
Boomslang Documentation

Release 1.2

Alex Rasmussen

May 02, 2016

1	Introduction	3
1.1	Prerequisites	3
1.2	Installation	3
1.3	As a Python Egg (via pip or easy_install)	3
1.4	Installing the development version	3
2	Getting Started	5
2.1	Design Principles	5
2.2	Writing your First Boomslang Script	5
3	Examples	7
4	Plot Elements	9
4.1	PlotInfo: The Common Plot Element Base Class	9
4.2	Simple Plot Elements	10
4.3	Compound Plot Elements	12
4.4	Miscellaneous Plot Elements	13
5	Styling Plot Elements	15
5.1	Markers	15
5.2	Labels	16
5.3	Line Styles	16
5.4	Colors	16
6	Plots	17
6.1	Grid Lines	19
6.2	Locations	20
7	Layouts	21
8	Utility Functions	23
9	Plotting from Django	25
	Python Module Index	27

Boomslang is an abstraction layer over Matplotlib designed for the creation of re-usable, modular plotting code.

Introduction

I created Boomslang to decrease the amount of boilerplate code I had to write when producing plots for research papers. Boomslang treats both plots and plot data as objects. This allows functions to return plot data or plots, and allows plots to be flexibly and cleanly composed. Boomslang also abstracts away much of Matplotlib's syntax, which can be confusing and brittle if not used properly.

1.1 Prerequisites

Boomslang requires a recent version of [matplotlib](#).

1.2 Installation

1.3 As a Python Egg (via pip or easy_install)

The latest stable version of Boomslang is available from [easy_install](#) or [pip](#).

```
easy_install boomslang  
pip install boomslang
```

This will install Boomslang in your Python installation's site-packages directory.

1.4 Installing the development version

If you want the latest features, it's probably best to use the development version of Boomslang until it reaches a point where I bring it out of beta. To do this, install [git](#) and clone the repository:

```
git clone git://github.com/alexras/boomslang.git
```

Once you've cloned Boomslang, you can either add it to your *PYTHONPATH* or symlink it to your site-packages directory.

Getting Started

Boomslang is designed to make creating common kinds of plots easier. In this section of the documentation, we'll look at how Boomslang tries to do that and how to start building plots with Boomslang.

2.1 Design Principles

Plotting code should be reusable and modular. Traditional wisdom holds that a plot's axes and the things that are plotted on those axes (lines, points, bars) are inseparable. Boomslang chooses to separate the axes (which we call a *plot*) from the things that are drawn on those axes (which we call *plot elements*).

Matplotlib is complicated. Boomslang should be easy. Matplotlib is an extremely powerful library, but its power comes at a cost in terms of complexity. Boomslang aims to abstract most or all of that complexity away from the end user.

Boomslang is not Matplotlib. There are many kinds of graphs that Matplotlib can generate and Boomslang cannot. Boomslang's goal is not to be a general-purpose figure plotting engine, but to provide an easier alternative to Matplotlib for plots that people find themselves wanting to generate the majority of the time.

2.2 Writing your First Boomslang Script

Let's walk through a simple example of how to use Boomslang. Consider the code below:

```
1 line = Line()
2 line.yValues = [25, 40, 30, 23, 10, 50]
3 line.xValues = range(len(line.yValues))
4
5 plot = Plot()
6 plot.add(line)
7 plot.xLabel = "X Label"
8 plot.yLabel = "Y Label"
9 plot.yLimits = (0, 60)
10 plot.save("simpleline.png")
```

On line 1, we create a plot element, in this case a `Line`. Plot elements are simply objects that know how to draw themselves. The values of their fields determines what gets drawn. In this case, we're setting the line's x and y values. We could also set things like the line's color, width, and whether it is dashed or dotted. See the API documentation for more information on the different kind of plot elements and their properties.

