plotannot

Mette Bentsen

Jun 12, 2022

CONTENTS

plotannot is a python package for highlighting and adjusting axes ticklabels, and remove overlapping labels. The project code lives on GitHub (msbentsen/plotannot).

Please submit any issues to the github issues page.

CHAPTER

ONE

CONTENT

1.1 Installation

To install from PyPI, simply run:

```
pip install plotannot
```

To install from the GitHub repository, run:

```
git clone https://github.com/msbentsen/plotannot.git
cd plotannot
pip install .
```

1.2 Module API

plotannot.functions.**annotate_ticks**(*ax*, *axis*, *labels*, *expand_axis=0*, *rel_label_size=1.1*, *perp_shift=5*, *rel_tick_size=0.25*, *resolution=1000*, *speed=0.1*, *verbosity=1*)

Annotate ticks with a subset of labels, and shift to overlapping labels.

Parameters

- **ax** (*matplotlib.axes.Axes*) – Axes object holding the plot and labels to annotate.
- **axis** (*str*) – Name of axis to annotate. Must be one of ["xaxis", "yaxis", "left", "right", "bottom", "top"].
- **labels** (*list of str*) – A list of labels to annotate. Must be a list of strings (or values convertible to strings) corresponding to the labels to show in plot.
- **expand_axis** (*float or tuple, optional*) – Expand the annotation axis by this amount of the total axis width. Can be either float or tuple of floats. Corresponds to the relative size of axes to expand with, e.g. 0.1 extends with 5% of the axis size in both directions (total 10%). A tuple of (0.1,0.2) extends the axis with 10% in the beginning (left or bottom) and 20% (right or top). Default: 0.
- **rel_label_size** (*float, optional*) – Relative size of labels to use for measuring overlaps. Default: 1.1.
- **perp_shift** (*float, optional*) – Perpendicular shift of labels. Represents the relative length of ticks of the axis. Default: 5 (5 times the length of ticks).
- **rel_tick_size** (*float, optional*) – Relative size of the horizontal part of the annotation lines as a fraction of perp_shift. Default: 0.25.

- **resolution** (*int, optional*) – Resolution for finding overlapping labels. Default: 1000.
- **speed** (*float, optional*) – The speed with which the labels are moving when removing overlaps. A float value between 0-1. Default: 0.1.
- **verbosity** (*int, optional*) – The level of logging from the function. An integer between 0 and 3, corresponding to: 0: only errors, 1: minimal, 2: debug, 3: spam debug. Default: 1.

plotannot.functions.**format_ticklabels**(*ax*, *axis*, *labels=None*, *format_ticks=False*, *verbosity=1*, ***kwargs*)

Format ticklabels of a given axis using attributes such as color, fontsize, fontweight, etc.

Parameters

- **ax** (*matplotlib.axes.Axes*) – Axes object holding the plot and labels to annotate.
- **axis** (*str*) – Name of axis to annotate. Must be one of ["xaxis", "yaxis", "left", "right", "bottom", "top"].
- **labels** (*list of str, optional*) – A list of labels to annotate. Must be a list of strings corresponding to the labels to show in plot. If None, all labels are used. Default: None.
- **format_ticks** (*bool, optional*) – If True, also format the ticklines of the axis. Default: False.
- **verbosity** (*int, optional*) – The level of logging from the function. An integer between 0 and 3, corresponding to: 0: only errors, 1: minimal, 2: debug, 3: spam debug. Default: 1.
- **kwargs** (*args, optional*) – Additional keyword arguments containing the attributes to set for labels. Each attribute is used as a function "set_" + attribute for the label, e.g. "color='red'" will set the color of the label to red using the label-function 'set_color'.

1.3 Example notebooks

1.3.1 Examples of plotannot annotations

This notebooks presents a few examples of the *plotannot* package for annotating axis labels.

