MatPlotTheme Documentation

Release 0.1.2

James Yu

July 31, 2014

Contents

MatPlotTheme is a theming library for MatPlotLib. Greatly inspired by prettyplotlib, MatPlotTheme aims to provide easy-to-use APIs for creating proper and attractive data visualizations.

In MatPlotTheme, theming MatPlotLib figures is controlled by `style` and `palette`, which defines how elements are customized and which colors are used, respectively. As MatPlotTheme provides multiple styles/palettes (at least that is what I am working on), using the library is as simple as picking a style-palette combination and plot. What's more, MatPlotTheme inherits MatPlotLib's API configuration, which means existing code can be migrated with minimal effort.

CHAPTER 1

Contents

1.1 Overview

MatPlotTheme is a theming library for MatPlotLib. Greatly inspired by prettyplotlib, MatPlotTheme aims to provide easy-to-use APIs for creating proper and attractive data visualizations.

In MatPlotTheme, theming MatPlotLib figures is controlled by `style` and `palette`, which defines how elements are customized and which colors are used, respectively. As MatPlotTheme provides multiple styles/palettes, using the library is as simple as picking a style-palette combination and plot. What's more, MatPlotTheme inherits MatPlotLib's API configuration, which means existing code can be migrated with minimal effort.

1.1.1 Usage

MatPlotTheme provides a default `Style` and a default `Palette`. Each of them are python classes and all other styles/palettes are derived classes of them. `matplottheme` provides interfaces to all plotting methods in `Style`, which enable library usage like `matplottheme.plot(ax, x, y)`.

```
# Use API provided by matplottheme module
import matplottheme as mpt
import matplotlib.pylab as plt
import numpy as np

x = np.arange(1000)
y = np.random.normal(size=1000).cumsum()

fig = plt.figure()
ax = fig.add_subplot(111)
# MatPlotTheme plots a line using ggplot2 style/palette
mpt.set_theme('ggplot2', 'ggplot2')
mpt.plot(ax, x, y)
```

This code block can also generate the same plot as the first one.

```
# Use style/palette objects
from matplottheme.style.ggplot2 import ggplot2Style
from matplottheme.palette.ggplot2 import ggplot2Palette
import matplotlib.pylab as plt
import numpy as np

x = np.arange(1000)
y = np.random.normal(size=1000).cumsum()
```

```
fig = plt.figure()
ax = fig.add_subplot(111)
# Manually using ggplot2 style/palette
ggplot2Style(ggplot2Palette()).plot(ax, x, y)
```

1.1.2 Dependency

- MatPlotLib. `pip install matplotlib` is the most simple installation method.

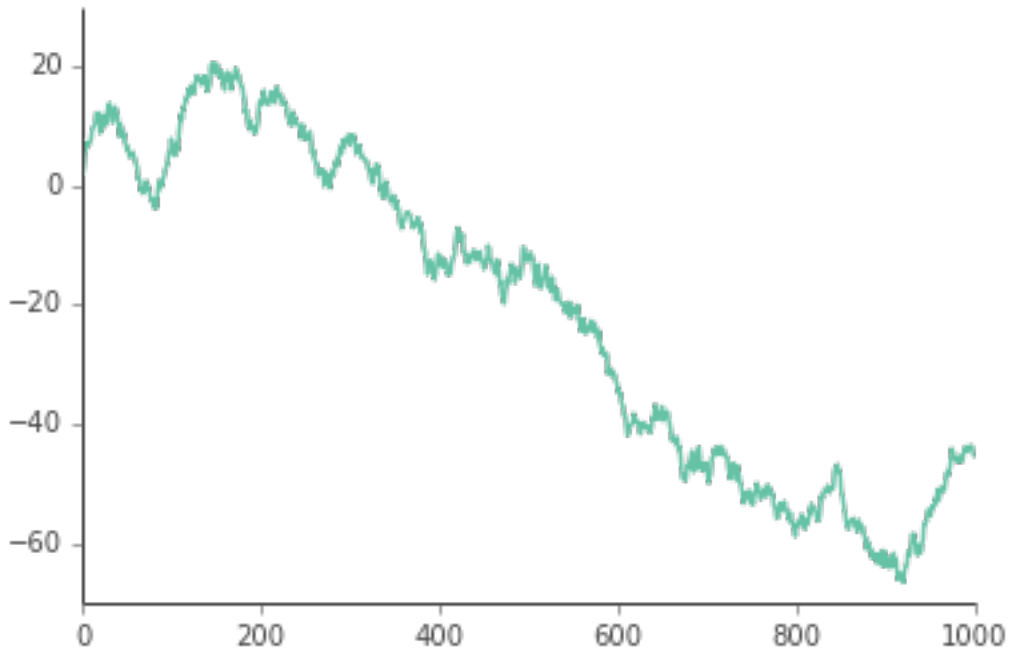
1.1.3 License

1.2 Gallery

Example plots with MatPlotTheme.

