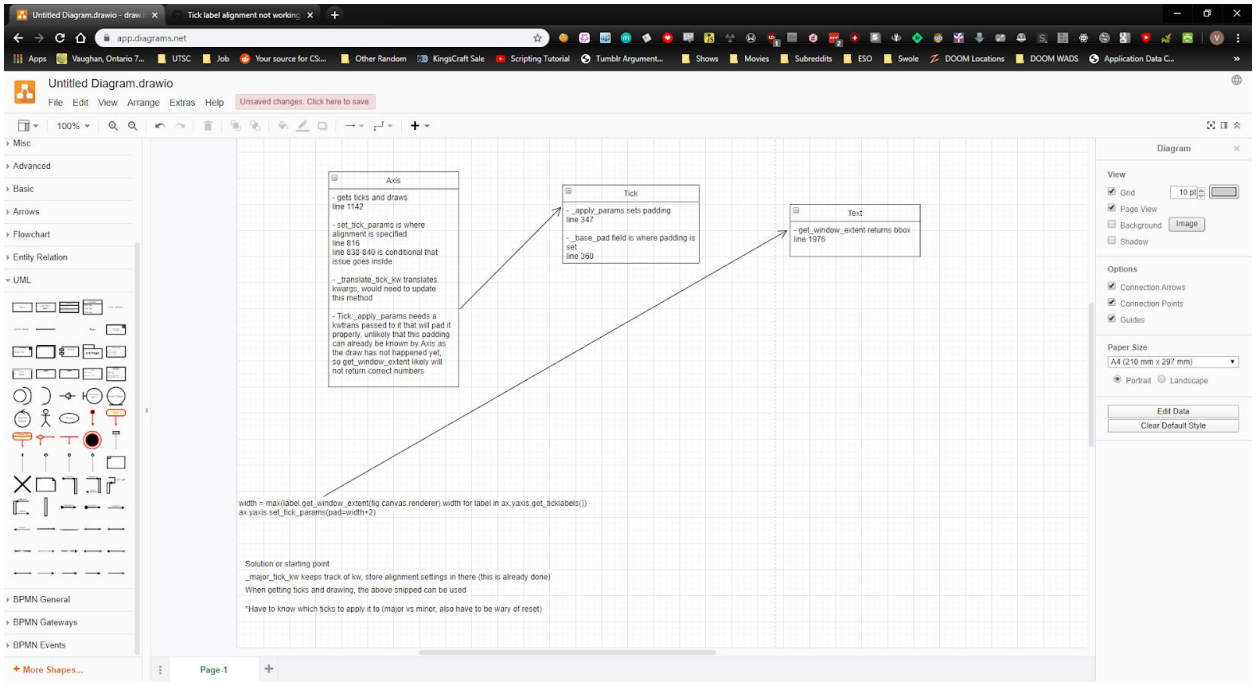
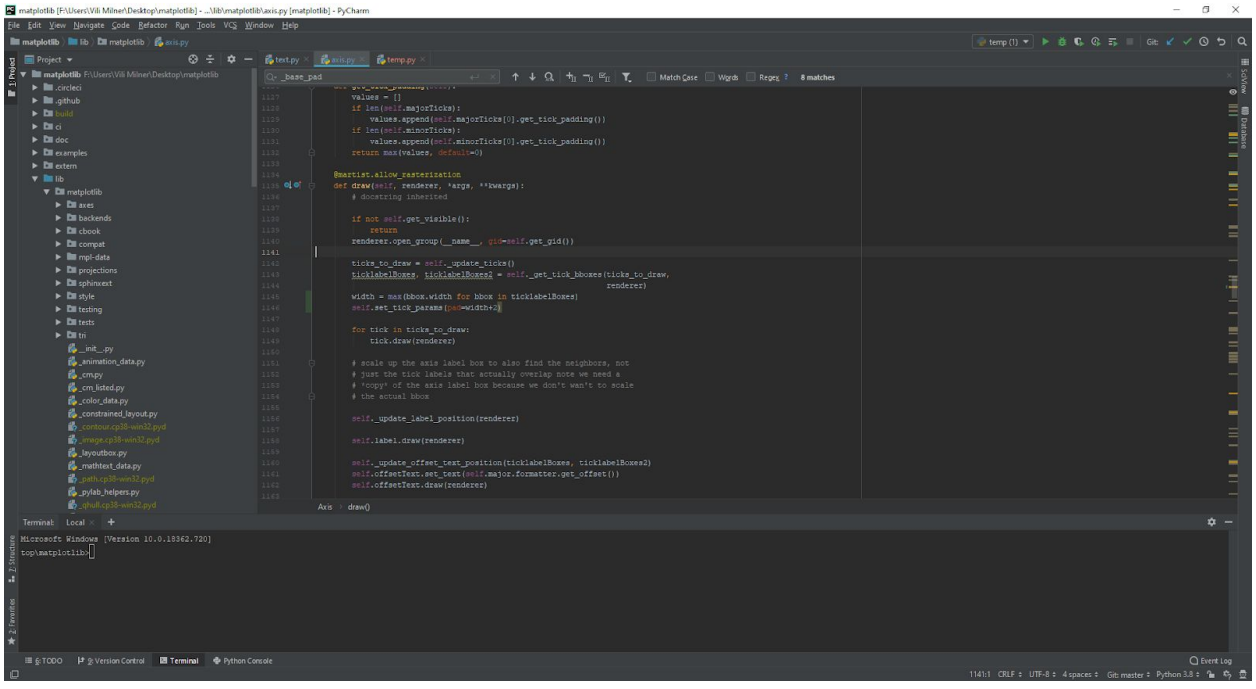