Getting ready

[1]:
```
#Load a few packages for plotting
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
np.random.seed(1) #make notebook reproducible
```

[2]:
```
#Load plotannot
import plotannot
```

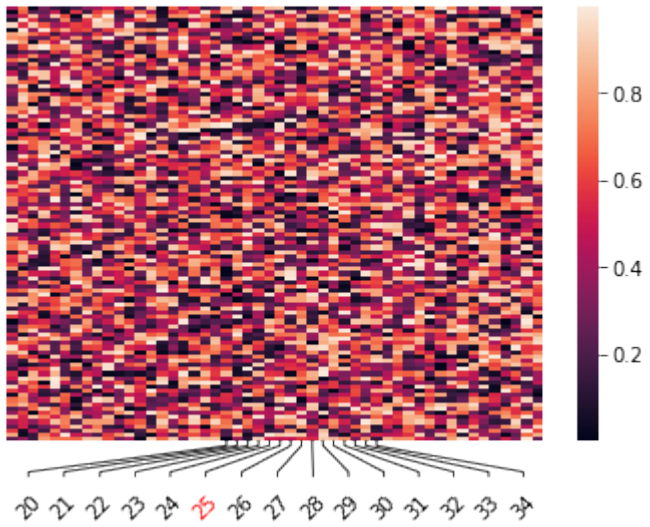
[3]:
```
#Create some fake data
table = pd.DataFrame(np.random.random((100,50)))
table.head()
```

[3]:
```
          0         1         2         3         4         5         6   \
0  0.417022  0.720324  0.000114  0.302333  0.146756  0.092339  0.186260
1  0.019367  0.678836  0.211628  0.265547  0.491573  0.053363  0.574118
2  0.326645  0.527058  0.885942  0.357270  0.908535  0.623360  0.015821
3  0.071974  0.967276  0.568100  0.203293  0.252326  0.743826  0.195429
4  0.950176  0.556653  0.915606  0.641566  0.390008  0.485991  0.604310

          7         8         9   ...        40        41        42        43  \
0  0.345561  0.396767  0.538817  ...  0.988861  0.748166  0.280444  0.789279
1  0.146729  0.589306  0.699758  ...  0.114746  0.949489  0.449912  0.578390
2  0.929437  0.690897  0.997323  ...  0.556240  0.136455  0.059918  0.121343
3  0.581359  0.970020  0.846829  ...  0.263297  0.065961  0.735066  0.772178
4  0.549548  0.926181  0.918733  ...  0.315245  0.892889  0.577857  0.184010

         44        45        46        47        48        49
0  0.103226  0.447894  0.908596  0.293614  0.287775  0.130029
1  0.408137  0.237027  0.903380  0.573679  0.002870  0.617145
2  0.044552  0.107494  0.225709  0.712989  0.559717  0.012556
3  0.907816  0.931972  0.013952  0.234362  0.616778  0.949016
4  0.787929  0.612031  0.053909  0.420194  0.679069  0.918602

[5 rows x 50 columns]
```

[4]:
```
#Prepare some english words to use for labels
import english_words
words = sorted(list(english_words.english_words_lower_set))
rand_integers = np.random.randint(0, len(words), size=100)
words = [words[i] for i in rand_integers] #100 random words
```

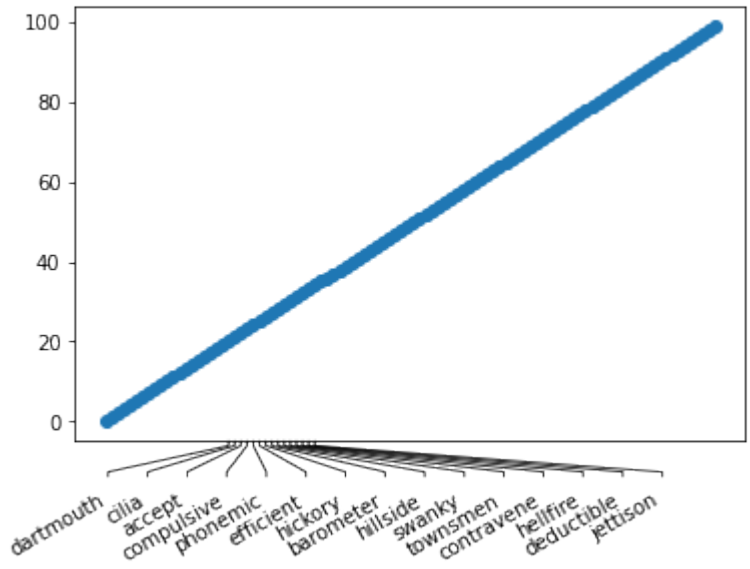
Simple example

[5]:
```
#Plot heatmap
ax = sns.heatmap(table, xticklabels=True, yticklabels=False)

#Rotate all labels
plotannot.format_ticklabels(ax, axis="xaxis", rotation=45)

#Annotate labels
to_label = range(20,35)
plotannot.annotate_ticks(ax, axis="xaxis", labels=to_label)

#Color individual labels
plotannot.format_ticklabels(ax, axis="xaxis", labels=[25], color="red")

plt.savefig("simple_example.png", bbox_inches="tight", facecolor='white')
```