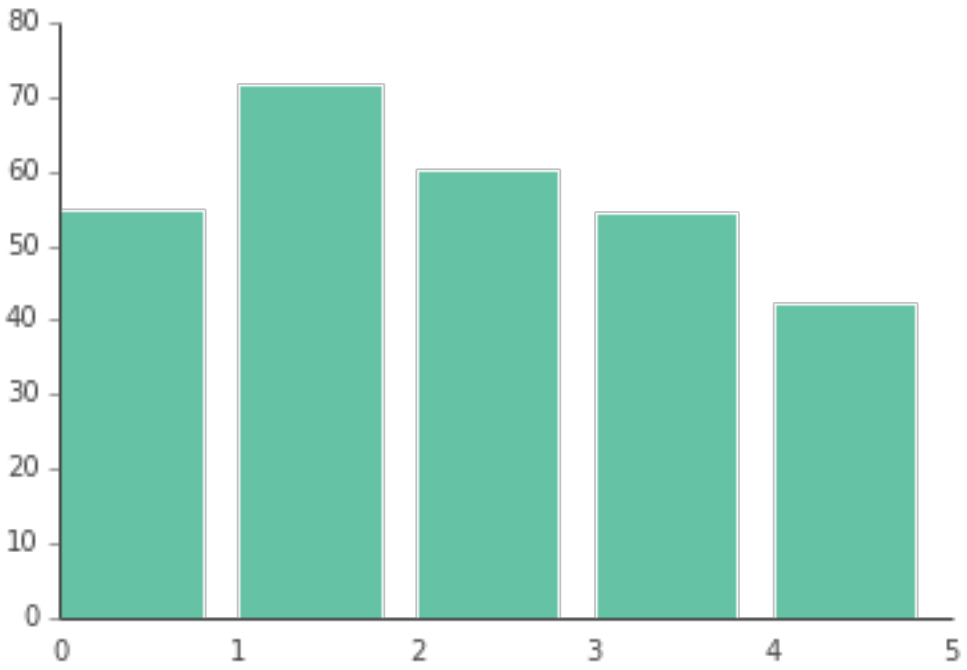
1.2.1 Plots

Line Plot



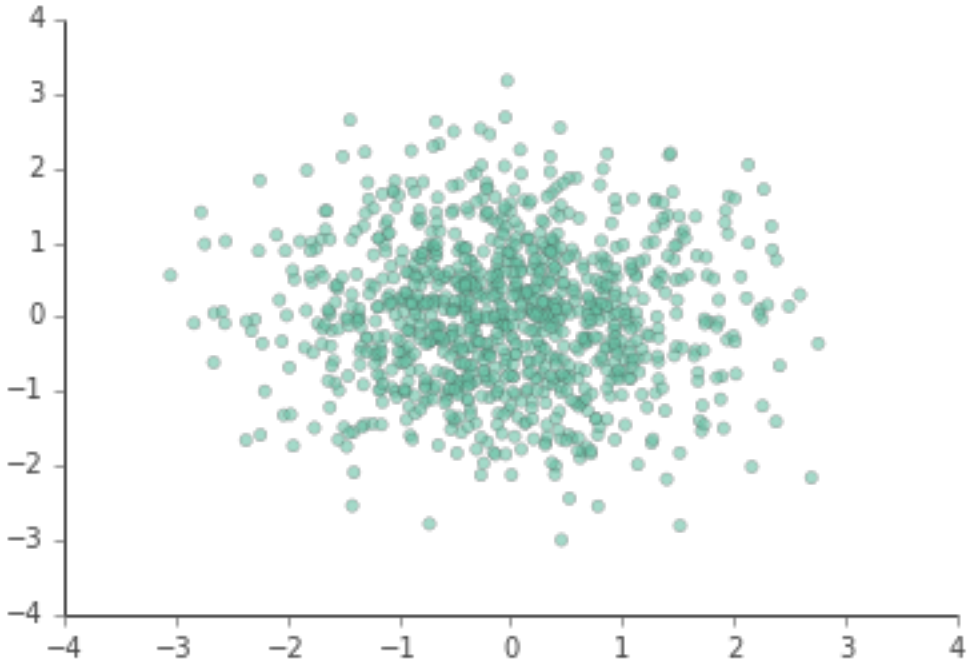

```
import numpy as np
import matplotlib.pylab as plt
import matplottheme as mpt

np.random.seed(0)

y = np.random.normal(size=1000).cumsum()

fig, ax = plt.subplots()

mpt.plot(ax, np.arange(1000), y)

plt.show()
```

Bar Plot



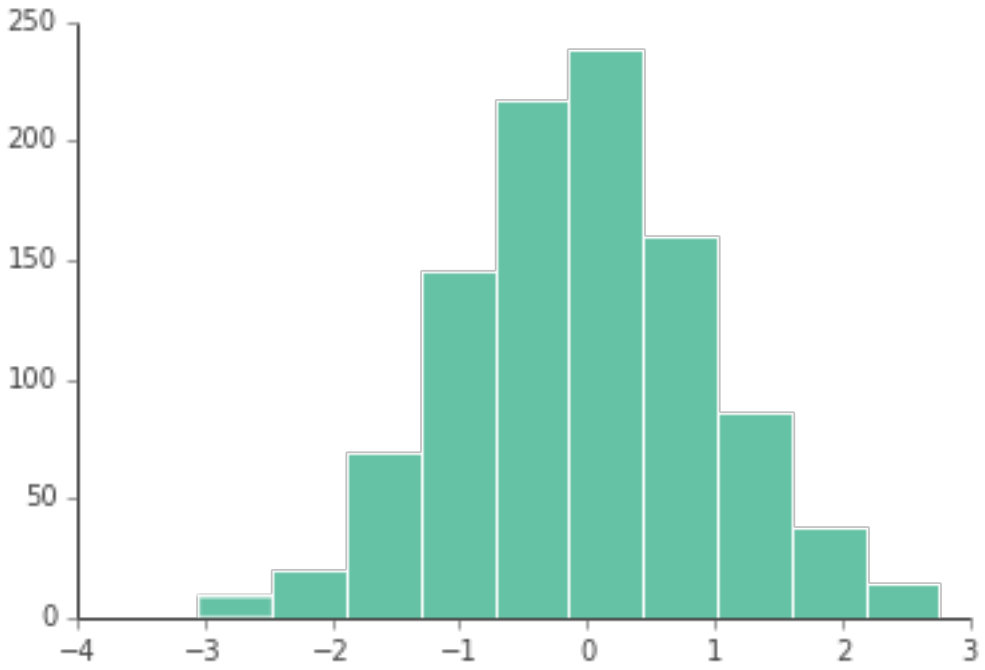

```
import numpy as np
import matplotlib.pylab as plt
import matplottheme as mpt

np.random.seed(0)

x = np.random.rand(5) * 100

fig, ax = plt.subplots()

mpt.bar(ax, np.arange(5), x)

plt.show()
```

