highlight-text 0.2

✓ Latest version

```
pip install highlight-text
```

Released: Apr 7, 2021

matplotlib functions to plot text with color highlighted substrings

Project links

Homepage

Statistics

GitHub statistics:

Stars: 76

Forks: 3

Project description

png

HighlightText

The purpose of this package is to make effective annotations easier in matplotlib.

In 2020 data journalism has played a vital role in communicating to the public. There are now many publications that routinely use various forms of colored text highlights of key information in the title, that until then has often been shown in legends.

The HighlightText package provides a natural way to specify substrings that should be highlighted and individual font properties that should be used for each of the highlights.
That means using different colors, shading backgrounds with bboxes, using path_effects or different fontsize, weights, or styles are all possible and you are free to choose what best supports highlighting the key information you want your viewers to know.

Open issues/PRs: 1

View statistics for this project via Libraries.io, or by using our public dataset on Google BigQuery

Meta

License: MIT License

Author: znstrider

Requires: Python >=3.6

Maintainers

Avatar for znstrider from gravatar.com znstrider

Classifiers

Framework
- Matplotlib

License
- OSI Approved :: MIT License

Operating System
- OS Independent

Programming Language
- Python :: 3

Topic
- Scientific/Engineering :: Visualization

Installation

```
pip install highlight-text
```

Note

The newest version breaks with the prior syntax of individually specifying highlight_colors and other params for eg. bboxes and path_effects.
You can now provide any matplotlib.text.Text keyword arguments for any of the highlighted substrings into the `highlight_textprops` parameter.
You can familiarize yourself with the new syntax and the possibilities this provides by having a look at the examples below.

Use

This package provides a HighlightText class and two wrapper functions that allow you to plot text with `<highlighted substrings>` in matplotlib:

- ax_text for plotting onto an axes in data coordinates.
- fig_text for plotting onto the figure in figure coordinates.

They take a string with substring delimiters = ['<', '>'] to be highlighted according to the specified highlight_textprops. You can provide other delimiters if necessary.
You must specify a list with the same number of textprop dictionaries as you use `<highlighted substrings>`.

The example below prints the text <font color='yellow'>sunny</font> as yellow and <font color='grey'>cloudy</font> as grey.

A minimal example would be:

```
import matplotlib.pyplot as plt
from highlight_text import HighlightText, ax_text, fig_
# or
import highlight_text # then use highlight_text.ax_text
```

Plotting text in axes coordinates

```
fig, ax = plt.subplots()

# You can either create a HighlightText object
HighlightText(x=0.25, y=0.5,
              s='The weather is <sunny>\nYesterday it w
              highlight_textprops=[{"color": 'yellow'},
                                   {"color": 'grey'}],
              ax=ax)

# You can use the wrapper around the class
ax_text(x = 0, y = 0.5,
        s='The weather is <sunny>\nYesterday it was <cl
        highlight_textprops=[{"color": 'yellow'},
                             {"color": 'grey'}],
        ax=ax)
```

Plotting text in figure coordinates:

```
fig, ax = plt.subplots()

# either pass 'boxcoords': fig.transFigure into the ann

HighlightText(x=0.25, y=0.5,
              s='The weather is <sunny>\nYesterday it w
              highlight_textprops=[{"color": 'yellow'},
                                   {"color": 'grey'}],
              annotationbbox_kw={'boxcoords': fig.trans

# or use the wrapper around the class
fig_text(x=0.25, y=0.5,
         s='The weather is <sunny>\nYesterday it was <c
         highlight_textprops=[{"color": 'yellow'},
                              {"color": 'grey'}])
```

Example1

Further Examples

<font style="color:#2171b5; font-size:16px">You can pass all matplotlib.Text keywords to HighlightText for all text,
and into the highlight_textprops for each of the text highlights.
The highlight_textprops overwrite all other passed keywords for the highlighted substrings. </font>

A showcase use is provided in this notebook
Source:
https://twitter.com/petermckeever/status/1346075580782047233
ColorEncodingExample

Using Path Effects

```
import matplotlib.patheffects as path_effects

def path_effect_stroke(**kwargs):
    return [path_effects.Stroke(**kwargs), path_effects
pe = path_effect_stroke(linewidth=3, foreground="orange

highlight_textprops =\
[{"color": "yellow", "path_effects": pe},
 {"color": "#969696", "fontstyle": "italic", "fontweigh

fig, ax = plt.subplots(figsize=(4, 4))

HighlightText(x=0.5, y=0.5,
              fontsize=16,
              ha='center', va='center',
              s='The weather is <sunny>\nYesterday it w
              highlight_textprops=highlight_textprops,
              ax=ax)
```

Example 2

BBox highlights

Just like colored substrings or using a path_effect, using a bbox to shade the background of
relevant text that is color coded in your plot can make a visualization much more accessible.

```
highlight_textprops =\
[{"bbox": {"edgecolor": "orange", "facecolor": "yellow'
 {"color": "#969696"}]

fig, ax = plt.subplots(figsize=(4, 4))

HighlightText(x=0.5, y=0.5,
              fontsize=16,
              ha='center', va='center',
              s='The weather is <sunny>\nYesterday it w
              highlight_textprops=highlight_textprops,
              ax=ax)
```

Example 3

Different Fontsizes (ie. for Title + Subtitle)

```
highlight_textprops =\
[{"fontsize": 24},
 {"color": "#969696"}]

fig, ax = plt.subplots(figsize=(4, 4))

HighlightText(x=0.5, y=0.5,
              fontsize=16,
              ha='center', va='center',
              s='<This is a title.>\n<and a subtitle>',
              highlight_textprops=highlight_textprops,
              fontname='Roboto',
              ax=ax)
```

Example 5

This example taken from german news publication "Der Spiegel" uses bbox highlights and a different fontsize for title and subtitle.