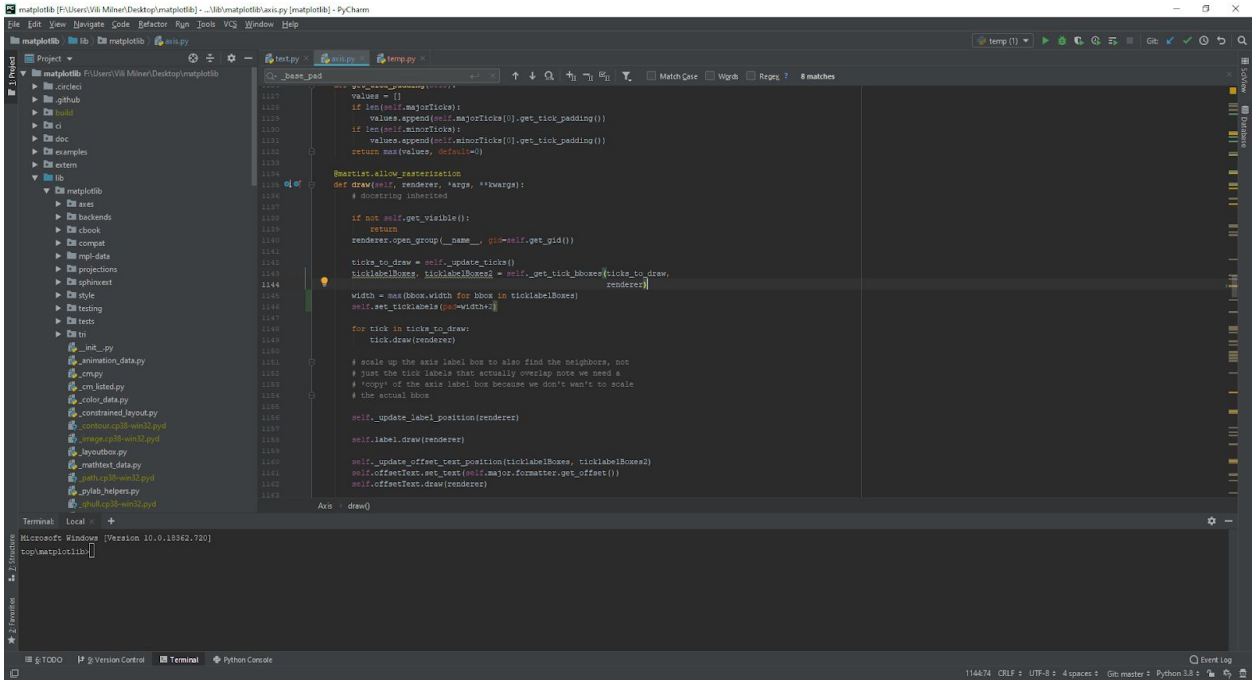
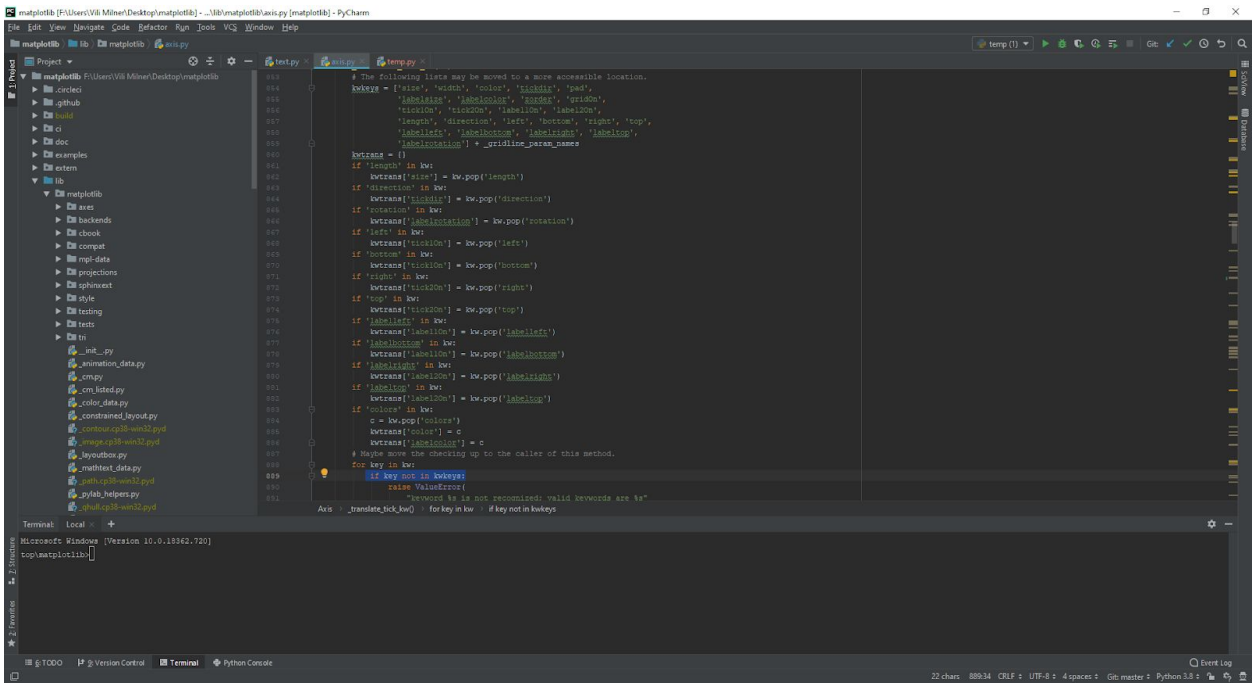
Possible solution:



- Draw twice
- Don't have to get_windows_extend
- Get the max of tick_label boxes




matplotlib [F:\Users\Vili Milner\Desktop\matplotlib] - ...\lib\matplotlib\axis.py [matplotlib] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
matplotlib lib matplotlib axis.py
temp (1)
Git:
Project
text.py
axis.py
temp.py
_base_pad
Match Case
Words
Regex
8 matches
values = []
if len(self.majorTicks):
values.append(self.majorTicks[0].get_tick_padding())
if len(self.minorTicks):
values.append(self.minorTicks[0].get_tick_padding())
return max(values, default=0)
@martist.allow_rasterization
def draw(self, renderer, *args, **kwargs):
# docstring inherited
if not self.get_visible():
return
renderer.open_group(__name__, gid=self.get_gid())
ticks_to_draw = self._update_ticks()
ticklabelBoxes, ticklabelBoxes2 = self._get_tick_bboxes(ticks_to_draw,
renderer)
width = max(bbox.width for bbox in ticklabelBoxes)
self.set_ticklabels(pad=width+2)
for tick in ticks_to_draw:
tick.draw(renderer)
# scale up the axis label box to also find the neighbors, not
# just the tick labels that actually overlap note we need a
# *copy* of the axis label box because we don't wan't to scale
# the actual bbox
self._update_label_position(renderer)
self.label.draw(renderer)
self._update_offset_text_position(ticklabelBoxes, ticklabelBoxes2)
self.offsetText.set_text(self.major.formatter.get_offset())
self.offsetText.draw(renderer)
Axis draw()
Terminal: Local
Microsoft Windows [Version 10.0.18362.720]
(c) 2019 Microsoft Corporation. All rights reserved.
top\matplotlib>
6: TODO
9: Version Control
Terminal
Python Console
Event Log
1144:74 CRLF UTF-8 4 spaces Git: master Python 3.8


matplotlib [F:\Users\Vili Milner\Desktop\matplotlib] - ...\lib\matplotlib\axis.py [matplotlib] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
matplotlib lib matplotlib axis.py
temp (1)
Git:
Project
text.py
axis.py
temp.py
# The following lists may be moved to a more accessible location.
kwkeys = ['size', 'width', 'color', 'tickdir', 'pad',
'labelsize', 'labelcolor', 'zorder', 'gridOn',
'tick1On', 'tick2On', 'label1On', 'label2On',
'length', 'direction', 'left', 'bottom', 'right', 'top',
'labelleft', 'labelbottom', 'labelright', 'labeltop',
'labelrotation'] + _gridline_param_names
kwtrans = {}
if 'length' in kw:
kwtrans['size'] = kw.pop('length')
if 'direction' in kw:
kwtrans['tickdir'] = kw.pop('direction')
if 'rotation' in kw:
kwtrans['labelrotation'] = kw.pop('rotation')
if 'left' in kw:
kwtrans['tick1On'] = kw.pop('left')
if 'bottom' in kw:
kwtrans['tick1On'] = kw.pop('bottom')
if 'right' in kw:
kwtrans['tick2On'] = kw.pop('right')
if 'top' in kw:
kwtrans['tick2On'] = kw.pop('top')
if 'labelleft' in kw:
kwtrans['label1On'] = kw.pop('labelleft')
if 'labelbottom' in kw:
kwtrans['label1On'] = kw.pop('labelbottom')
if 'labelright' in kw:
kwtrans['label2On'] = kw.pop('labelright')
if 'labeltop' in kw:
kwtrans['label2On'] = kw.pop('labeltop')
if 'colors' in kw:
c = kw.pop('colors')
kwtrans['color'] = c
kwtrans['labelcolor'] = c
# Maybe move the checking up to the caller of this method.
for key in kw:
if key not in kwkeys:
raise ValueError(
"keyword %s is not recognized; valid keywords are %s"
Axis _translate_tick_kw() for key in kw if key not in kwkeys
Terminal: Local
Microsoft Windows [Version 10.0.18362.720]
(c) 2019 Microsoft Corporation. All rights reserved.
top\matplotlib>
6: TODO
9: Version Control
Terminal
Python Console
Event Log
22 chars 889:34 CRLF UTF-8 4 spaces Git: master Python 3.8

Notes:
Ticks are created during draw, we can modify it before drawing
On line 1142 in axis class:

```
        ticks_to_draw = self._update_ticks()
        ticklabelBoxes, ticklabelBoxes2 =
self._get_tick_bboxes(ticks_to_draw,renderer)

     <our code goes here>

        for tick in ticks_to_draw:
            tick.draw(renderer)
```

Set_tick_params: kw_trans updates the minor ticks translate

- Make sure horizontal alignment is in kw keys (in the _translate_tick_kw method)