Scatter Plot



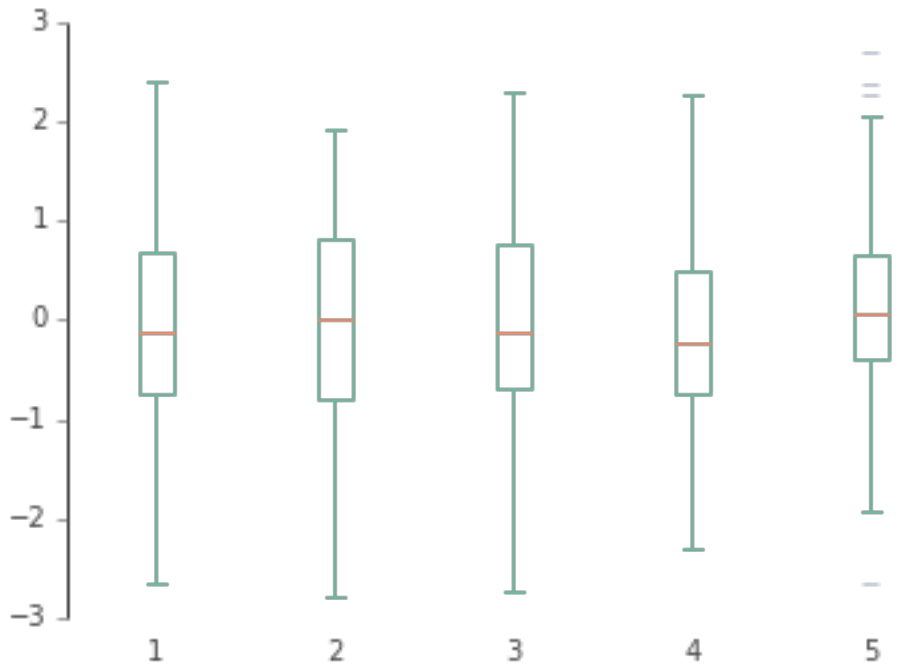

```
import numpy as np
import matplotlib.pylab as plt
import matplottheme as mpt

np.random.seed(0)

x = np.random.normal(size = 1000)
y = np.random.normal(size = 1000)

fig, ax = plt.subplots()

mpt.scatter(ax, x, y)

plt.show()
```

Histogram Plot

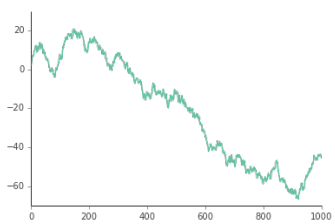


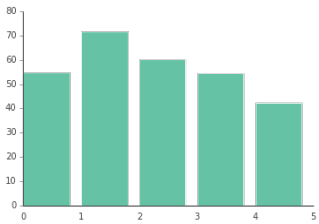

```
import numpy as np
import matplotlib.pylab as plt
import matplottheme as mpt

x = np.random.normal(size=1000)

fig, ax = plt.subplots()

mpt.hist(ax, x)

plt.show()
```

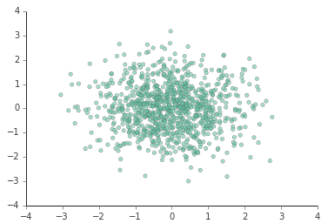
Box Plot

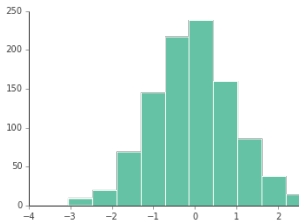



```
import numpy as np
import matplotlib.pylab as plt
import matplottheme as mpt

np.random.seed(0)

x = np.random.normal(size=(100,5))

fig, ax = plt.subplots()

mpt.boxplot(ax, x)

plt.show()
```









1.3 Library Reference

Description of the functions, classes and modules contained within MatPlotTheme.

1.3.1 MatPlotTheme

matplottheme is the starting point of MatPlotTheme library. It wraps the instances of Style and Palette, and provides plotting interfaces to the users.

matplottheme.**bar**(*ax*, **args*, ***kwargs*)

Add a bar plot to the input matplotlib.axes.Axes object.

This method is a wrapper of the bar method in the Style object which is used for stylization. All parameters are directly handed over to the wrapped bar method.

Parameters **ax** – The input axes object.

Returns matplotlib.patches.Rectangle instances.

All additional input parameters are passed to the wrapped bar method.

See also:

matplottheme.style.default.Style.bar()

Note: Different style may introduce different input parameters besides those from matplotlib.axes.Axes.bar().

---

matplottheme.**barh**(*ax*, **args*, ***kwargs*)

Add a horizontal bar plot to the input matplotlib.axes.Axes object.