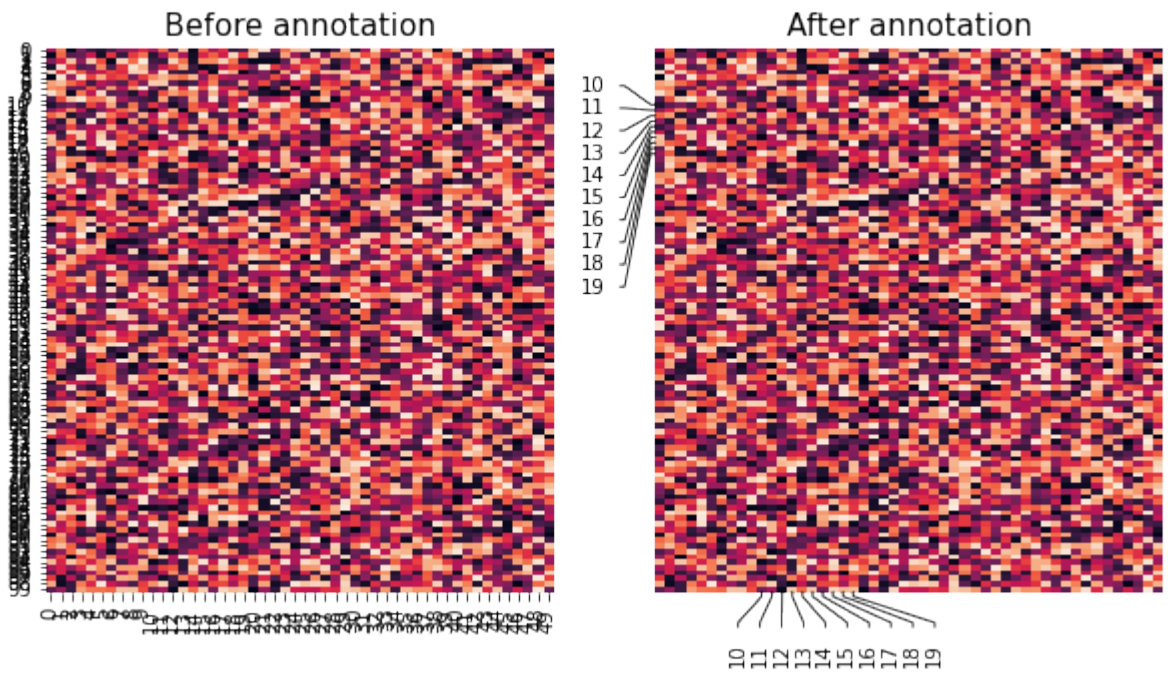
Annotate axis of scatter plots

[6]:
```
fig, ax = plt.subplots()
ax.scatter(words, range(len(words)))

#First rotate to make labels fit
plotannot.format_ticklabels(ax, axis="xaxis", rotation=90)

#Annotate labels
to_label = [words[i] for i in range(20,35)]
plotannot.annotate_ticks(ax, axis="xaxis", labels=to_label, rel_label_size=2) #rel_label_
↪size gives a bit better space for the labels

#Finally, rotate labels to be more visible
plotannot.format_ticklabels(ax, axis="xaxis", rotation=30, ha="right")
```