The code is provided in this notebook
Source of the Graphic:
https://www.spiegel.de/wissenschaft/medizin/coronavirus-in-europa-die-zweite-welle-rollt-a-1d5b12a1-162d-48a3-8e1e-40235c996080?sara_ecid=soci_upd_wbMbjhOSvViISjc8RPU89NcCvtlFcJ

Title BBox Example

Original Graphic:

Original Spiegel Graphic

Text Alignment and seperation between lines

```
highlight_textprops =\
[{"fontsize": 12, 'color': '0.4'},
 {"fontsize": 24, "weight": "bold"},
 {"fontsize": 14, "color": "0.3"}]

fig, ax = plt.subplots(figsize=(12, 2))
ax.axis('off')

HighlightText(x=0.5, y=0.5,
              ha='center', va='center', # alignment of
              s='<In 2021>\n'
                '<Manchester City dominates>\n'
                '<With a series of 11 straight wins Cit
              highlight_textprops=highlight_textprops,
              textalign='center', # horizontal alignmen
              vsep=12, # vertical seperation between l
              ax=ax)
```

Example 8

Custom Linespacing by using invisible text with a fitting fontsize

```
highlight_textprops =\
[{"fontsize": 24},
 {"alpha": 0, "fontsize": 6},
 {"color": "#969696"}]

fig, ax = plt.subplots(figsize=(4, 4))

HighlightText(x=0.5, y=0.5,
              fontsize=16,
              ha='center', va='center',
              s='<This is a title.>\n<ZERO ALPHA TEXT>\
              highlight_textprops=highlight_textprops,
              fontname='Roboto',
              ax=ax)
```

Example 6

Axes insets on top of highlighted substrings

This is great for embedding legends into your title or markers into annotations.
Look at some of John Burn-Murdoch's (@jburnmurdoch) Plots. He has mastered this.

An Example is provided in this notebook
Source:
https://twitter.com/jburnmurdoch/status/1319277057650556936/photo/1
Financial-Times Example

A more basic example looks like follows:
Instead of plotting on the inset axes you can also inset images with this.

```
highlight_textprops =\
[{"alpha": 0},
 {"alpha": 0}]

fig, ax = plt.subplots(figsize=(4, 4))

ht = HighlightText(x=0.5, y=0.5,
              fontsize=16,
              ha='center', va='center',
              s='Today it rained this much <SPACE>\n'
                'Yesterday only this much  <SPACE>',
                highlight_textprops=highlight_textprops,
                ax=ax)

insets = ht.make_highlight_insets([True, True])
for haxes, color, height in zip(ht.highlight_axes, ['b
    if haxes:
        haxes.bar(x=[0.25], height=[height], bottom=0.2
        haxes.set_ylim(0, 1)
        haxes.set_xlim(0, 1)
```

<font color="red">Important:</font>
If you make an axes inset using a script, you will have to redraw the canvas!

So at the end of your plotting call:

```
fig.canvas.draw()
plt.show()
```

Example 4

AnnotationBbox BBox

We can also place a Bounding Box around the whole AnnotationBbox that holds all of our text by setting 'frameon': True within the annotationbbox_kw dictionary.

```
fig, ax = plt.subplots(figsize=(4, 2))

ht = HighlightText(x=0.5, y=0.5,
                fontsize=12,
                ha='center', va='center',
                s='<Grocery List:>\nBananas\nOatmeal',
                highlight_textprops=[{'size': 20}],
                annotationbbox_kw={'frameon': True, 'pad'
                                   'bboxprops': {'facecol
                ax=ax)
```

Example 7

Arrowprops

The AnnotationBBox that holds our texts takes a `xybox` keyword argument that you can input to `annotationbbox_kw`. In combination with `arrowprops` this allows us to draw an arrow from xybox to the annotation point given by (x, y).

```
fig, ax = plt.subplots(figsize=(4, 3))

ht = HighlightText(x=0.5, y=0.5,
                   fontsize=12,
                   ha='center', va='center',
                   s='<Annotation Title:>\nPoint 1\nPo
                   highlight_textprops=[{'size': 20}],
                   annotationbbox_kw={'frameon': True,
                                      'arrowprops': dict
                                      'xybox': (3, 0.5),
                                       },
              ax=ax)

ax.set_xlim(0, 3)
```

Example 9

```
"""
Args:
    x (float): x-position
    y (float): y-position
    s (str): textstring with <highlights>
    ha (str, optional): horizontal alignment of the Ann
    va (str, optional): vertical alignment of the Annot
    highlight_textprops (List[dict], optional): list of
    textalign (str, optional): Text Alignment for the A
    delim (tuple, optional): characters that enclose <h
    annotationbbox_kw (dict, optional): AnnotationBbox
    ax (Axes, optional): Defaults to None.
    fig (Figure, optional): Defaults to None.
    add_artist (bool, optional): Whether to add the Ann
    vpad (int, optional): vertical padding of the Highl
    vsep (int, optional): vertical seperation between t
    hpad (int, optional): horizontal padding of a rows
    hsep (int, optional): horizontal seperation between
"""
```

AWS

AWS
Cloud computing

Datadog

Datadog
Monitoring

Facebook / Instagram

Facebook / Instagram
PSF Sponsor

Fastly
CDN

Google
Object Storage and Download Analytics

Huawei
PSF Sponsor

Microsoft
PSF Sponsor

NVIDIA
PSF Sponsor

Pingdom
Monitoring

Salesforce
PSF Sponsor

Sentry
Error logging

StatusPage
Status page