This method is a wrapper of the barh method in the Style object which is used for stylization. All parameters are directly handed over to the wrapped barh method.

Parameters **ax** – The input axes object.

Returns matplotlib.patches.Rectangle instances.

All additional input parameters are passed to the wrapped barh method.

See also:

matplottheme.style.default.Style.barh()

Note: Different style may introduce different input parameters besides those from matplotlib.axes.Axes.barh().

---

matplottheme.**boxplot**(*ax*, **args*, ***kwargs*)

Add a box plot to the input matplotlib.axes.Axes object.

This method is a wrapper of the boxplot method in the Style object which is used for stylization. All parameters are directly handed over to the wrapped boxplot method.

Parameters **ax** – The input axes object.

Returns A dictionary. See boxplot().

All additional input parameters are passed to the wrapped boxplot method.

See also:

matplottheme.style.default.Style.boxplot()

Note: Different style may introduce different input parameters besides those from matplotlib.axes.Axes.boxplot().

---

matplottheme.**cohere**(*ax*, **args*, ***kwargs*)

Add a coherence plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of the `cohere` method in the `Style` object which is used for stylization. All parameters are directly handed over to the wrapped `cohere` method.

Parameters **ax** – The input axes object.

Returns A tuple (Cxy, f), where f are the frequencies of the coherence vector.

All additional input parameters are passed to the wrapped `cohere` method.

See also:

`matplottheme.style.default.Style.cohere()`

Note: Different style may introduce different input parameters besides those from `matplotlib.axes.Axes.cohere()`.

---

matplottheme.**csd**(*ax*, **args*, ***kwargs*)

Add a cross-spectral density plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of the `csd` method in the `Style` object which is used for stylization. All parameters are directly handed over to the wrapped `csd` method.

Parameters **ax** – The input axes object.

Returns A tuple (Pxy, freqs). P is the cross spectrum (complex valued).

All additional input parameters are passed to the wrapped `csd` method.

See also:

`matplottheme.style.default.Style.csd()`

Note: Different style may introduce different input parameters besides those from `matplotlib.axes.Axes.csd()`.

---

matplottheme.**errorbar**(*ax*, **args*, ***kwargs*)

Add an errorbar plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of the `errorbar` method in the `Style` object which is used for stylization. All parameters are directly handed over to the wrapped `errorbar` method.

Parameters **ax** – The input axes object.

Returns A tuple (plotline, caplines, barlinecols).

All additional input parameters are passed to the wrapped `errorbar` method.

See also:

`matplottheme.style.default.Style.errorbar()`

Note: Different style may introduce different input parameters besides those from `matplotlib.axes.Axes.errorbar()`.

---

matplottheme.**fill_between**(*ax*, **args*, ***kwargs*)

Add filled polygons to `matplotlib.axes.Axes` object.

This method is a wrapper of the `fill_between` method in the `Style` object which is used for stylization. All parameters are directly handed over to the wrapped `fill_between` method.

Parameters **ax** – The input axes object.

All additional input parameters are passed to the wrapped `fill_between` method.

See also:

`matplottheme.style.default.Style.fill_between()`

Note: Different style may introduce different input parameters besides those from `matplotlib.axes.Axes.fill_between()`.

`matplottheme.`**`fill_betweenx`**(*ax*, **args*, ***kwargs*)

Add filled polygons to `matplotlib.axes.Axes` object.

This method is a wrapper of the `fill_betweenx` method in the `Style` object which is used for stylization. All parameters are directly handed over to the wrapped `fill_betweenx` method.

Parameters **ax** – The input axes object.

All additional input parameters are passed to the wrapped `fill_betweenx` method.

See also:

`matplottheme.style.default.Style.fill_betweenx()`

Note: Different style may introduce different input parameters besides those from `matplotlib.axes.Axes.fill_betweenx()`.

`matplottheme.`**`hist`**(*ax*, **args*, ***kwargs*)

Add a histogram plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of the `hist` method in the `Style` object which is used for stylization. All parameters are directly handed over to the wrapped `histogram` method.

Parameters **ax** – The input axes object.

Returns (n, bins, patches) or ([n0, n1, ...], bins, [patches0, patches1,...])

All additional input parameters are passed to the wrapped `hist` method.

See also:

`matplottheme.style.default.Style.hist()`

Note: Different style may introduce different input parameters besides those from `matplotlib.axes.Axes.hist()`.

`matplottheme.`**`legend`**(*ax*, **args*, ***kwargs*)

Place a legend to the input `matplotlib.axes.Axes` object.

This method is a wrapper of the `legend` method in the `Style` object which is used for stylization. All parameters are directly handed over to the wrapped `legend` method.

Parameters **ax** – The input axes object.

Returns The legend

All additional input parameters are passed to the wrapped `legend` method.

See also:

`matplottheme.style.default.Style.legend()`

Note: Different style may introduce different input parameters besides those from `matplotlib.legend.Legend`.

`matplottheme.`**`pcolormesh`**(*ax*, **args*, ***kwargs*)

Add a quadrilateral mesh to `matplotlib.axes.Axes` object.

This method is a wrapper of the `pcolormesh` method in the `Style` object which is used for stylization. All parameters are directly handed over to the wrapped `pcolormesh` method.

Parameters **ax** – The input axes object.

All additional input parameters are passed to the wrapped `pcolormesh` method.

See also:

`matplottheme.style.default.Style.pcolormesh()`

Note: Different style may introduce different input parameters besides those from `matplotlib.axes.Axes.pcolormesh()`.