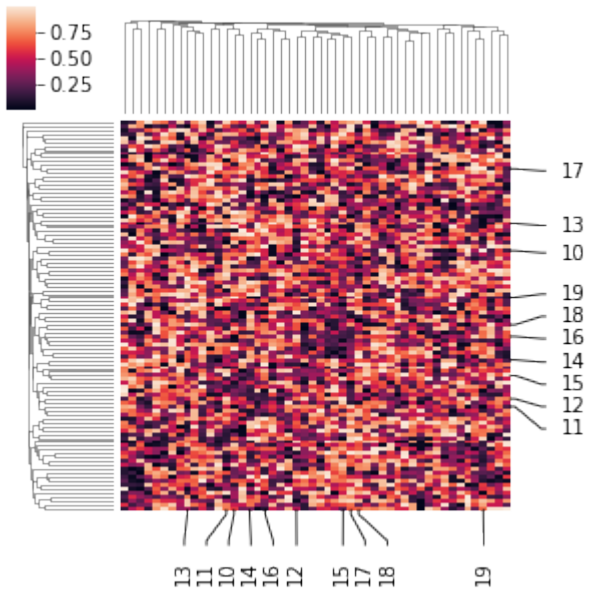
Annotate x- and/or y-axis ticks

[7]:
```
#Plot data with labels
fig, (ax1, ax2) = plt.subplots(1,2, figsize=(10,5))
sns.heatmap(table, yticklabels=True, xticklabels=True, cbar=False, ax=ax1)
sns.heatmap(table, yticklabels=True, xticklabels=True, cbar=False, ax=ax2)

#Apply annotation to second axes
to_label = range(10,20)
plotannot.annotate_ticks(ax2, axis="xaxis", labels=to_label)
plotannot.annotate_ticks(ax2, axis="yaxis", labels=to_label)

#Label axes
_ = ax1.set_title("Before annotation", size=15)
_ = ax2.set_title("After annotation", size=15)

plt.savefig("before_after.png", bbox_inches="tight", facecolor='white')
```



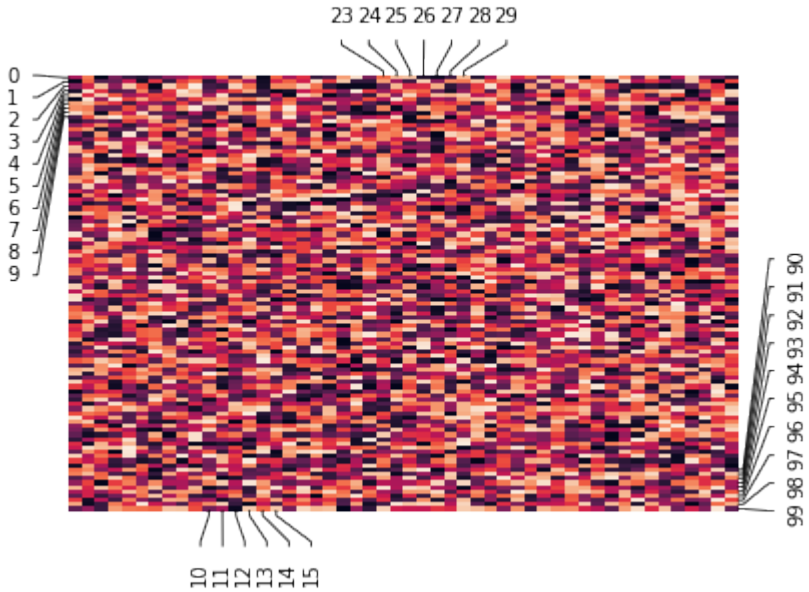
The function *annotate_ticks* also works directly on seaborn objects such as the ClusterGrid of *sns.clustermap*:

[8]:
```
g = sns.clustermap(table, yticklabels=True, xticklabels=True, figsize=(4,4))

to_label = range(10,20)
plotannot.annotate_ticks(g, axis="xaxis", labels=to_label)
plotannot.annotate_ticks(g, axis="yaxis", labels=to_label)
```