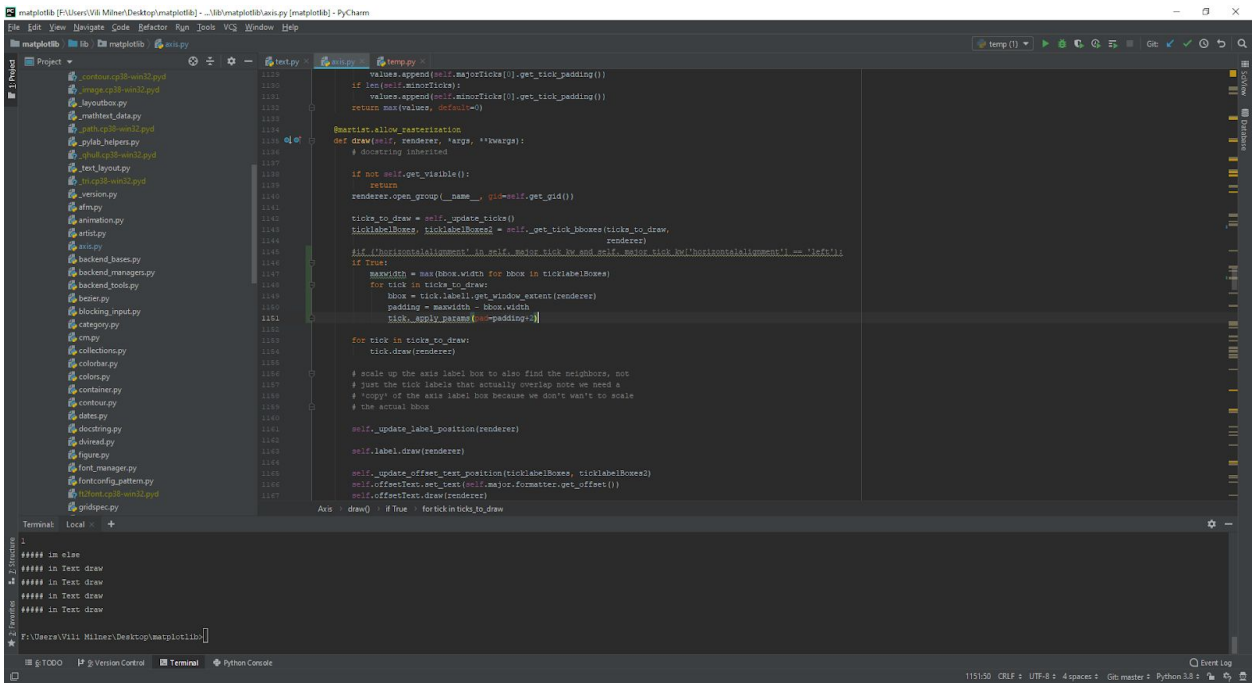
Set_horizontal_alignment: can't, because it's within the bounding box

Not sure if need to differentiate between x and y axis???

- Might not need to

apply_params applies to each tick individually: it does it for every tick

Attempt:



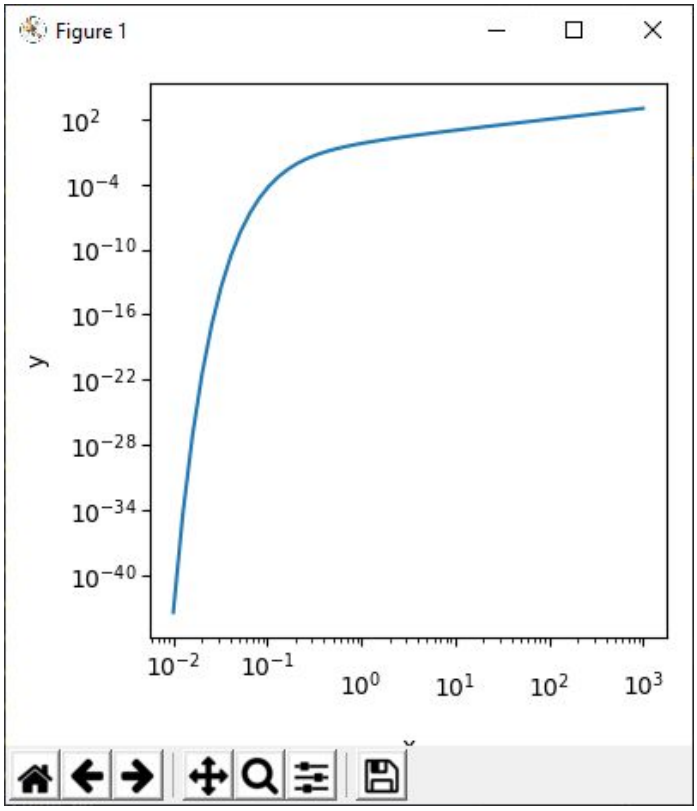


Figure 1
10^{2}
10^{-4}
10^{-10}
10^{-16}
10^{-22}
10^{-28}
10^{-34}
10^{-40}
y
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}

