
SpiffyPlots Documentation

Release 0.6.1

Julian Rossbroich

Jul 27, 2022

Contents:

1	Installation	1
1.1	Stable release	1
1.2	From sources	1
2	spiffyplots package	3
2.1	spiffyplots.colors module	3
2.2	spiffyplots.multipanel module	3
3	Indices and tables	7
	Python Module Index	9
	Index	11

1.1 Stable release

To install SpiffyPlots, run this command in your terminal:

```
$ pip install spiffyplots
```

This is the preferred method to install SpiffyPlots, as it will always install the most recent stable release.

If you don't have [pip](#) installed, this [Python installation guide](#) can guide you through the process.

1.2 From sources

The sources for SpiffyPlots can be downloaded from the [Github repo](#).

You can either clone the public repository:

```
$ git clone git://github.com/JRBCH/spiffyplots
```

Or download the [tarball](#):

```
$ curl -OJL https://github.com/JRBCH/spiffyplots/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```


2.1 spiffyplots.colors module

2.2 spiffyplots.multipanel module

The Spiffy MultiPanel class and its methods.

```
class spiffyplots.multipanel.MultiPanel (shape: Optional[Tuple[int, int]] = (2, 2), grid: Union[Iterable[Tuple], Iterable[int]] = None, labels: Union[bool, Iterable[str], Dict[str, tuple], numpy.array] = False, **kwargs)
```

Bases: object

The central object of the *multipanel* module. Initiates a figure with multiple panels.

The `MultiPanel` object is basically a wrapper of matplotlib's `GridSpec`, but tries to simplify some aspects of multi-panel figure generation, such as Figure labels and the layout of panels. Depending on the input, the layout is initialized in one of three ways:

OPTION 1: Initialization based on the `labels` parameter

The `labels` parameter can be passed in as a dictionary, mapping custom figure labels (e.g. 'a', 'b', 'c') to locations in the grid that are defined by Tuples (e.g. {'A': (0, range(2,5))} will make a plot in the first row spanning columns 2-4 and give it the label A.

Similarly, `labels` can be passed as a 2-dimensional np.array of strings. In this case, the strings in the cells of the array correspond to the label of the panels. Adjacent identical labels are considered one panel. For example, the array:

```
[ 'A', 'A', 'D' ]
[ 'B', 'C', 'D' ]
[ 'E', 'E', 'E' ]
```

will create 5 panels, each occupying the space that the respective label takes up in the array.

This option is useful when you want to control both the arrangement of panels, and the order and format of their labels. If `label` is passed in as a dictionary or `np.array`, the `grid` and `shape` parameters are ignored.

OPTION 2: Initialization based on the `grid` parameter:

If option 1 does not apply, the class will try to be initialized through the `grid` parameter.

Example: Generate a two-row figure with 3 columns (panels) in the first row and 2 columns (panels) in the second row:

```
>>> fig = MultiPanel(grid=[3, 2])
```

Example: Generate a 2x3 figure with 5 panels, where one panel spans both rows in the last column:

```
>>> fig = MultiPanel(grid=[(0, 0), (0, 1), (1, 0), (1, 1), (range(0, 2), 2)])
```

OPTION 3: initialization based on the `shape` parameter:

if neither `labels` or `grid` are supplied, the class will generate one panel in each cell of the grid matrix, as defined by the `shape` parameter.

Example: Generate a 3x3 grid with 9 plots of equal size:

```
>>> fig = MultiPanel(shape=(3, 3))
```

Parameters

- **shape** (*Tuple*) – Determines the shape of the MultiPanel grid layout.
- **grid** (*Iterable[*Tuple*], Iterable[*int*]*) – Determines the layout of subplots across the MultiPanel matrix. Defaults to one plot in each cell of the `shape` matrix. Can be one of:
 - Iterable of grid location tuples of form `[rows, columns]`, in which rows and columns are either `int` (for a single cell) or `Iterable` (for spanning multiple cells).
 - Iterable of ints with length `shape[0]`, which defines the number of plots in each row. Each plot then has the size `1 x shape[0]/int`. **Attention:** `shape[0]` must be divisible by every element in `grid`.
- **labels** (*bool, Iterable[*str*], dict, np.array*) – Assigns labels to subplots. Defaults to `False`. Can be one of:
 - Boolean. If `True`, labels are assigned to plots first across rows, then across columns.
 - Iterable of strings assigning labels to subplots, in the same order as defined by `grid`.
 - A Dictionary mapping `[str]` keys to `[Tuple]` locations in the grid. This setting overrides the grid.
 - A `np.array` of the same shape as `shape`, mapping string names to the locations in the grid. Figures can span multiple cells in the grid. Also overrides the grid.

Keyword Arguments

- **figsize** (*Tuple*) – Size of the figure. Will be passed into `matplotlib.pyplot.figure`.
- **label_case** (*str*) – ‘uppercase’ or ‘lowercase’. This and following kwargs are passed to `MultiPanel._draw_labels`.

- **label_weight** (*str*) – Weight of the figure labels. defaults to ‘bold’
- **label_size** (*int*) – Font Size for figure labels. defaults to 14.
- **label_location** (*Tuple*) – Tuple. Location of the figure labels relative to axis origin. Defaults to (-0.25, 1.05)
- **left** (*float*) – left margin. This and following kwargs are passed to `matplotlib.gridspec.GridSpec`
- **right** (*float*) – right margin
- **bottom** (*float*) – bottom margin
- **top** (*float*) – top margin
- **wspace** (*float*) – horizontal spacing
- **hspace** (*float*) – vertical spacing
- **width_ratios** (*Iterable*) – width ratios of columns
- **height_ratios** (*Iterable*) – height ratios of rows

close()

Closes the matplotlib figure object

save (*path: str, format: Union[str, tuple, list] = 'pdf', **kwargs*)

Saves the figure as one or multiple file types

Parameters

- **path** – file path .. rubric:: Example
save(path='figures/figure1, format='pdf') will save the object as figures/figure1.pdf
- **format** – the file format(s) to save as. Defaults to ‘pdf’

CHAPTER 3

Indices and tables

- `genindex`
- `modindex`
- `search`

S

`spiffyplots.colors`, 3

`spiffyplots.multipanel`, 3

C

`close()` (*spiffyplots.multipanel.MultiPanel method*), 5

M

`MultiPanel` (*class in spiffyplots.multipanel*), 3

S

`save()` (*spiffyplots.multipanel.MultiPanel method*), 5

`spiffyplots.colors` (*module*), 3

`spiffyplots.multipanel` (*module*), 3