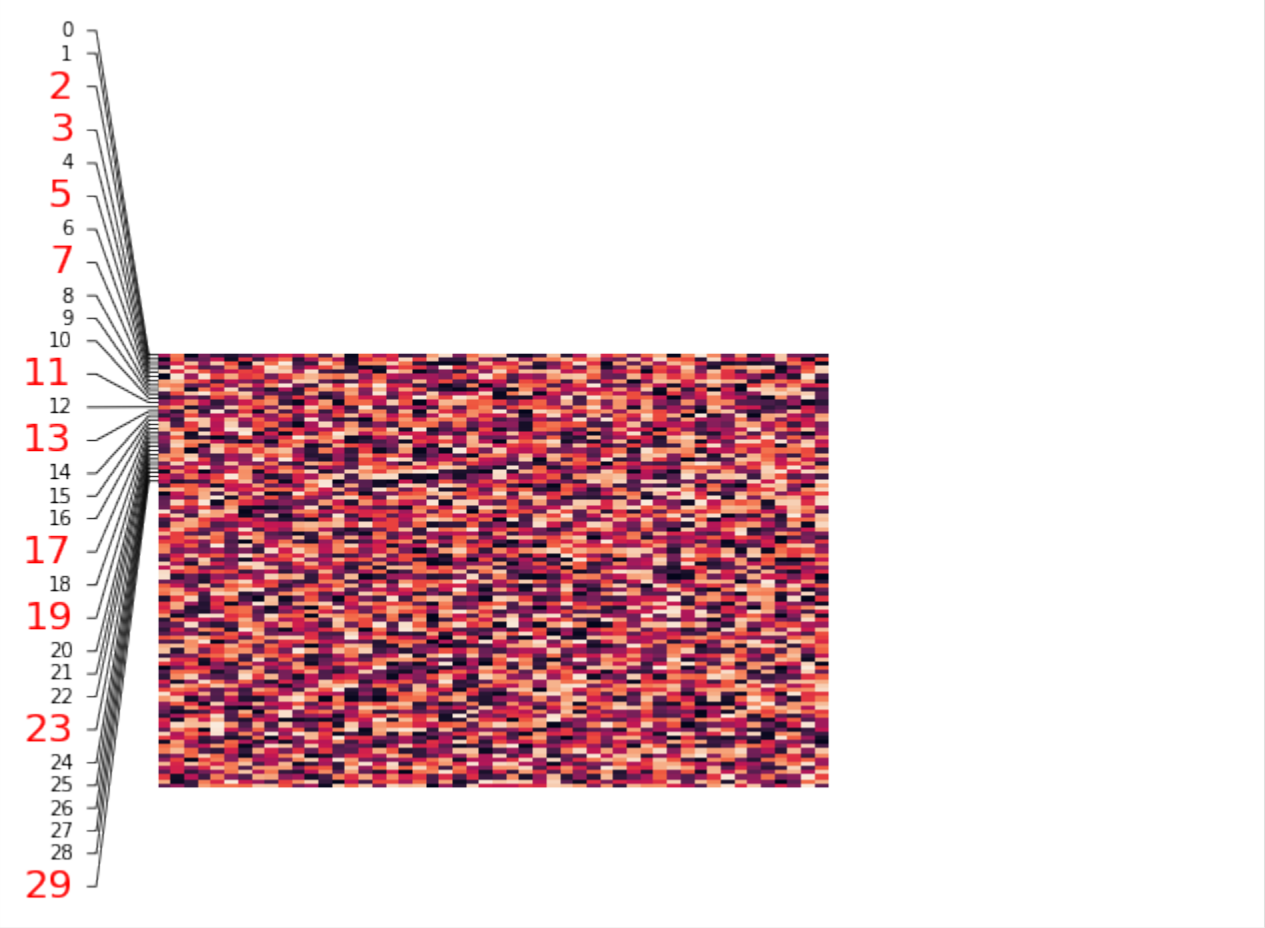
Choose specific labels for bottom/top and left/right axes

[9]:
```
#Plot data and turn on all labels on
ax = sns.heatmap(table, yticklabels=True, xticklabels=True, cbar=False)
ax.tick_params(right=True, top=True, left=True, bottom=True,
               labelleft=True, labelbottom=True, labelright=True, labeltop=True)

#Apply annotation for each axis
left_labels = range(10)
plotannot.annotate_ticks(ax, axis="left", labels=left_labels)

right_labels = range(90,100)
plotannot.annotate_ticks(ax, axis="right", labels=right_labels)

bottom_labels = range(10,16)
plotannot.annotate_ticks(ax, axis="bottom", labels=bottom_labels)

top_labels = range(23,30)
plotannot.annotate_ticks(ax, axis="top", labels=top_labels)
```



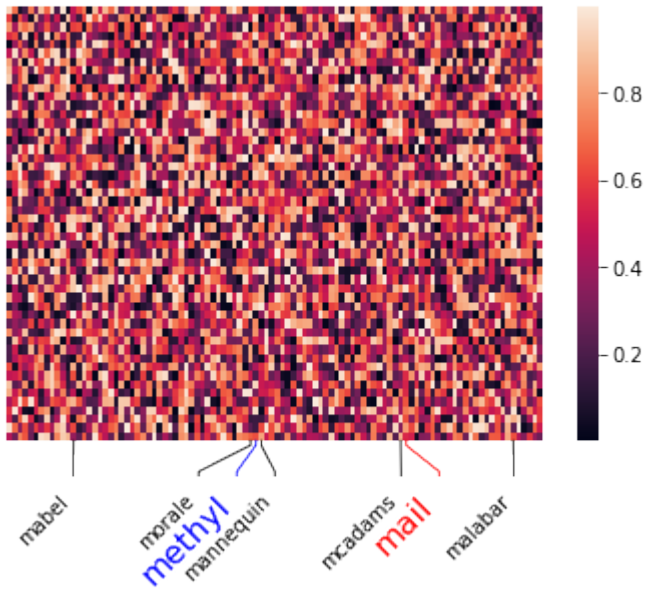
Highlight specific rows with color, size and annotation

