or height.
- ▷ `/tikz/mark repeat = 1|<integer>` option
Draw a marker at every <integer>th sample.
- ▷ `/tikz/mark phase = 1|<integer>` option
Draw the first marker at the <integer>th sample; <integer> is one based.
- ▷ `/tikz/mark indices = { }|{<comma-separated list of integers>}` option
Explicit sample indices for drawing markers.
<comma-separated list of integers> can contain ... expressions, for example
'mark indices = {1, 2, ..., 7}'.
- ▷ `/tikz/every mark` style
This style is applied before drawing a marker.

- ▷ `/tikz/mark options = {<options>}` option
 Redefine ‘every mark’ so that it sets <options>.
- ▷ `/pgfplots/no markers` style
 Disable markers; even for cycle lists that contain markers.
- ▷ `/pgf/mark color = white|<color>` option
 Additional fill color for `halfcircle`, `halfcircle*`, `halfdiamond*`, and `halfsquare*` markers.
- ▷ `/pgf/text mark = p|<text>` option
 Define the text for ‘mark = text’.
- ▷ `/pgf/text mark as node = false|true` option
 Whether or not to draw text markers as nodes.
- ▷ `/pgf/text mark style = {<options>}` option
 Customize the appearance of text markers. When ‘text mark as node’ is true, ‘text mark style’ are `\node` options. Otherwise, ‘text mark style’ are `\pgftext` options.


4.6 Colors

Color support is provided by the `xcolor` package. Standard color names:

 black	 red	 green	 blue
 darkgray	 cyan	 magenta	 yellow
 gray	 brown	 lime	 olive
 lightgray	 orange	 pink	 purple
 white	 teal	 violet	none

- ▷ `/tikz/color = <color>` option
 Set the color for drawing and filling. You can omit the option key if <color> is a color name.
- ▷ `/tikz/draw = <color>` option
 ▷ `/tikz/fill = <color>` option
 Set the color for drawing or filling respectively. You can use `none` as <color> to disable drawing or filling.

```
\definecolor{<name>}{<model>}{<spec>}
<model> → rgb|cmy|cmyk|hsb|Hsb|tHsb|gray|RGB|HSB|Gray|HTML|
|wave
<rgb spec> → x, x, x
<cmy spec> → x, x, x
<cmyk spec> → x, x, x, x
<hsb spec> → x, x, x
<Hsb spec> → H, x, x
<tHsb spec> → H, x, x
<gray spec> → x
<RGB spec> → L, L, L
<HSB spec> → M, M, M
<Gray spec> → N
<HTML spec> → [00000016, FFFFFFFF16]
<wave spec> → [363, 814]
x = [0, 1], H = [0, 360], L = [0, 255] ∩ ℤ, M = [0, 240] ∩ ℤ, and N = [0, 15] ∩ ℤ.
All colors are defined in the sRGB color space. HSB is a synonym for HSL.
```

 unired	 unigreen	 uniblue	 unigray1
 uniorange	 unisea	 univiolet	 unigray3
 uniyellow	 unicyan	 unimagenta	 unigray2
 unilawn	 unisky	 unirose	 unigray3

```
\definecolor{unired}{HTML}{D82F00}
\definecolor{uniorange}{HTML}{DC7500}
\definecolor{uniyellow}{HTML}{D8AB00}
\definecolor{unilawn}{HTML}{7D9700}
\definecolor{unigreen}{HTML}{007C00}
\definecolor{unisea}{HTML}{00AC9B}
\definecolor{unicyan}{HTML}{27D0FF}
\definecolor{unisky}{HTML}{009EFF}
\definecolor{uniblue}{HTML}{2754FF}
\definecolor{univiolet}{HTML}{B565FF}
\definecolor{unimagenta}{HTML}{FF83FF}
\definecolor{unirose}{HTML}{FF3687}
\definecolor{unigray1}{HTML}{6C6C6C}
\definecolor{unigray2}{HTML}{B6B6B6}
\definecolor{unigray3}{HTML}{919191}
```

These colors are perceptually uniform, i.e., the primary colors red, green, and blue have similar lightness in the CIE $L^*a^*b^*$ color space. Likewise for the secondary colors cyan, magenta and yellow. They also satisfy the RGB and CMY color models. The gray levels have the same lightness as the primary, secondary, and tertiary colors.

Option Index

B		u value	5
bar cycle list _{sty}	4	update limits	5
bar direction	4	v	5
bar shift	4	v value	5
bar shift auto _{sty}	4	w	5
bar width	4	w value	5
C		S	
color	7	samples	3
const plot	3	samples at	3
const plot mark left	3	samples y	3
const plot mark mid	3	semithick _{sty}	5
const plot mark right	3	sharp plot	3
D		smooth	3
dash pattern	6	solid _{sty}	6
dash phase	6	T	
dashdotdotted _{sty}	6	table/	
dashed _{sty}	6	col sep	2
densely dashdotdotted _{sty}	6	comment chars	2
densely dashed _{sty}	6	header	2
densely dotted _{sty}	6	ignore chars	2
domain	3	meta	2
domain y	3	read completely	3
dotted _{sty}	6	row sep	2
draw	7	search path	3
E		search path/ implicit	3
empty line	2	skip first n	2
every axis plot	2	white space chars	2
every linear axis	1	x	2
every loglog axis	1	x error	2
every mark _{sty}	6	x error minus	2
every semilogx axis	1	x error plus	2
every semilogy axis	1	y	2
F		y error	2
fill	7	y error minus	2
J		y error plus	2
jump mark left	3	z	2
jump mark mid	3	z error	2
jump mark right	3	z error minus	2
L		z error plus	2
line cap	5	tension	3
line join	5	text mark	7
line width	5	text mark as node	7
log basis	2	text mark style	7
loosely dashdotdotted _{sty}	6	thick _{sty}	5
loosely dashed _{sty}	6	thin _{sty}	5
loosely dotted _{sty}	6	U	
M		ultra thick _{sty}	5
mark	6	ultra thin _{sty}	5
mark color	7	V	
mark indices	6	variable	3
mark options	6	variable y	3
mark phase	6	very thick _{sty}	5
mark repeat	6	very thin _{sty}	5
mark size	6	X	
N		xbar	4
no markers	7	xbar _{sty}	4
P		xbar interval	4
plot coordinates/ math parser	2	xbar interval _{sty}	4
Q		xcomb	4
quiver	5	xmode	1
quiver/ after arrow _{code}	5	xticklabel interval boundaries _{sty}	4
before arrow _{code}	5	Y	
colored	5	ybar	4
every arrow _{sty}	5	ybar _{sty}	4
quiver legend _{sty}	5	ybar interval	4
scale arrows	5	ybar interval _{sty}	4
u	5	ycomb	4
		ymode	1
		yticklabel interval boundaries _{sty}	4
		Z	
		zmode	1
		zticklabel interval boundaries _{sty}	4

Concept Index

@	
+~	2
A	
\addplot	2
.append style key handler	1
auto	2
axisenv	1
axis scaling	2
basis for logarithm	2
C	
.cd key handler	1
.code 2 args key handler	1
.code key handler	1
code option	<i>see</i> key handler
coordinates	2
coordinates list	
input data	2
\coordindex	2
D	
dash pattern	6
dash phase	6
E	
expression	3
H	
handler	<i>see</i> key handler
I	
input data	
coordinates list	2
table data	2
J	
jump	2
K	
key handler	
.append style	1
.cd	1
.code 2 args	1
.code	1
.style	1
L	
line style	6
line width	5