`matplottheme.`**`plot`**(*ax*, **args*, ***kwargs*)

Add a line plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of the `plot` method in the `Style` object which is used for stylization. All parameters are directly handed over to the wrapped `plot` method.

Parameters **ax** – The input axes object.

Returns A list of lines that were added.

All additional input parameters are passed to the wrapped `plot` method.

See also:

`matplottheme.style.default.Style.plot()`

Note: Different style may introduce different input parameters besides those from `matplotlib.axes.Axes.plot()`.

`matplottheme.`**`psd`**(*ax*, **args*, ***kwargs*)

Add a power spectral density plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of the `psd` method in the `Style` object which is used for stylization. All parameters are directly handed over to the wrapped `psd` method.

Parameters **ax** – The input axes object.

Returns A tuple (Pxx, freqs).

All additional input parameters are passed to the wrapped `psd` method.

See also:

`matplottheme.style.default.Style.psd()`

Note: Different style may introduce different input parameters besides those from `matplotlib.axes.Axes.psd()`.

`matplottheme.`**`scatter`**(*ax*, **args*, ***kwargs*)

Add a scatter plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of the `scatter` method in the `Style` object which is used for stylization. All parameters are directly handed over to the wrapped `scatter` method.

Parameters **ax** – The input axes object.

Returns `matplotlib.collections.PathCollection` objects.

All additional input parameters are passed to the wrapped `scatter` method.

See also:

`matplottheme.style.default.Style.scatter()`

Note: Different style may introduce different input parameters besides those from `matplotlib.axes.Axes.scatter()`.

`matplottheme.`**`set_theme`**(*style_name=None*, *palette_name=None*)

Set the global `Style` and `Palette`.

This method sets the input `Style` and `Palette` as default customization options. All plotting methods in `matplottheme` employ these options.

Parameters

- **palette_name** – The name of a `Palette`. Input value can be None, `'default'`, or `'ggplot2'`.
- **style_name** – The name of a `Style`. Input value can be None, `'default'`, or `'ggplot2'`.

1.3.2 Style

`style` is the collection of all available `Style` provided by MatPlotTheme. All style classes are derived classes of `Style`.

Default Style

class `matplottheme.style.default.`**`Style`**(*palette*)

This class is a collection of all painting methods provided by the default style of MatPlotTheme.

Parameters **palette** – The palette used for coloring.

`bar`(*ax*, *position*, *length*, *width=0.8*, *offset=None*, **args*, ***kwargs*)

Add a bar plot to the input `matplotlib.axes.Axes` object.

Parameters

- **ax** – The input axes object.
- **position** – The position of each bar. Equivalent to `left` parameter of `matplotlib.axes.Axes.bar()` when `orientation` is `vertical`, or `bottom` when `horizontal`.
- **length** – The length of each bar. Equivalent to `height` parameter of `matplotlib.axes.Axes.bar()` when `orientation` is `vertical`, or `width` when `horizontal`.
- **width** – The width of each bar. Equivalent to `width` parameter of `matplotlib.axes.Axes.bar()` when `orientation` is `vertical`, or `height` when `horizontal`.

- **offset** – The start offset of each bar. Equivalent to `bottom` parameter of `matplotlib.axes.Axes.bar()` when `orientation` is `vertical`, or `left` when `horizontal`.
- **grid** – Add grid lines perpendicular to the bar orientation. Default is `None`. Value can be `None`, `'x'`, `'y'`, `'both'`, or `'auto'`.
- **ticks** – Remove the default positional labels and add custom tick labels. Default is `None`.
- **annotations** – Add annotations to each bar. Default is `None`.
- **annotations_loc** – Control the position of annotations. Default is `'out'`. Value can be `'out'`, `'in'`, and `'center'`.
- **annotations_margin** – Control the margin size between annotations and bars. The value is the portion of plot size. Default is `0.025`.
- **reset_color_cycle** – Reset the color cycle iterator of bars. Default is `False`.

Returns `matplotlib.patches.Rectangle` instances.

Parameters `position`, `length`, `width`, and `offset` corresponds to the first four parameters of `matplotlib.axes.Axes.bar()` and `matplotlib.axes.Axes.barh()`.

A major modification made on the bar plot is the change of color cycle, which is used to color different bars. `matplotlib.axes.Axes` uses blue as default bar color. MatPlotTheme add a color cycle, which is control by the `Palette` employed. `reset_color_cycle` can reset the iterable and the color for current bar will reset to the start of the cycle.

All additional input parameters are passed to `bar()`.

See also:

`matplotlib.axes.bar()` for documentation on valid kwargs.

barh (*ax*, *position*, *length*, *width=0.8*, *offset=None*, **args*, ***kwargs*)

Add a horizontal bar plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of `self.bar()` method. The parameter `orientation` is set to `'horizontal'` and all other parameters are passed to `self.bar()`.

boxplot (*ax*, *x*, **args*, ***kwargs*)

Add a box plot to the input `matplotlib.axes.Axes` object.

Parameters

- **ax** – The input axes object.
- **x** – Input data.
- **grid** – Add grid lines perpendicular to the bar orientation. Default is `None`. Value can be `None`, `'x'`, `'y'`, `'both'`, or `'auto'`.
- **ticks** – Remove the default positional labels and add custom tick labels. Default is `None`.

Returns A dictionary. See `boxplot()`.

All additional input parameters are passed to `boxplot()`.

See also:

`matplotlib.axes.boxplot()` for documentation on valid kwargs.

cohere (*ax*, *x*, *y*, **args*, ***kwargs*)

Add a coherence plot to the input `matplotlib.axes.Axes` object.