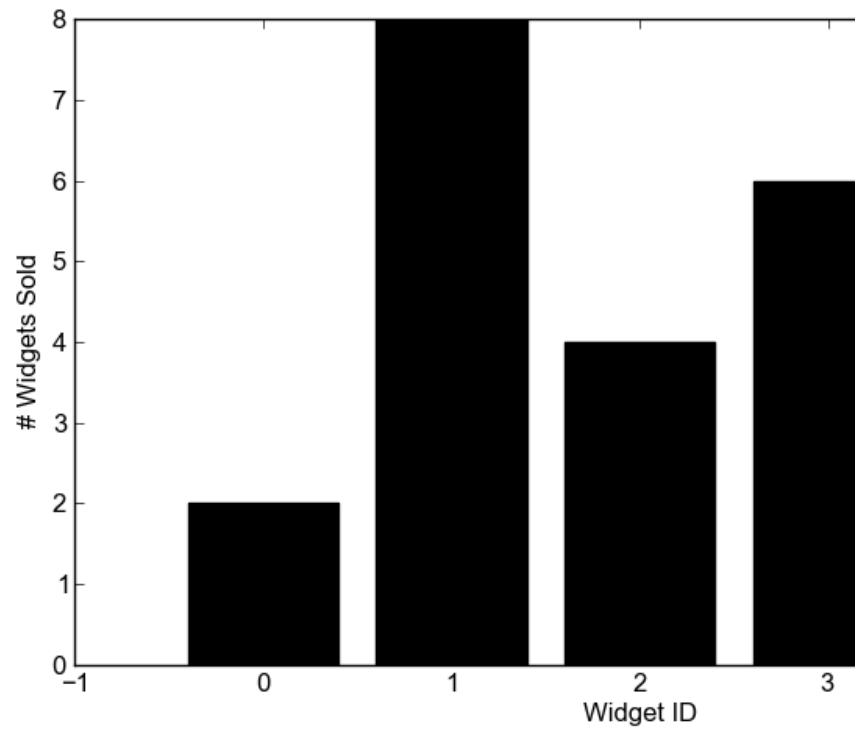
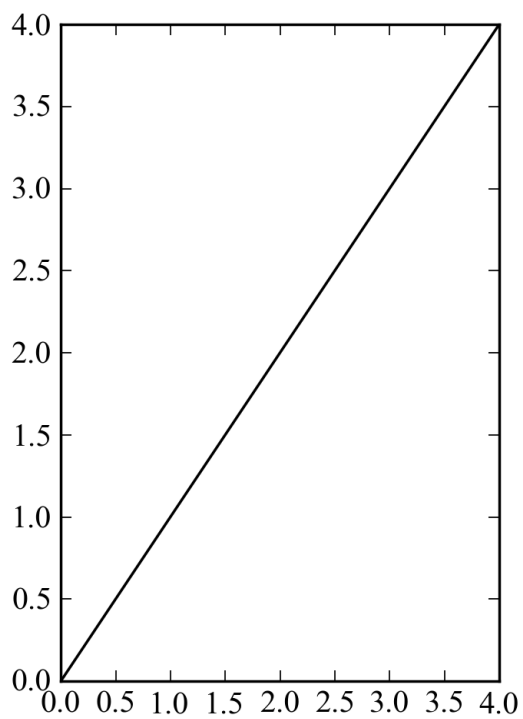
On line 4, we create a `Plot`. You can think of plots as containers for plot elements, but also as objects that know how to draw themselves. On line 5, we add the line to the plot using the `Plot.add()` method.

The next few lines do things like setting the labels of the plot's X and Y axes, and setting bounds on the Y axis.

The last line saves the plot to a file. This feature uses Matplotlib's underlying save functionality, and can support a wide range of formats. If you want an interactive view of the plot, you could call `plot.plot()`.

Examples

This page provides a gallery of example Boomslang scripts. Clicking on one of the images below will direct you to that image and a listing of its source code.



Plot Elements

Plot elements in Boomslang are represented by objects that encapsulate all the state that a plot element needs to draw itself. The one required method that all plot elements must implement is *draw(self, fig, axes)*, which the element uses to draw itself within the figure *fig* on the axes *axes*.

4.1 PlotInfo: The Common Plot Element Base Class

Many of Boomslang's plot elements inherit from `PlotInfo`. `PlotInfo` contains several useful fields (x-axis and y-axis values, error bars, etc.)

```
class boomslang.PlotInfo.PlotInfo (plotType, xValues=None, yValues=None, xTickLabels=None,
                                   yTickLabels=None, xTickLabelPoints=None, yTickLabelPoints=None,
                                   xTickLabelProperties=None, yTickLabelProperties=None, label=None,
                                   yMins=None, yMaxes=None, yErrors=None, autosort=True, xLimits=None,
                                   errorBarColor=None, errorBarLineWidth=None, errorBarCapSize=None,
                                   picker=None)
```

autosort = None

If true, x/y pairs in `xValues` and `yValues` are sorted by x value before the plot is rendered

draw (*fig, axis, transform=None*)

The base method for drawing a plot element on the axis *axis* within the figure *fig*. Other plot elements should override this method, but should also be sure to call this method before doing any element-specific processing.

drawErrorBars (*axis, transform=None*)

Draw error bars for the plot element using the plot element's *yMins/yMaxes* or *yErrors*. If *yMins* or *yMaxes* have non-zero length, they are used to draw error bars that can be asymmetric. Otherwise, *yErrors* are used to draw symmetric error bars.

If some transform should be applied to the error bars, it is provided by the caller with the *transform* parameter.

getAttributes ()

Get the attributes dictionary used to style the plot element. Extending this method allows plot elements to inherit attributes.

label = None

A string used to label this plot element in a legend

plotType = None

Names the type of plot element

split (*pieces*)

Split this plot element into *pieces* separate plot elements, each of which is responsible for a disjoint range of the original plot element's x-axis. This is useful for situations where a plot element cannot fit comfortably in a single plot.

xTickLabelPoints = None

The locations on the x-axis where the labels in `xTickLabels` should be drawn

xTickLabelProperties

A dictionary of properties that control the appearance of the X axis' tick labels. See [Labels](#) for more information on which properties can be set.