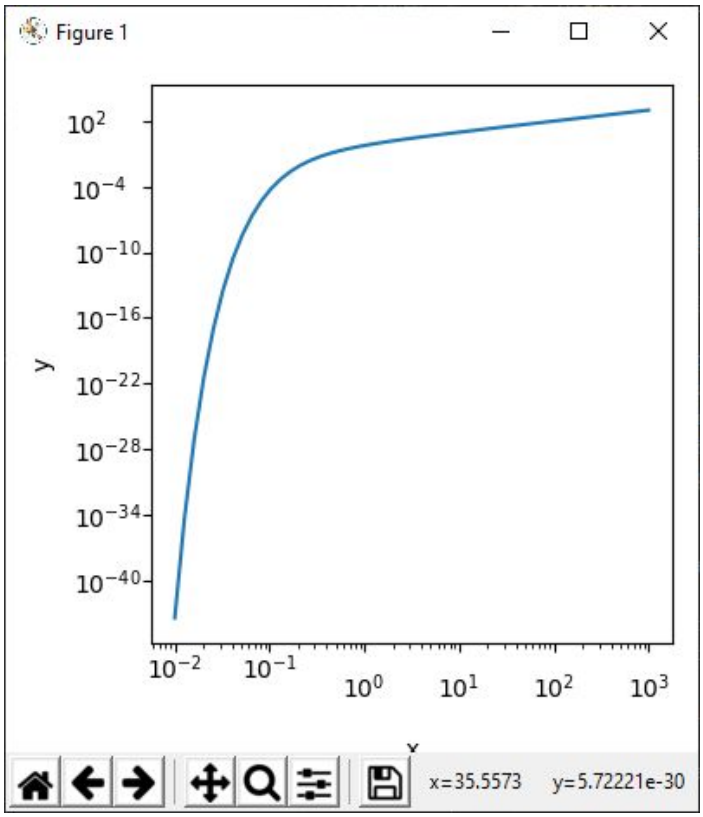
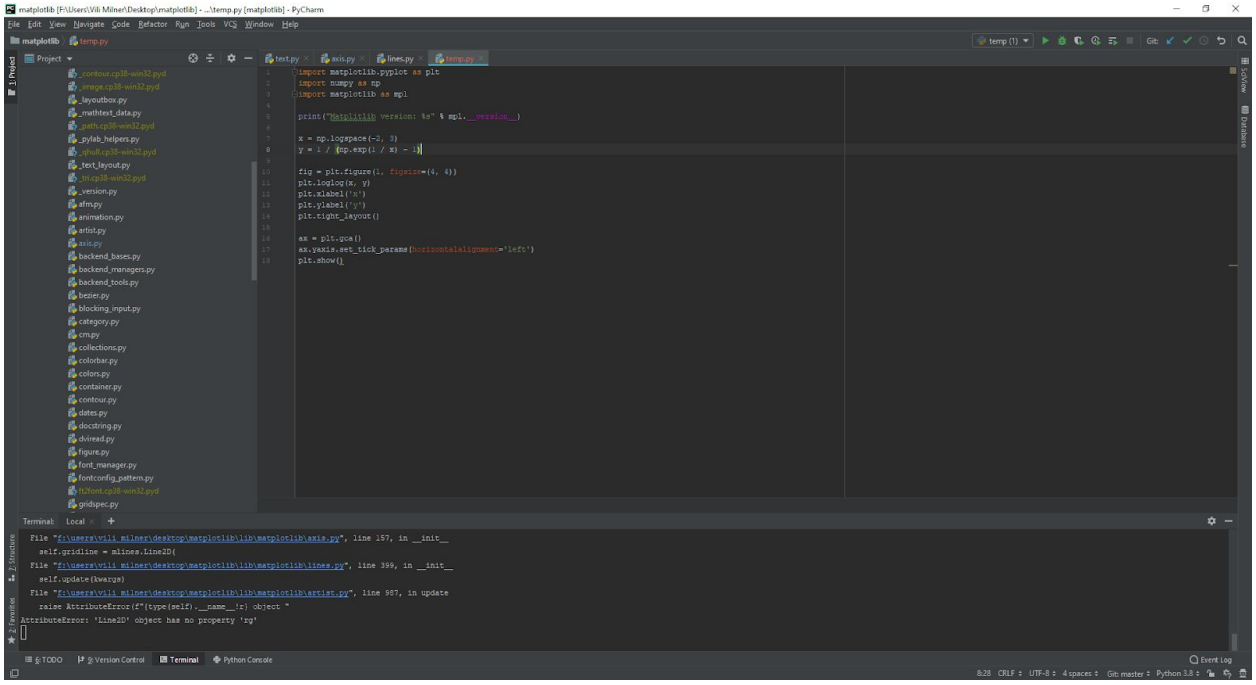
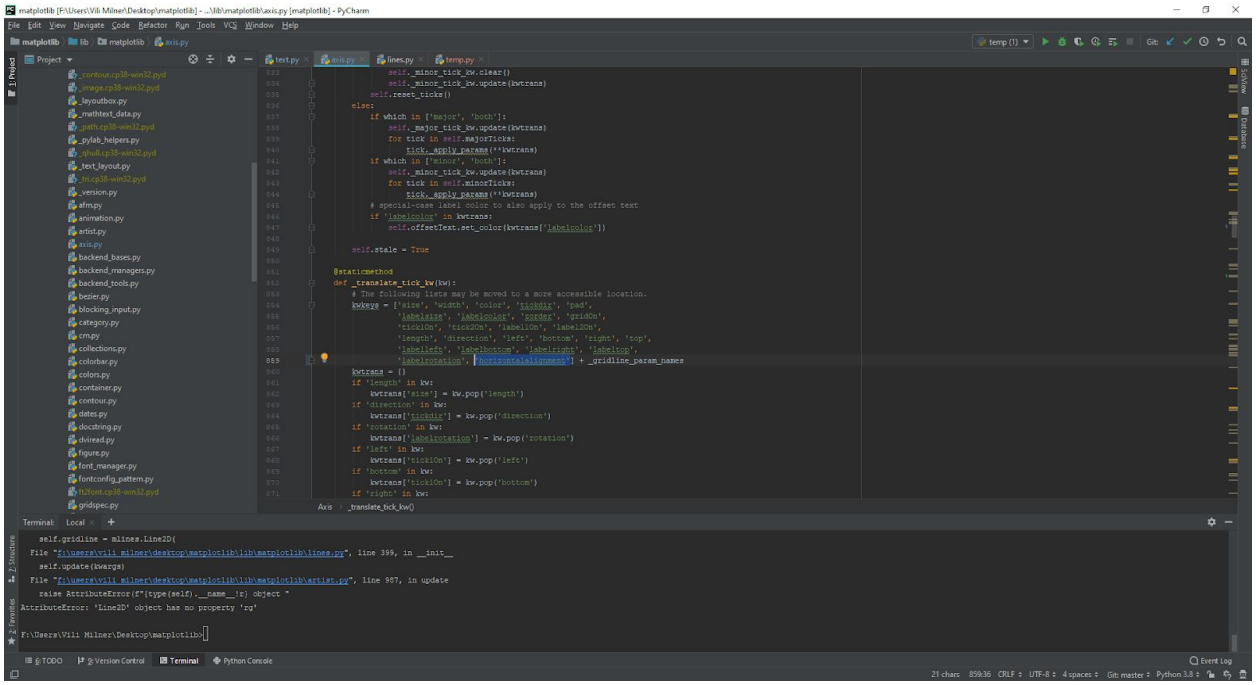
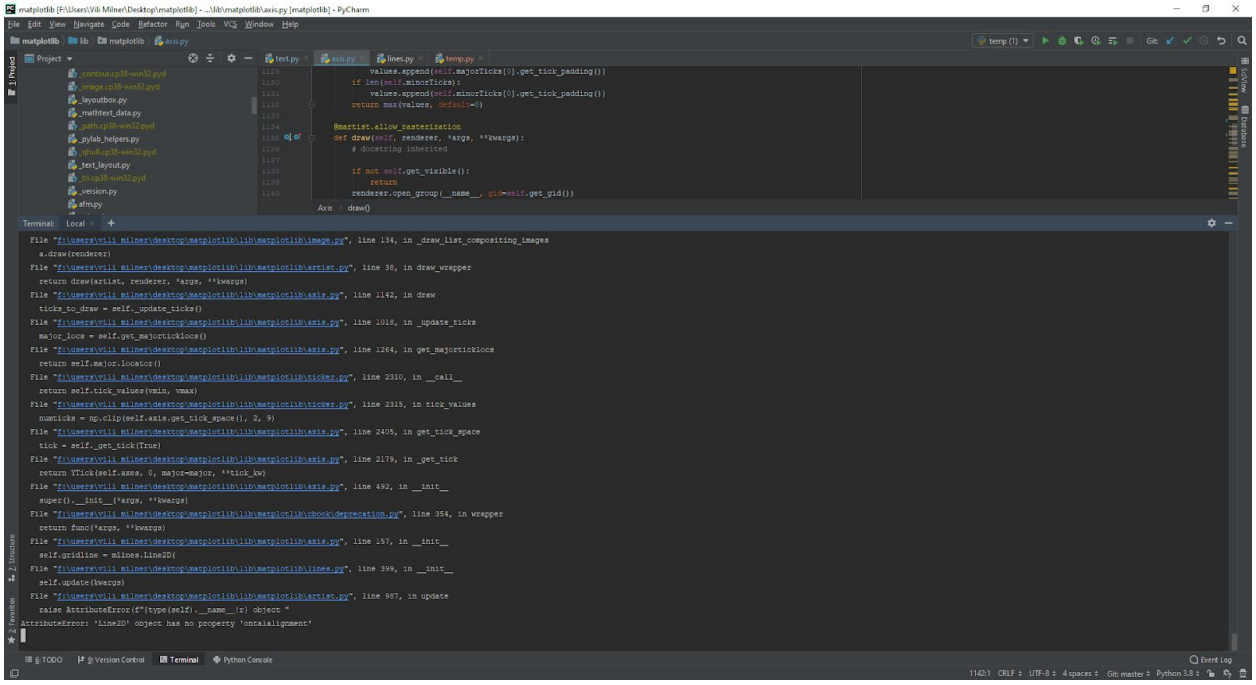