Parameters

- **ax** – The input axes object.
- **x** – Input x-data.
- **y** – Input y-data.
- **grid** – Add grid lines to the plot. Default is `None`. Value can be `None`, `'x'`, `'y'`, or `'both'`.
- **reset_color_cycle** – Reset the color cycle iterator of lines. Default is `False`.

Returns A tuple (Cxy, f), where f are the frequencies of the coherence vector.

A major modification made on the coherence plot is the change of color cycle, which is used to color different lines. `matplotlib.axes.Axes` uses an iterable cycle to generate colors for different lines. The color cycle is changed by the `Palette` employed. `reset_color_cycle` can reset the iterable and the color for current line will reset to the start of the cycle.

All additional input parameters are passed to `cohere()`.

See also:

`matplotlib.axes.cohere()` for documentation on valid kwargs.

csd(*ax*, *x*, *y*, **args*, ***kwargs*)

Add a cross-spectral density plot to the input `matplotlib.axes.Axes` object.

Parameters

- **ax** – The input axes object.
- **x** – Input x-data.
- **y** – Input y-data.
- **grid** – Add grid lines to the plot. Default is `None`. Value can be `None`, `'x'`, `'y'`, or `'both'`.
- **reset_color_cycle** – Reset the color cycle iterator of lines. Default is `False`.

Returns A tuple (Pxy, freqs). P is the cross spectrum (complex valued).

A major modification made on the cross-spectral density plot is the change of color cycle, which is used to color different lines. `matplotlib.axes.Axes` uses an iterable cycle to generate colors for different lines. The color cycle is changed by the `Palette` employed. `reset_color_cycle` can reset the iterable and the color for current line will reset to the start of the cycle.

All additional input parameters are passed to `csd()`.

See also:

`matplotlib.axes.csd()` for documentation on valid kwargs.

errorbar(*ax*, *x*, *y*, **args*, ***kwargs*)

Add an errorbar plot to the input `matplotlib.axes.Axes` object.

Parameters

- **ax** – The input axes object.
- **x** – Input x-data.
- **y** – Input y-data.
- **grid** – Add grid lines to the plot. Default is `None`. Value can be `None`, `'x'`, `'y'`, or `'both'`.
- **reset_color_cycle** – Reset the color cycle iterator of lines. Default is `False`.

Returns A tuple (plotline, caplines, barlinecols).

A major modification made on the errorbar plot is the change of color cycle, which is used to color different lines. `matplotlib.axes.Axes` uses an iterable cycle to generate colors for different lines. The color cycle is changed by the `Palette` employed. `reset_color_cycle` can reset the iterable and the color for current line will reset to the start of the cycle.

All additional input parameters are passed to `errorbar()`.

See also:

`matplotlib.axes.errorbar()` for documentation on valid kwargs.

fill_between(*ax*, *x*, *y1*, **args*, ***kwargs*)
Add filled polygons to `matplotlib.axes.Axes` object.

Parameters

- **ax** – The input axes object.
- **x** – Input x-data.
- **y1** – Input y-data.
- **grid** – Add grid lines to the plot. Default is `None`. Value can be `None`, `'x'`, `'y'`, or `'both'`.
- **reset_color_cycle** – Reset the color cycle iterator of lines. Default is `False`.

A major modification made on the filled polygons is the change of color cycle, which is used to color different lines. `matplotlib.axes.Axes` uses an iterable cycle to generate colors for different lines. The color cycle is changed by the `Palette` employed. `reset_color_cycle` can reset the iterable and the color for current line will reset to the start of the cycle.

All additional input parameters are passed to `fill_between()`.

See also:

`matplotlib.axes.fill_between()` for documentation on valid kwargs.

fill_betweenx(*ax*, *y*, *x1*, **args*, ***kwargs*)
Add filled polygons to `matplotlib.axes.Axes` object.

Parameters

- **ax** – The input axes object.
- **y** – Input y-data.
- **x1** – Input x-data.
- **grid** – Add grid lines to the plot. Default is `None`. Value can be `None`, `'x'`, `'y'`, or `'both'`.
- **reset_color_cycle** – Reset the color cycle iterator of lines. Default is `False`.

Returns A tuple (plotline, caplines, barlinecols).

A major modification made on the filled polygons is the change of color cycle, which is used to color different lines. `matplotlib.axes.Axes` uses an iterable cycle to generate colors for different lines. The color cycle is changed by the `Palette` employed. `reset_color_cycle` can reset the iterable and the color for current line will reset to the start of the cycle.

All additional input parameters are passed to `fill_betweenx()`.

See also:

matplotlib.axes.fill_betweenx() for documentation on valid kwargs.

hist(*ax*, *x*, **args*, ***kwargs*)

Add a histogram plot to the input matplotlib.axes.Axes object.

Parameters

- **ax** – The input axes object.
- **x** – Input data.
- **grid** – Add grid lines perpendicular to the bar orientation. Default is None. Value can be None, 'x', 'y', 'both', or 'auto'.
- **reset_color_cycle** – Reset the color cycle iterator of bars. Default is False.

Returns (n, bins, patches) or ([n0, n1, ...], bins, [patches0, patches1,...])

A major modification made on the histogram plot is the change of color cycle, which is used to color different bars. matplotlib.axes.Axes uses an iterable cycle to generate colors for different lines. The color cycle is changed by the Palette employed. reset_color_cycle can reset the iterable and the color for current bar will reset to the start of the cycle.