xTickLabels = None

A list of labels that should be drawn on the x-axis

xValues = None

An array of x-axis values for the plot element

yErrors = None

For symmetric error bars, a list of error bar widths for this plot element's error bars

yMaxes = None

For asymmetric error bars, a list of locations for the tops of this plot element's error bars

yMins = None

For asymmetric error bars, a list of locations for the bottoms of this plot element's error bars

yTickLabelPoints = None

The locations on the x-axis where the labels in `xTickLabels` should be drawn

yTickLabelProperties

A dictionary of properties that control the appearance of the Y axis' tick labels. See [Labels](#) for more information on which properties can be set.

yTickLabels = None

A list of labels that should be drawn on the y-axis

yValues = None

An array of y-axis values for the plot elements

4.2 Simple Plot Elements

Boomslang comes equipped with the standard plot elements for making lines, bar graphs, scatter plots, and box-and-whisker plots.

```
class boomslang.Line.Line(color='black', width=1, lineStyle='-', marker=None, markerSize=8.0,  
                           dates=False, loglog=False, steps=None, alpha=None, antialiased=False,  
                           **kwargs)
```

A plot element that represents a line in two-dimensional space

antialiased = None

If True, the line will be anti-aliased, removing jagged edges in some output formats

color = None

The line's color. see [Colors](#) for valid colors.

dates = None

If True, the x-axis values of this line will be interpreted as dates

lineStyle

The line's style. See [Line Styles](#) for more information about available line styles.

lineWidth = None

The line's width

marker

The marker used to mark points on the line. See [Markers](#) for more information on available marker types.

markerSize

The size of the markers used to mark points on the line.

steps

If not None, the line will be plotted as a step function. If "pre", each point comes before a step. If "mid", each point is in the middle of a step. If "post", each point comes after a step.

class boomslang.HLine.**HLine** (*marker=None, markerSize=8.0, width=1.0, color='black', lineStyle='-', dates=False, **kwargs*)

A plot element that represents a horizontal line that passes through a given point on the y-axis.

color = None

The line's color. See [Colors](#) for valid colors.

lineStyle

The line style for this line; see [Line Styles](#) for more information about available line styles.

marker

The marker used to mark points on the line. See [Markers](#) for more information about available markers.

markerSize

The size of this line's markers.

width = None

The line's width

class boomslang.VLine.**VLine** (*marker=None, markerSize=8.0, width=1.0, color='black', lineStyle='-', dates=False, **kwargs*)

color = None

The line's color. See [Colors](#) for valid colors.

lineStyle

The line style for this line; see [Line Styles](#) for more information about available line styles.

marker

The marker used to mark points on the line. See [Markers](#) for more information about available markers.

markerSize

The size of this line's markers.

width = None

The line's width

class boomslang.Scatter.**Scatter** (*marker='o', markerSize=20, color='black', **kwargs*)

color = None

The color of points in the scatter plot. See [Colors](#) for valid colors.

marker

The marker type used to mark points in the scatter plot. See [Markers](#) for valid marker types.

markerSize

The size of the scatter plot's markers.

class `boomslang.Bar.Bar` (*label=None, labelBars=False, width=0.8, linewidth=1.0, color='black', edge-Color=None, hatch=None, align='center', **kwargs*)

A bar chart consisting of a single series of bars.

align = None

If *center* (default), *xValues* values denote the center of each bar. If *edge*, *xValues* values denote the left edge of each bar.

color = None

The color of the bars

edgeColor = None

The color of the line that borders each bar

hatch = None

The style of hatching used within the bar, if any. Valid values are any marker type, /, //, \ and \

linewidth = None

The width of the line that borders each bar

padding = None

Amount of padding between the edge of the left- and rightmost bars and the bounding box for the plot

width = None

The width of each bar in the set of bars

class `boomslang.BoxAndWhisker.BoxAndWhisker`

A modified box-and-whisker plot.

width = None

The width of each of the element's "boxes".

xSequence = None

A list of lists defining the values to be plotted. Each list gives a set of y-axis values. The first list gives the values for x=0, the second for x=1, and so on.

4.3 Compound Plot Elements

Boomslang can combine certain plot elements together to easily create stacks and clusters of plot elements while maintaining the individual elements themselves, making it easy to re-compose them by changing a couple of lines of code.

class `boomslang.ClusteredBars.ClusteredBars`

A clustered bar chart consisting of multiple series of bars with the same X axis values.

add (bar)

Add *bar* (a `boomslang.Bar.Bar`) to the cluster of bars

barWidth = None

The width of each bar in the cluster

deduplicateLegend = None

If multiple legend items have the same key, only show the first one

padding = None

Amount of padding between the edge of the left- and rightmost bars and the bounding box for the plot

spacing = None

The spacing between each bar in the cluster

width

The overall width of each cluster of bars

class `boomslang.StackedBars.StackedBars`

A stacked bar chart consisting of multiple series of bars with the same X axis values

add (*bar*)

Add *bar* (a `boomslang.Bar.Bar`) to the stack

barWidth = None

The width of the bars in the stack

padding = None

Amount of padding between the edge of the left- and rightmost bars and the bounding box for the plot

spacing = None

The spacing between stacks of bars

class `boomslang.StackedLines.StackedLines`

Lines stacked one atop the other, with colors filling in the gaps

addLine (*line*, *color*='white')

Add *line* (a `boomslang.Line.Line`) to the stack of lines, coloring the gap between it and the previous line with *color*