[10]:
```
ax = sns.heatmap(table, xticklabels=False, yticklabels=True, cbar=False)

#Color a subset of labels
to_format = [2,3,5,7,11,13,17,19,23,29]
plotannot.format_ticklabels(ax, "yaxis", labels=to_format, color="red", fontsize=20)

#Annotate all ticks
plotannot.annotate_ticks(ax, "yaxis", labels=range(0,30), expand_axis=1, perp_shift=10)
```



Highlight and rotate labels

[11]:
```
#Plot data
word_table = pd.DataFrame(np.random.random((50,100)), columns=words)
ax = sns.heatmap(word_table, yticklabels=False, xticklabels=True)

#Label specific words
plotannot.format_ticklabels(ax, "xaxis", labels=["mail"], color="red", format_ticks=True,
↪ fontsize=16)
plotannot.format_ticklabels(ax, "xaxis", labels=["methyl"], color="blue", format_
↪ticks=True, fontsize=16)

#Annotate ticks starting with "m"
m_words = [word for word in words if word.startswith("m")]
plotannot.annotate_ticks(ax, "xaxis", labels=m_words, rel_label_size=1.5)

#Rotate all ticklabels to 45 degrees
plotannot.format_ticklabels(ax, "xaxis", rotation=45, ha="right")
```



1.3.2 Customization of plotannot annotations

This notebook contains examples of how to customize the look of *plotannot* annotations.

Load packages and data

[1]:
```
#Load a few packages for plotting
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
np.random.seed(1) #make notebook reproducible
```

[2]:
```
#Load plotannot
import plotannot
```

[3]:
```
#Create some fake data
table = pd.DataFrame(np.random.random((100,50)))
table.head()
```

[3]:

	0	1	2	3	4	5	6	\
0	0.417022	0.720324	0.000114	0.302333	0.146756	0.092339	0.186260	
1	0.019367	0.678836	0.211628	0.265547	0.491573	0.053363	0.574118	
2	0.326645	0.527058	0.885942	0.357270	0.908535	0.623360	0.015821	
3	0.071974	0.967276	0.568100	0.203293	0.252326	0.743826	0.195429	
4	0.950176	0.556653	0.915606	0.641566	0.390008	0.485991	0.604310	

	7	8	9	...	40	41	42	43	\
0	0.345561	0.396767	0.538817	...	0.988861	0.748166	0.280444	0.789279	
1	0.146729	0.589306	0.699758	...	0.114746	0.949489	0.449912	0.578390	

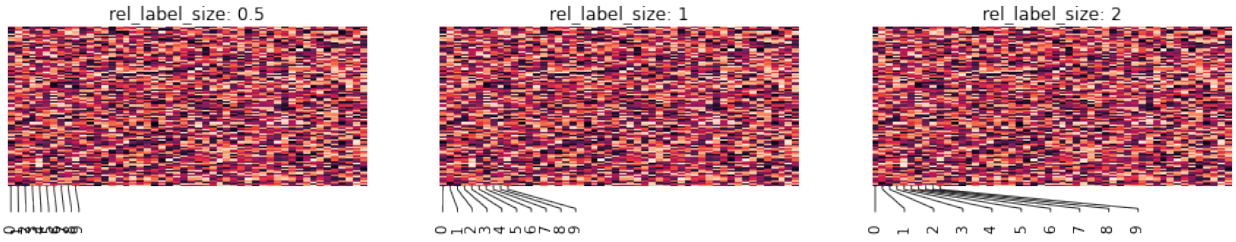
(continues on next page)