All additional input parameters are passed to hist().

See also:

matplotlib.axes.hist() for documentation on valid kwargs.

legend(*ax*, **args*, ***kwargs*)

Place a legend to the input matplotlib.axes.Axes object.

Parameters

- **ax** – The input axes object.
- **legend_alpha** – The opacity of background rectangle of the legend. Default is 0.8.

Returns The legend

All additional input parameters are passed to legend().

See also:

matplotlib.axes.legend() for documentation on valid kwargs.

pcolormesh(*ax*, **args*, ***kwargs*)

Add a quadrilateral mesh to matplotlib.axes.Axes object.

Parameters

- **ax** – The input axes object.
- **color** – Use input color for meshing. Default is 'auto'. Value can be 'auto', all, 'cold', and warm.
- **colorbar** – Draw a color bar. Default is 'vertical'. Value can be 'vertical', 'horizontal', and None.
- **xticks** – Remove the default positional labels and add custom x-axis tick labels. Default is None.
- **yticks** – Remove the default positional labels and add custom y-axis tick labels. Default is None.

Returns A (matplotlib.colorbar.Colorbar, matplotlib.collections.QuadMesh) tuple.

A major modification made on the quadrilateral mesh is the change of color map. In each palette at least three color maps are defined: cold, warm, and cold-warm. If the value for parameter `color` is set to `cold`, `warm`, or `all`, corresponding color map will be used. If the value is `all`, this method will check the data values and deside which color map to use. Cold color map is used when the maximum value for all input data is smaller than zero. Warm color map is used when the minimum value is larger than zero. Otherwise the cold-warm color map is used.

All additional input parameters are passed to `pcolormesh()`.

See also:

`matplotlib.axes.pcolormesh()` for documentation on valid kwargs.

plot(*ax*, **args*, ***kwargs*)

Add a line plot to the input `matplotlib.axes.Axes` object.

Parameters

- **ax** – The input axes object.
- **grid** – Add grid lines to the plot. Default is `None`. Value can be `None`, `'x'`, `'y'`, or `'both'`.
- **reset_color_cycle** – Reset the color cycle iterator of lines. Default is `False`.

Returns A list of lines that were added.

A major modification made on the line plot is the change of color cycle, which is used to color different lines. `matplotlib.axes.Axes` uses an iterable cycle to generate colors for different lines. The color cycle is changed by the `Palette` employed. `reset_color_cycle` can reset the iterable and the color for current line will reset to the start of the cycle.

All additional input parameters are passed to `plot()`.

See also:

`matplotlib.axes.plot()` for documentation on valid kwargs.

psd(*ax*, *x*, **args*, ***kwargs*)

Add a power spectral density plot to the input `matplotlib.axes.Axes` object.

Parameters

- **ax** – The input axes object.
- **x** – Input x-data.
- **grid** – Add grid lines to the plot. Default is `None`. Value can be `None`, `'x'`, `'y'`, or `'both'`.
- **reset_color_cycle** – Reset the color cycle iterator of lines. Default is `False`.

Returns A tuple (Pxy, freqs). P is the cross spectrum (complex valued).

A major modification made on the power spectral density plot is the change of color cycle, which is used to color different lines. `matplotlib.axes.Axes` uses an iterable cycle to generate colors for different lines. The color cycle is changed by the `Palette` employed. `reset_color_cycle` can reset the iterable and the color for current line will reset to the start of the cycle.

All additional input parameters are passed to `psd()`.

See also:

`matplotlib.axes.psd()` for documentation on valid kwargs.

scatter(*ax*, *x*, *y*, **args*, ***kwargs*)
Add a scatter plot to the input `matplotlib.axes.Axes` object.

Parameters

- **ax** – The input axes object.
- **x** – Input x-data.
- **y** – Input y-data.
- **grid** – Add grid lines to the plot. Default is `None`. Value can be `None`, `'x'`, `'y'`, or `'both'`.
- **reset_color_cycle** – Reset the color cycle iterator of lines. Default is `False`.

Returns `matplotlib.collections.PathCollection` objects.

A major modification made on the scatter plot is the change of color cycle, which is used to color different bars. `matplotlib.axes.Axes` uses blue as default bar color. MatPlotTheme add a color cycle, which is control by the `Palette` employed. `reset_color_cycle` can reset the iterable and the color for current bar will reset to the start of the cycle.

All additional input parameters are passed to `scatter()`.

See also:

`matplotlib.axes.scatter()` for documentation on valid kwargs.

set_palette(*palette*)
Set the palette used for coloring.

Parameters **palette** – The palette used for coloring.

ggplot2 Style

class `matplottheme.style.ggplot2.`**ggplot2Style**(*palette*)
Bases: `matplottheme.style.default.Style`

This class is a collection of all painting methods provided by the ggplot2 style of MatPlotTheme.

Parameters **palette** – The palette used for coloring.

bar(*ax*, *position*, *length*, *width=0.8*, *offset=None*, **args*, ***kwargs*)
Add a bar plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of `bar()` with modifications on the design.

Notable modification on input argument is

- `grid` is set to `'auto'` by default.

See also:

`bar()` for documentation on valid kwargs.

barh(*ax*, *position*, *length*, *width=0.8*, *offset=None*, **args*, ***kwargs*)
Add a horizontal bar plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of `self.bar()` method. The parameter `orientation` is set to `'horizontal'` and all other parameters are passed to `self.bar()`.

boxplot(*ax*, *x*, **args*, ***kwargs*)
Add a box plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of `boxplot()` with modifications on the design.