4.4 Miscellaneous Plot Elements

class `boomslang.Label.Label` (*x*, *y*, *text*=None, *bbox*=None)

Labels a point on the plot with text and/or arrows

hasArrow (*style*='->', *color*='black')

Defines an arrow between the label's text and its point. Valid arrow styles are given in [Matplotlib's documentation](#).

labelProperties

A dictionary of properties that control the appearance of the label. See [Labels](#) for more information on which properties can be set.

marker

The marker type that should be used to mark the labeled point

text = None

The text that should be displayed with the label

x = None

The label's x coordinate

y = None

The label's y coordinate

Styling Plot Elements

This page describes the various ways that Boomslang’s plot elements can be styled. Classes described below are typically used as Python properties, allowing them to be treated like normal fields with the benefits of aliasing and argument checking.

5.1 Markers

Markers are used to annotate or distinguish particular points on the Cartesian plane. There are a large number of options for configuring markers in matplotlib; Boomslang exports a subset of them in a way that’s a little easier to deal with.

Valid marker styles are as follows:

Description	Valid Values of <code>boomslang.Marker.Marker.marker</code>
No marker	"", "none", None
Points	., "point", "points"
Pixels	, "pixel", "pixels"
Circles	o, "circle", "circles"
Triangles Pointing Down	v, "down triangle", "down triangles"
Triangles Pointing Up	^, "up triangle", "up triangles"
Triangles Pointing Left	<, "left triangle", "left triangles"
Triangles Pointing Right	>, "right triangle", "right triangles"
Squares	s, "square", "squares"
Pentagons	p, "pentagon", "pentagons"
Stars	*, "star", "stars"
Hexagons with Parallel Sides Vertical	h, "hexagon", "hexagons", "vertical hexagon", "vertical hexagons"
Hexagons with Parallel Sides Horizontal	H, "horizontal hexagon", "horizontal hexagons"
Pluses	+, "plus", "pluses", "plusses"
Crosses (Xs)	x, "cross", "crosses"
Thick Diamonds	D, "diamond", "diamonds"
Thin Diamonds	d, "thin diamond", "thin diamonds"
Vertical Lines	, "vline", "vlines", "vertical line", "vertical lines"
Horizontal Lines	_, "hline", "hlines", "horizontal line", "horizontal lines"

5.2 Labels

Labels include things like tick labels (the numbers on the axes themselves), axis labels (labels that apply to an entire axis), and the `boomslang.Label.Label` plot element. They can also be styled in a number of different ways.

Valid label style properties are as follows:

Property	Description
<code>alpha</code>	Transparency of the label in [0.0, 1.0]
<code>backgroundcolor</code>	The label's background color
<code>color</code>	The label's text color
<code>fontsize</code>	The size of the label's font
<code>horizontalalignment</code>	The horizontal alignment of the label's text ('left', 'center', or 'right')
<code>linespacing</code>	The spacing between label lines (multiple of font size)
<code>multialignment</code>	Alignment for multiple lines of text ('left', 'center', or 'right')
<code>rotation</code>	Angle of the label's rotation in degrees
<code>rotation_mode</code>	Text rotation mode ("anchor" to align before rotating, None (default) to rotate before aligning)
<code>stretch</code>	Label's font stretch (horizontal condensation or expansion)
<code>style</code>	The font's style ('normal', 'italic', or 'oblique')
<code>verticalalignment</code>	The vertical alignment of the label's text ('top', 'center', 'bottom', or 'baseline')
<code>weight</code>	The font's weight (a number in [0, 1000] or one of 'ultralight', 'light', 'normal', 'regular', 'book', 'medium', 'roman', 'semibold', 'demibold', 'demi', 'bold', 'heavy', 'extra bold', 'black')

5.3 Line Styles

Boomslang has a standard level of control over how lines are rendered. At the moment, they can be solid, dashed, dotted, or alternate between dashes and dots. Boomslang includes some aliases for these line styles so that you don't have to remember the specific Matplotlib glyphs for each style.

Valid line styles are as follows:

Description	Valid line style values
Solid	–, “solid”, None
Dashed	--, “dashed”
Alternating Dash and Dot	– ., “dash-dot”, “dot-dash”
Dotted	:, “dotted”

5.4 Colors

Colors in Boomslang can be specified by name (“black”, “blue”), a floating-point shade of gray (“0.75”), or an HTML hex string (“#ab45bc”). Certain common colors can also be specified by one-letter abbreviations. Supported colors are red (r), green (g), blue (b), cyan (c), yellow (y), magenta (m), black (k), and white (w).

Plots

Plots in Boomslang are constructed by adding plot elements to Plot objects. The Plot object's API is defined below.

class `boomslang.Plot.Plot`

Represents a single plot, usually consisting of a single X and Y axis.

add (*plottableObject*)

Add a plottable object (a Line, Bar, etc.) to this plot, causing it to be drawn when the plot is drawn.

addInset (*plot*, ***kwargs*)

Add *plot* (a Plot object) as an inset to this plot object.