(continued from previous page)

```
2  0.929437  0.690897  0.997323  ...  0.556240  0.136455  0.059918  0.121343
3  0.581359  0.970020  0.846829  ...  0.263297  0.065961  0.735066  0.772178
4  0.549548  0.926181  0.918733  ...  0.315245  0.892889  0.577857  0.184010

         44        45        46        47        48        49
0  0.103226  0.447894  0.908596  0.293614  0.287775  0.130029
1  0.408137  0.237027  0.903380  0.573679  0.002870  0.617145
2  0.044552  0.107494  0.225709  0.712989  0.559717  0.012556
3  0.907816  0.931972  0.013952  0.234362  0.616778  0.949016
4  0.787929  0.612031  0.053909  0.420194  0.679069  0.918602

[5 rows x 50 columns]
```

Control the relative width of labels when calculating overlaps with 'rel_label_size'

[4]:
```
fig, axarr = plt.subplots(ncols=3, figsize=(15,2))

to_label = range(0,10)
rel_label_sizes = [0.5,1,2]

for i, ax in enumerate(axarr):
    sns.heatmap(table, xticklabels=True, yticklabels=False, cbar=False, ax=ax)

    plotannot.annotate_ticks(ax, axis="xaxis", labels=to_label, rel_label_size=rel_label_
↪sizes[i])
    ax.set_title(f"rel_label_size: {rel_label_sizes[i]}")
```



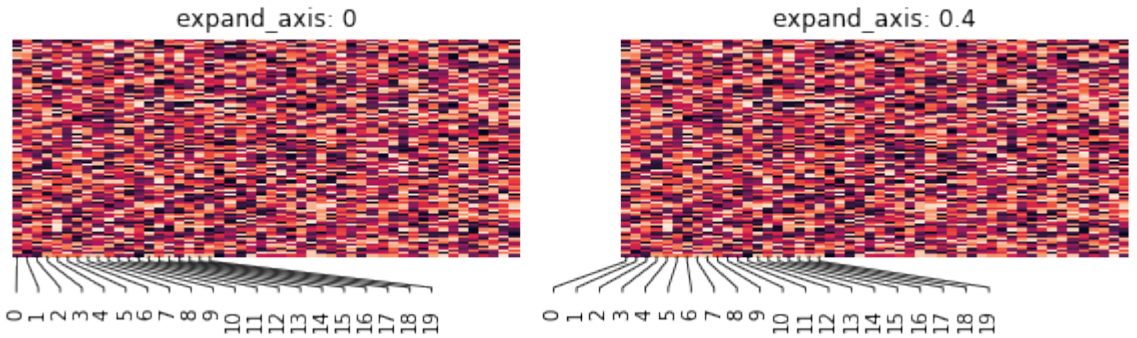
Control extent of axis with 'expand_axis'

[5]:
```
fig, axarr = plt.subplots(ncols=2, figsize=(10,2))

ax = sns.heatmap(table, xticklabels=True, yticklabels=False, cbar=False)
to_label = range(0,20)
expand_axis_lst = [0, 0.4]

#Apply annotation
for i, ax in enumerate(axarr):
    sns.heatmap(table, xticklabels=True, yticklabels=False, cbar=False, ax=ax)

    plotannot.annotate_ticks(ax, axis="xaxis", labels=to_label, expand_axis=expand_axis_
↪lst[i])
```

(continues on next page)