Notable modification on input argument is

- `grid` is set to `'auto'` by default.

See also:

`boxplot()` for documentation on valid kwargs.

cohere(*ax*, *x*, *y*, **args*, ***kwargs*)

Add a coherence plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of `cohere()` with modifications on the design.

Notable modification on input argument is

- `grid` is set to `'both'` by default.

See also:

`cohere()` for documentation on valid kwargs.

csd(*ax*, *x*, *y*, **args*, ***kwargs*)

Add a cross-spectral density plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of `csd()` with modifications on the design.

Notable modification on input argument is

- `grid` is set to `'both'` by default.

See also:

`csd()` for documentation on valid kwargs.

errorbar(*ax*, *x*, *y*, **args*, ***kwargs*)

Add an errorbar plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of `errorbar()` with modifications on the design.

Notable modification on input argument is

- `grid` is set to `'both'` by default.

See also:

`errorbar()` for documentation on valid kwargs.

fill_between(*ax*, *x*, *y1*, **args*, ***kwargs*)

Add filled polygons to `matplotlib.axes.Axes` object.

This method is a wrapper of `fill_between()` with modifications on the design.

Notable modification on input argument is

- `grid` is set to `'both'` by default.

See also:

`fill_between()` for documentation on valid kwargs.

fill_betweenx(*ax*, *y*, *x1*, **args*, ***kwargs*)

Add filled polygons to `matplotlib.axes.Axes` object.

This method is a wrapper of `fill_betweenx()` with modifications on the design.

Notable modification on input argument is

- `grid` is set to `'both'` by default.

See also:

`fill_betweenx()` for documentation on valid kwargs.

hist(*ax*, *x*, **args*, ***kwargs*)

Add a histogram plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of `hist()` with modifications on the design.

Notable modification on input argument is

- `grid` is set to `'auto'` by default.

See also:

`hist()` for documentation on valid kwargs.

legend(*ax*, **args*, ***kwargs*)

Place a legend to the input `matplotlib.axes.Axes` object.

Parameters

- **ax** – The input axes object.
- **position** – The position of the legend. Default is `'right'`. Value can be `'left'`, `'right'`, `'top'`, or `'bottom'`.
- **fraction** – The fraction of the height/width of the axes object that will be shrunk to fit the legend.

Returns The legend

All additional input parameters are passed to `legend()`.

Note: The legend in ggplot2 may not work well with `fig.tight_layout()`, which resizes and repositions the `matplotlib.axes.Axes` objects.

See also:

`matplotlib.axes.legend()` for documentation on valid kwargs.

pcolormesh(*ax*, **args*, ***kwargs*)

Add a quadrilateral mesh to `matplotlib.axes.Axes` object.

Parameters

- **ax** – The input axes object.
- **colorbar** – Draw a color bar. Default is `'vertical'`. Value can be `'vertical'`, `'horizontal'`, and `None`.
- **xticks** – Remove the default positional labels and add custom x-axis tick labels. Default is `None`.
- **yticks** – Remove the default positional labels and add custom y-axis tick labels. Default is `None`.

Returns A (`matplotlib.colorbar.Colorbar`, `matplotlib.collections.QuadMesh`) tuple.

All additional input parameters are passed to `pcolormesh()`.

See also:

`matplotlib.axes.pcolormesh()` for documentation on valid kwargs.

plot(*ax*, **args*, ***kwargs*)
Add a line plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of `plot()` with modifications on the design.

Notable modification on input argument is

- `grid` is set to `'both'` by default.

See also:

`plot()` for documentation on valid kwargs.

psd(*ax*, *x*, **args*, ***kwargs*)
Add a power spectral density plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of `psd()` with modifications on the design.

Notable modification on input argument is

- `grid` is set to `'both'` by default.

See also:

`psd()` for documentation on valid kwargs.

scatter(*ax*, *x*, *y*, **args*, ***kwargs*)
Add a scatter plot to the input `matplotlib.axes.Axes` object.

This method is a wrapper of `scatter()` with modifications on the design.

Notable modification on input argument is

- `grid` is set to `'both'` by default.

See also:

`scatter()` for documentation on valid kwargs.

set_palette(*palette*)
Set the palette used for coloring.

Parameters **palette** – The palette used for coloring.

1.3.3 Palette

`palette` is the collection of all available `Palette` provided by MatPlotTheme. All palette classes are derived classes of `Palette`.

Default Palette

class `matplottheme.palette.default.`**Palette**
This class is a collection of all colors provided by the default palette of MatPlotTheme.

cold_map
Defines the cold color map.

cold_warm_map
Defines the cold-warm color map.

color_cycle = ['#66c2a5', '#fc8d62', '#8da0cb', '#e78ac3', '#a6d854', '#ffd92f', '#e5c494', '#b3b3b3']
Defines the color cycle used in all `matplottheme.style.default.Style.plot()`,

matplottheme.style.default.Style.bar(),matplottheme.style.default.Style.barh() and other methods.

dark_frame = '#444444'
Defines the color of plot frame and labels/texts.

frame_bgcolor = '#ffffff'
Defines the background color of plots.

legend_bgcolor = '#dddddd'
Defines the background color of legend

warm_map
Defines the warm color map.

ggplot2 Palette

class matplottheme.palette.ggplot2.**ggplot2Palette**
Bases: matplottheme.palette.default.Palette

This class is a collection of all colors provided by the ggplot2 palette of MatPlotTheme.

CHAPTER 2

Indices and tables

- *genindex*
- *modindex*
- *search*

m