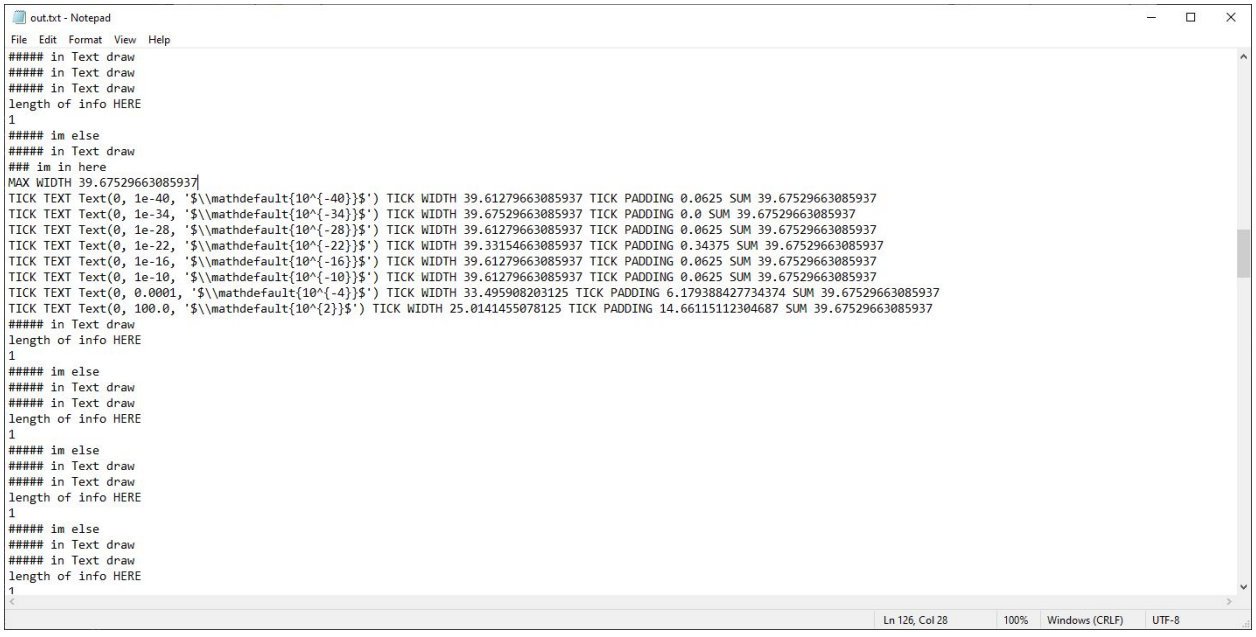
Figure 1
10^{2}
10^{-4}
10^{-10}
10^{-16}
10^{-22}
10^{-28}
10^{-34}
10^{-40}
y
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
x=35.5573
y=5.72221e-30