Valid arguments are as follows:

Argument	Description
<code>width</code>	The width of the inset as a fraction of the size of the parent plot
<code>height</code>	The height of the inset as a fraction of the size of the parent plot
<code>location</code>	The location of the inset. See Locations for valid locations.
<code>padding</code>	The amount of padding between the edge of the parent plot and the inset as a fraction of the size of the parent plot

addLineColor (*color*)

Add a line color to the list of line colors through which the plot will cycle when drawing lines. If no line colors are specified, the line colors specified by the Line objects themselves will be used.

Note that, when drawing lines, all line styles for a given color are cycled through before another color is used.

addLineStyle (*style*)

Add a line style to the list of line styles through which the plot will cycle when drawing lines. If no line styles are specified, the plot will default to the line style specified in the Line objects themselves.

Note that, when drawing lines, all line styles for a given color are cycled through before another color is used.

addMarker (*marker*)

Add a marker style to the list of marker styles through which the plot will cycle when drawing lines. If no markers are specified, the markers specified by the Line objects themselves will be used.

Note that, when drawing lines, all line style/color combinations are cycled through with a given marker before a new marker is chosen.

axesLabelSize = None

The size of the text used for labeling the x and y axes.

drawPlot (*fig, ax*)

Used by PlotLayout to plot the graph at a given location in the layout.

getDimensions ()

Get the dimensions of this plot.

grid

A boomslang.Grid.Grid that defines the properties of this plot's grid lines. See [Grid Lines](#) for configuration options.

hasFigLegend (*columns=1, location='best', scatterPoints=3, draw_frame=True, bbox_to_anchor=None, labelSize=None, title=None, labelOrdering=None, leftToRight=False*)

Declare that the figure has a legend with a given number of columns and location.

hasLegend (*columns=1, location='best', scatterPoints=3, draw_frame=True, bbox_to_anchor=None, labelSize=None, title=None, labelOrdering=None, leftToRight=False*)

Declare that the plot has a legend with a given number of columns and location.

hideTicks = None

If True, this plot's tick labels are hidden.

legendLabelSize

The size of the text in this plot's legend.

logbase = None

The base of the logarithm used to draw log scale axes.

logbasex = None

The base of the logarithm used to draw the x-axis. Overrides *logbase*, and only takes effect if *logx* or *loglog* are True.

logbasey = None

The base of the logarithm used to draw the y-axis. Overrides *logbase*, and only takes effect if *logy* or *loglog* are True.

loglog = None

If true, both this plot's axes will be drawn in log scale.

logx = None

If true, this plot's x-axis will be drawn in log scale.

logy = None

If true, this plot's y-axis will be drawn in log scale.

plot ()

Draw this plot to a matplotlib canvas.

projection = None

Defines the projection used when drawing this plot. The only currently supported value other than the standard (no projection) is 'polar'.

save (*filename, **kwargs*)

Save this plot to a file.

setDimensions (*width=None, height=None, dpi=100*)

Set the dimensions for this plot to *width* x *height*

setPlotParameters (***kwdict*)

Set the margins of this plot. See [matplotlib's SubplotParams documentation](#) for more details. It is recommended that you set [boomslang.Plot.Plot.tight](#) to True instead of setting these parameters.

setTitleProperties (***propList*)

Set the properties of the title. See [Labels](#) for more information about valid properties.

setTwinX (*label, index, yMin=None, yMax=None*)

Make the plot use a secondary y-axis with the provided label. All registered plottable objects at or after the given index will use this second axis.

setXFormatter (*formatter*)

Set the x-axis formatter used by this plot to the given function.

setYFormatter (*formatter*)

Set the y-axis formatter used by this plot to the given function.

split (*pieces*)

Split this plot into *pieces* separate plots, each showing a different portion of the x-axis

tight = **None**

If True, this plot is auto-scaled so that axis labels don't get cut off.

title = **None**

The plot's title (will be centered above the axes themselves)

titleSize = **None**

The size of the text used for plot title.

xLabel = **None**

The label for the plot's x-axis

xLabelProperties

A dictionary of properties that control the appearance of the X axis' axis label. See [Labels](#) for more information on which properties can be set.

xLimits

A pair giving the minimum and maximum values visible on the x-axis.

xTickLabelSize = **None**

The size of the text used for x-axis tick labels

yLabel = **None**

The label for the plot's y-axis

yLabelProperties

A dictionary of properties that control the appearance of the Y axis' axis label. See [Labels](#) for more information on which properties can be set.

yLimits

A pair giving the minimum and maximum values visible on the x-axis.

yTickLabelSize = **None**

The size of the text used for y-axis tick labels

6.1 Grid Lines

`boomslang.Plot.Plot.grid` is a `boomslang.Grid.Grid` object that defines the appearance of the plot's grid lines. By default, there are no grid lines.

class `boomslang.Grid.Grid` (*color='#dddddd', style='-', width=0.5, visible=True*)

The Grid plot element defines a set of gridlines that are displayed behind all other plot elements

color = **None**

The color of the gridlines; see [Colors](#) for more information on available colors.

lineWidth = None

The width of gridlines.

style

The style of the grid's lines; see [Line Styles](#) for more information about available line styles.

visible = None

If True, the grid's lines are visible. If False, they are invisible.

which = None

If "major", the plot's major ticks have grid lines. If "minor", the plot's minor ticks have grid lines. If "both", both major and minor ticks have grid lines.