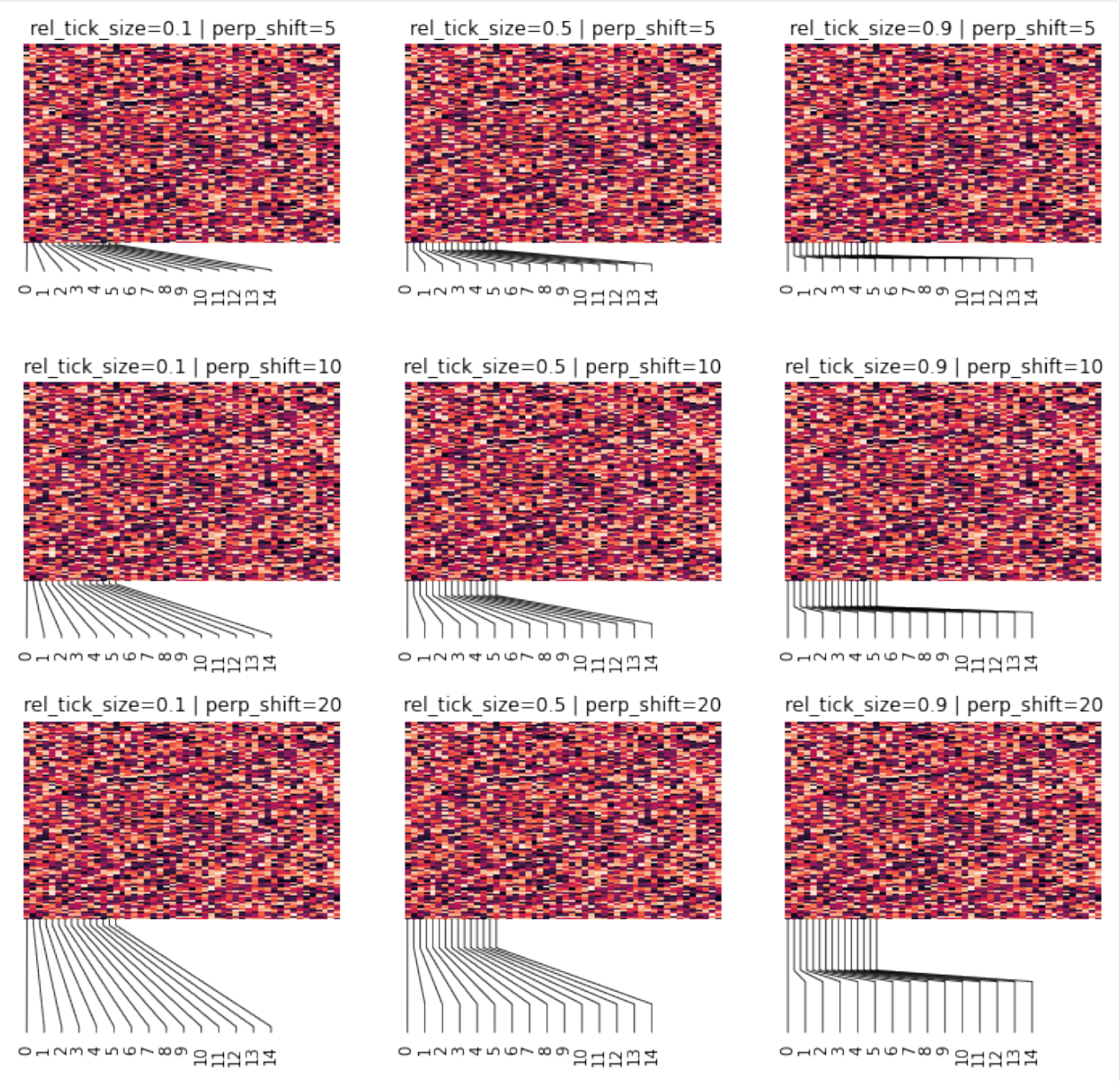
(continued from previous page)

```
    ax.set_title(f"expand_axis: {expand_axis_lst[i]}")
```



Control the extent of the annotation lines with 'rel_tick_size' and 'perp_shift'

[6]:
```
#Options for plotannot
to_label = range(0,15)
rel_tick_size_lst = [0.1,0.5,0.9]
perp_shift_lst = [5,10,20]

#Setup plot
fig, axarr = plt.subplots(3,3, figsize=(12,10))
plt.subplots_adjust(hspace=0.7)

#Plot and annotate with different options
for i, perp_shift in enumerate(perp_shift_lst):
    for j, rel_tick_size in enumerate(rel_tick_size_lst):

        ax = axarr[i,j]
        sns.heatmap(table, xticklabels=True, yticklabels=False, cbar=False, ax=ax)

        plotannot.annotate_ticks(ax, axis="xaxis", labels=to_label, rel_tick_size=rel_
↪tick_size, perp_shift=perp_shift)
        _ = ax.set_title(f"rel_tick_size={rel_tick_size} | perp_shift={perp_shift}")
```


rel_tick_size=0.1 | perp_shift=5
rel_tick_size=0.5 | perp_shift=5
rel_tick_size=0.9 | perp_shift=5
rel_tick_size=0.1 | perp_shift=10
rel_tick_size=0.5 | perp_shift=10
rel_tick_size=0.9 | perp_shift=10
rel_tick_size=0.1 | perp_shift=20
rel_tick_size=0.5 | perp_shift=20
rel_tick_size=0.9 | perp_shift=20
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14

PYTHON MODULE INDEX

INDEX