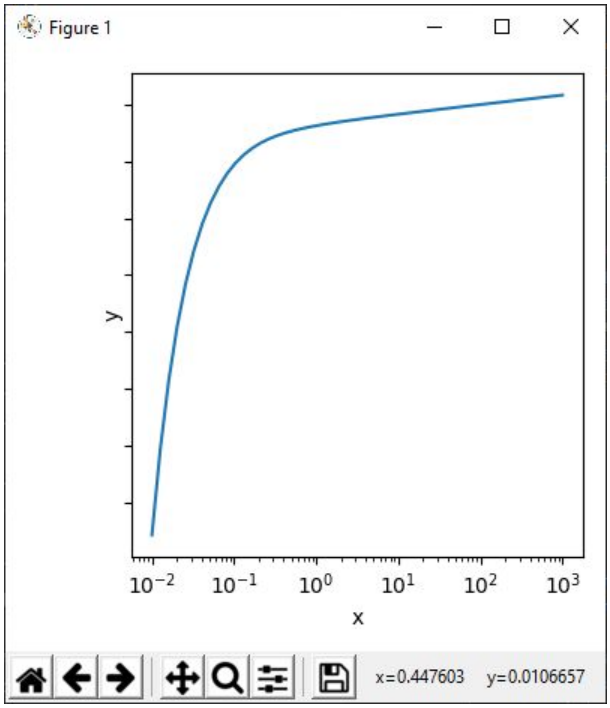
Attempt 2:


matplotlib [F:\Users\Vili Milner\Desktop\matplotlib] - ...\temp.py [matplotlib] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
import matplotlib.pyplot as plt
import numpy as np
import matplotlib as mpl
print("Matplotlib version: %s" % mpl.__version__)
x = np.logspace(-2, 3)
y = 1 / (np.exp(1 / x) - 1)
fig = plt.figure(1, figsize=(4, 4))
plt.loglog(x, y)
plt.xlabel('x')
plt.ylabel('y')
plt.tight_layout()
ax = plt.gca()
ax.yaxis.set_tick_params(horizontalalignment='left')
plt.show()
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\axis.py", line 157, in __init__
self.gridline = mlines.Line2D(
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\lines.py", line 399, in __init__
self.update(kwargs)
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\artist.py", line 987, in update
raise AttributeError(f"{type(self).__name__!r} object "
AttributeError: 'Line2D' object has no property 'rg'


matplotlib [F:\Users\Vili Milner\Desktop\matplotlib] - ...\lib\matplotlib\axis.py [matplotlib] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
self._minor_tick_kw.clear()
self._minor_tick_kw.update(kwtrans)
self.reset_ticks()
else:
if which in ['major', 'both']:
self._major_tick_kw.update(kwtrans)
for tick in self.majorTicks:
tick._apply_params(**kwtrans)
if which in ['minor', 'both']:
self._minor_tick_kw.update(kwtrans)
for tick in self.minorTicks:
tick._apply_params(**kwtrans)
# special-case label color to also apply to the offset text
if 'labelcolor' in kwtrans:
self.offsetText.set_color(kwtrans['labelcolor'])
self.stale = True
@staticmethod
def _translate_tick_kw(kw):
# The following lists may be moved to a more accessible location.
kwkeys = ['size', 'width', 'color', 'tickdir', 'pad',
'labelsize', 'labelcolor', 'zorder', 'gridOn',
'tick1On', 'tick2On', 'label1On', 'label2On',
'length', 'direction', 'left', 'bottom', 'right', 'top',
'labelleft', 'labelbottom', 'labelright', 'labeltop',
'labelrotation', 'horizontalalignment'] + _gridline_param_names
kwtrans = {}
if 'length' in kw:
kwtrans['size'] = kw.pop('length')
if 'direction' in kw:
kwtrans['tickdir'] = kw.pop('direction')
if 'rotation' in kw:
kwtrans['labelrotation'] = kw.pop('rotation')
if 'left' in kw:
kwtrans['tick1On'] = kw.pop('left')
if 'bottom' in kw:
kwtrans['tick1On'] = kw.pop('bottom')
if 'right' in kw:
Axis _translate_tick_kw()
self.gridline = mlines.Line2D(
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\lines.py", line 399, in __init__
self.update(kwargs)
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\artist.py", line 987, in update
raise AttributeError(f"{type(self).__name__!r} object "
AttributeError: 'Line2D' object has no property 'rg'
F:\Users\Vili Milner\Desktop\matplotlib>