6.2 Locations

Parts of a plot like legends and insets allow the user to explicitly specify their location (by default, matplotlib tries to place the legend/inset where it is least obtrusive). Boomslang defines the standard matplotlib location strings along with a number of aliases that some find more intuitive than the defaults.

Valid location strings (clockwise from top middle, plus centered) are as follows:

- "upper center", "top middle", "upper middle"
- "upper right", "top right"
- "center right", "middle right", "right"
- "lower right", "bottom right"
- "lower center", "bottom middle", "bottom center"
- "lower left", "bottom left"
- "center left", "middle left", "left",
- "upper left", "top left"
- "center", "middle"

Layouts

Boomslang supports two different ways to lay out plots: `boomslang.PlotLayout.PlotLayout` and `boomslang.WeightedPlotLayout.WeightedPlotLayout`.

class `boomslang.PlotLayout.PlotLayout`

Displays multiple `boomslang.Plot.Plot` objects in one canvas in a grid, allowing the user to group related Plots in the same row.

addPlot (*plot*, *grouping=None*)

Add *plot* (a `boomslang.Plot.Plot`) to the layout. Optionally, specify a group name with *grouping*. Plots grouped in the same grouping are drawn on the same row of the grid.

groupOrder = `None`

If not `None`, a list of groups in order by the order in which they should be displayed

plot ()

Draw this layout to a matplotlib canvas.

save (*filename*, ***kwargs*)

Save this layout to a file.

setFigureDimensions (*x*, *y*, *dpi=100*)

Set the dimensions of the layout to be *x* by *y*. This overrides any values set with `boomslang.PlotLayout.PlotLayout.setPlotDimensions()`.

setPlotDimensions (*x*, *y*, *dpi=100*)

Set the dimensions of each plot in the layout to be *x* by *y*.

width

The width, in number of plots, of the layout.

class `boomslang.WeightedPlotLayout.WeightedPlotLayout`

A more sophisticated version of `boomslang.PlotLayout.PlotLayout` that allows some plots to be wider than others.

Like `PlotLayout`, `WeightedPlotLayout` allows plots to be grouped together into named groupings, with one grouping per row. However, it does not have a fixed width in number of plots to guide layout. Instead, the `_weights_` of the plots in a grouping determine how wide each plot in the grouping is. Any plot that has no grouping or is the only plot in its grouping will take up an entire row regardless of its weight.

For example, if there are two plots in a grouping and both plots have weight 1, they will each take up half the row. If one of the plots has weight 2 and the other has weight 1, the first plot will take up 2/3 of the row and the second will take up the remaining 1/3.

addPlot (*plot*, *grouping=None*, *weight=1*)

Add *plot* to the layout, optionally grouping it with all other plots added to the group *grouping*. *weight*

denotes the plot's weight within its grouping.

plot ()

Draw this layout to a matplotlib canvas.

save (*filename*, ***kwargs*)

Save this layout to a file.

Utility Functions

`boomslang.Uutils.cdf(values)`

Returns a *boomslang.Line.Line* representing the CDF of the list of values given in *values*.

`boomslang.Uutils.getBarsFromFile(filename, regex, postFunction=None, autofillXValues=False)`

Turns a regularly-structured file into a collection of *boomslang.Bar.Bar* objects.

For more details on arguments, see *getLinesFromFile()*.

Returns a list of *boomslang.Bar.Bar* objects.

`boomslang.Uutils.getLinesFromFile(filename, regex, postFunction=None, autofillXValues=False)`

Turn a regularly-structured file into a collection of *boomslang.Line.Line* objects.

Parses each line in *filename* using the regular expression *regex*. By default, the first matching group from the regular expression gives the x-axis value for a set of points and all subsequent matching groups give the y-axis values for each line. If *postFunction* is not None, it is a function that is applied to the matching groups before they are inserted into the lines. If *autofillXValues* is True, all matching groups are treated as y-axis values for lines and the x-axis value is the line number, indexed from 0.

Returns a list of *boomslang.Line.Line* objects.

Example: Suppose I had a file *blah.txt* that looked like this:

```
1980 - 1, 2, 3
1981 - 4, 5, 6
1982 - 7, 8, 9
```

The snippet below shows the result of running *boomslang.Uutils.getLinesFromFile()* on *blah.txt*:

```
>>> lines = boomslang.Uutils.getLinesFromFile("blah.txt", "(\\d+) - (\\d+), (\\d+), (\\d+)")
>>> len(lines)
3
>>> lines[0].xValues
[1980, 1981, 1982]
>>> lines[1].xValues
[1980, 1981, 1982]
>>> lines[2].xValues
[1980, 1981, 1982]
>>> lines[0].yValues
[1, 4, 7]
>>> lines[1].yValues
[2, 5, 8]
>>> lines[1].yValues
[3, 6, 9]
```

`boomslang.Utls.histogram(values, binSize)`

Returns a `boomslang.Line.Line` representing a histogram of the list of values given in *values* with bin size *binSize*.

Plotting from Django

This is a riff on <http://www.scipy.org/Cookbook/Matplotlib/Django>:

```
import django
from boomslang import *

def simple(request):
    plot = Plot()
    line = Line()
    line.yValues = [25, 40, 30, 23, 10, 50]
    line.xValues = range(len(line.yValues))

    plot.add(line)
    plot.setXLabel("X Label")
    plot.setYLabel("Y Label")
    plot.setYLimits(0,60)

    response=django.http.HttpResponse(content_type='image/png')

    plot.save(response)
    return response
```

This is getting a little off-topic, but multiple views of plots has produced strange results in mod-python, this is a dumb hack to plot in a separate process (replacing the `plot.save(response)` line above):

```
subproc = subprocess.Popen(['python', '-c',
    """
    from boomslang import *
    from cPickle import *
    import sys

    plot = load(sys.stdin)
    plot.save(sys.stdout)
    """],
    env={'PYTHONPATH':":.".join(sys.path),
        'MPLCONFIGDIR':os.environ.get('MPLCONFIGDIR','/tmp')},
    stdin=subprocess.PIPE,
    stdout=subprocess.PIPE,
    stderr=subprocess.PIPE)

cPickle.dump(multiplot, subproc.stdin)
png = subproc.communicate()[0]

print >>response, png
```


b

boomslang.Utils, [23](#)

A

add() (boomslang.ClusteredBars.ClusteredBars method), 12

add() (boomslang.Plot.Plot method), 17

add() (boomslang.StackedBars.StackedBars method), 13

addInset() (boomslang.Plot.Plot method), 17

addLine() (boomslang.StackedLines.StackedLines method), 13

addLineColor() (boomslang.Plot.Plot method), 17

addLineStyle() (boomslang.Plot.Plot method), 17

addMarker() (boomslang.Plot.Plot method), 17

addPlot() (boomslang.PlotLayout.PlotLayout method), 21

addPlot() (boomslang.WeightedPlotLayout.WeightedPlotLayout method), 21

align (boomslang.Bar.Bar attribute), 12

antialiased (boomslang.Line.Line attribute), 10

autosort (boomslang.PlotInfo.PlotInfo attribute), 9

axesLabelSize (boomslang.Plot.Plot attribute), 17

B

Bar (class in boomslang.Bar), 12

barWidth (boomslang.ClusteredBars.ClusteredBars attribute), 12

barWidth (boomslang.StackedBars.StackedBars attribute), 13

boomslang.Utills (module), 23

BoxAndWhisker (class in boomslang.BoxAndWhisker), 12

C

cdf() (in module boomslang.Utills), 23

ClusteredBars (class in boomslang.ClusteredBars), 12

color (boomslang.Bar.Bar attribute), 12

color (boomslang.Grid.Grid attribute), 19

color (boomslang.HLine.HLine attribute), 11

color (boomslang.Line.Line attribute), 10

color (boomslang.Scatter.Scatter attribute), 11

color (boomslang.VLine.VLine attribute), 11

D

dates (boomslang.Line.Line attribute), 10

deduplicateLegend (boomslang.ClusteredBars.ClusteredBars attribute), 12

draw() (boomslang.PlotInfo.PlotInfo method), 9

drawErrorBars() (boomslang.PlotInfo.PlotInfo method), 9

drawPlot() (boomslang.Plot.Plot method), 17

E

edgeColor (boomslang.Bar.Bar attribute), 12

G

getAttributes() (boomslang.PlotInfo.PlotInfo method), 9

getBarsFromFile() (in module boomslang.Utills), 23

getDimensions() (boomslang.Plot.Plot method), 18

getLinesFromFile() (in module boomslang.Utills), 23

grid (boomslang.Plot.Plot attribute), 18

Grid (class in boomslang.Grid), 19

groupOrder (boomslang.PlotLayout.PlotLayout attribute), 21

H

hasArrow() (boomslang.Label.Label method), 13

hasFigLegend() (boomslang.Plot.Plot method), 18

hasLegend() (boomslang.Plot.Plot method), 18

hatch (boomslang.Bar.Bar attribute), 12

hideTicks (boomslang.Plot.Plot attribute), 18

histogram() (in module boomslang.Utills), 23

HLine (class in boomslang.HLine), 11

L

label (boomslang.PlotInfo.PlotInfo attribute), 9

Label (class in boomslang.Label), 13

labelProperties (boomslang.Label.Label attribute), 13

legendLabelSize (boomslang.Plot.Plot attribute), 18

Line (class in boomslang.Line), 10

lineStyle (boomslang.HLine.HLine attribute), 11

lineStyle (boomslang.Line.Line attribute), 11

lineStyle (boomslang.VLine.VLine attribute), 11
 linewidth (boomslang.Bar.Bar attribute), 12
 lineWidth (boomslang.Grid.Grid attribute), 19
 lineWidth (boomslang.Line.Line attribute), 11
 logbase (boomslang.Plot.Plot attribute), 18
 logbasex (boomslang.Plot.Plot attribute), 18
 logbasey (boomslang.Plot.Plot attribute), 18
 loglog (boomslang.Plot.Plot attribute), 18
 logx (boomslang.Plot.Plot attribute), 18
 logy (boomslang.Plot.Plot attribute), 18