matplotlib [F:\Users\Vili Milner\Desktop\matplotlib] - ...\lib\matplotlib\axis.py [matplotlib] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
Project
text.py
axis.py
lines.py
temp.py
Terminal: Local
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\image.py", line 134, in _draw_list_compositing_images
a.draw(renderer)
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\artist.py", line 38, in draw_wrapper
return draw(artist, renderer, *args, **kwargs)
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\axis.py", line 1142, in draw
ticks_to_draw = self._update_ticks()
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\axis.py", line 1018, in _update_ticks
major_locs = self.get_majorticklocs()
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\axis.py", line 1264, in get_majorticklocs
return self.major.locator()
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\ticker.py", line 2310, in __call__
return self.tick_values(vmin, vmax)
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\ticker.py", line 2315, in tick_values
numticks = np.clip(self.axis.get_tick_space(), 2, 9)
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\axis.py", line 2405, in get_tick_space
tick = self._get_tick(True)
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\axis.py", line 2179, in _get_tick
return YTick(self.axes, 0, major=major, **tick_kw)
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\axis.py", line 492, in __init__
super().__init__(*args, **kwargs)
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\cbook\deprecation.py", line 354, in wrapper
return func(*args, **kwargs)
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\axis.py", line 157, in __init__
self.gridline = mlines.Line2D(
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\lines.py", line 399, in __init__
self.update(kwargs)
File "f:\users\vili milner\desktop\matplotlib\lib\matplotlib\artist.py", line 987, in update
raise AttributeError(f"{type(self).__name__!r} object "
AttributeError: 'Line2D' object has no property 'ontalalignment'
TODO
Version Control
Terminal
Python Console
Event Log

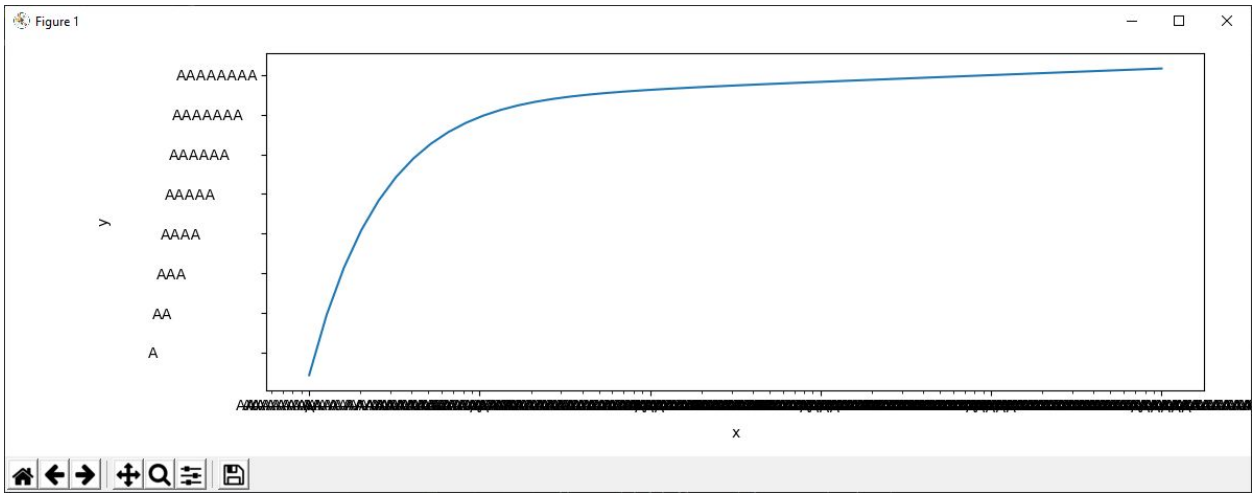

out.txt - Notepad
File Edit Format View Help
in Text draw
in Text draw
in Text draw
length of info HERE
1
im else
in Text draw
im in here
MAX WIDTH 39.67529663085937
TICK TEXT Text(0, 1e-40, '$\\mathdefault{10^{-40}}$') TICK WIDTH 39.61279663085937 TICK PADDING 0.0625 SUM 39.67529663085937
TICK TEXT Text(0, 1e-34, '$\\mathdefault{10^{-34}}$') TICK WIDTH 39.67529663085937 TICK PADDING 0.0 SUM 39.67529663085937
TICK TEXT Text(0, 1e-28, '$\\mathdefault{10^{-28}}$') TICK WIDTH 39.61279663085937 TICK PADDING 0.0625 SUM 39.67529663085937
TICK TEXT Text(0, 1e-22, '$\\mathdefault{10^{-22}}$') TICK WIDTH 39.33154663085937 TICK PADDING 0.34375 SUM 39.67529663085937
TICK TEXT Text(0, 1e-16, '$\\mathdefault{10^{-16}}$') TICK WIDTH 39.61279663085937 TICK PADDING 0.0625 SUM 39.67529663085937
TICK TEXT Text(0, 1e-10, '$\\mathdefault{10^{-10}}$') TICK WIDTH 39.61279663085937 TICK PADDING 0.0625 SUM 39.67529663085937
TICK TEXT Text(0, 0.0001, '$\\mathdefault{10^{-4}}$') TICK WIDTH 33.495908203125 TICK PADDING 6.179388427734374 SUM 39.67529663085937
TICK TEXT Text(0, 100.0, '$\\mathdefault{10^{2}}$') TICK WIDTH 25.0141455078125 TICK PADDING 14.66115112304687 SUM 39.67529663085937
in Text draw
length of info HERE
1
im else
in Text draw
in Text draw
length of info HERE
1
im else
in Text draw
in Text draw
length of info HERE
1
im else
in Text draw
in Text draw
length of info HERE
1
Ln 126, Col 28
100%
Windows (CRLF)
UTF-8



- Looks like they're aligned

Current solution applies more padding than necessary for the smaller ones and not enough for the larger ones

- **Problem may be the padding that we're passing into it**
- **Maybe one of them is in pixels and the other is not**
- **Stack Overflow (https://stackoverflow.com/questions/5320205/matplotlib-text-dimensions): there's inconsistencies**



In x tick and y tick, padding stored in base pad

Have to understand what we’re getting from the padding and the b box