M

marker (boomslang.HLine.HLine attribute), 11
 marker (boomslang.Label.Label attribute), 13
 marker (boomslang.Line.Line attribute), 11
 marker (boomslang.Scatter.Scatter attribute), 11
 marker (boomslang.VLine.VLine attribute), 11
 markerSize (boomslang.HLine.HLine attribute), 11
 markerSize (boomslang.Line.Line attribute), 11
 markerSize (boomslang.Scatter.Scatter attribute), 11
 markerSize (boomslang.VLine.VLine attribute), 11

P

padding (boomslang.Bar.Bar attribute), 12
 padding (boomslang.ClusteredBars.ClusteredBars attribute), 12
 padding (boomslang.StackedBars.StackedBars attribute), 13
 Plot (class in boomslang.Plot), 17
 plot() (boomslang.Plot.Plot method), 18
 plot() (boomslang.PlotLayout.PlotLayout method), 21
 plot() (boomslang.WeightedPlotLayout.WeightedPlotLayout method), 22
 PlotInfo (class in boomslang.PlotInfo), 9
 PlotLayout (class in boomslang.PlotLayout), 21
 plotType (boomslang.PlotInfo.PlotInfo attribute), 9
 projection (boomslang.Plot.Plot attribute), 18

S

save() (boomslang.Plot.Plot method), 18
 save() (boomslang.PlotLayout.PlotLayout method), 21
 save() (boomslang.WeightedPlotLayout.WeightedPlotLayout method), 22
 Scatter (class in boomslang.Scatter), 11
 setDimensions() (boomslang.Plot.Plot method), 18
 setFigureDimensions() (boomslang.PlotLayout.PlotLayout method), 21
 setPlotDimensions() (boomslang.PlotLayout.PlotLayout method), 21
 setPlotParameters() (boomslang.Plot.Plot method), 18
 setTitleProperties() (boomslang.Plot.Plot method), 18
 setTwinX() (boomslang.Plot.Plot method), 19
 setXFormatter() (boomslang.Plot.Plot method), 19
 setYFormatter() (boomslang.Plot.Plot method), 19

spacing (boomslang.ClusteredBars.ClusteredBars attribute), 12
 spacing (boomslang.StackedBars.StackedBars attribute), 13
 split() (boomslang.Plot.Plot method), 19
 split() (boomslang.PlotInfo.PlotInfo method), 10
 StackedBars (class in boomslang.StackedBars), 13
 StackedLines (class in boomslang.StackedLines), 13
 steps (boomslang.Line.Line attribute), 11
 style (boomslang.Grid.Grid attribute), 20

T

text (boomslang.Label.Label attribute), 13
 tight (boomslang.Plot.Plot attribute), 19
 title (boomslang.Plot.Plot attribute), 19
 titleSize (boomslang.Plot.Plot attribute), 19

V

visible (boomslang.Grid.Grid attribute), 20
 VLine (class in boomslang.VLine), 11

W

WeightedPlotLayout (class in boomslang.WeightedPlotLayout), 21
 which (boomslang.Grid.Grid attribute), 20
 width (boomslang.Bar.Bar attribute), 12
 width (boomslang.BoxAndWhisker.BoxAndWhisker attribute), 12
 width (boomslang.ClusteredBars.ClusteredBars attribute), 13
 width (boomslang.HLine.HLine attribute), 11
 width (boomslang.PlotLayout.PlotLayout attribute), 21
 width (boomslang.VLine.VLine attribute), 11

X

x (boomslang.Label.Label attribute), 13
 xLabel (boomslang.Plot.Plot attribute), 19
 xLabelProperties (boomslang.Plot.Plot attribute), 19
 xLimits (boomslang.Plot.Plot attribute), 19
 xSequence (boomslang.BoxAndWhisker.BoxAndWhisker attribute), 12
 xTickLabelPoints (boomslang.PlotInfo.PlotInfo attribute), 10
 xTickLabelProperties (boomslang.PlotInfo.PlotInfo attribute), 10
 xTickLabels (boomslang.PlotInfo.PlotInfo attribute), 10
 xTickLabelSize (boomslang.Plot.Plot attribute), 19
 xValues (boomslang.PlotInfo.PlotInfo attribute), 10

Y

y (boomslang.Label.Label attribute), 13
 yErrors (boomslang.PlotInfo.PlotInfo attribute), 10
 yLabel (boomslang.Plot.Plot attribute), 19

`yLabelProperties` (boomslang.Plot.Plot attribute), [19](#)
`yLimits` (boomslang.Plot.Plot attribute), [19](#)
`yMaxes` (boomslang.PlotInfo.PlotInfo attribute), [10](#)
`yMins` (boomslang.PlotInfo.PlotInfo attribute), [10](#)
`yTickLabelPoints` (boomslang.PlotInfo.PlotInfo attribute), [10](#)
`yTickLabelProperties` (boomslang.PlotInfo.PlotInfo attribute), [10](#)
`yTickLabels` (boomslang.PlotInfo.PlotInfo attribute), [10](#)
`yTickLabelSize` (boomslang.Plot.Plot attribute), [19](#)
`yValues` (boomslang.PlotInfo.PlotInfo attribute), [10](#)