

How to organize a Python script ...

- Executable script under LINUX:

`#!/usr/bin/env python`

#takes python version in path

`#!/usr/bin/python`

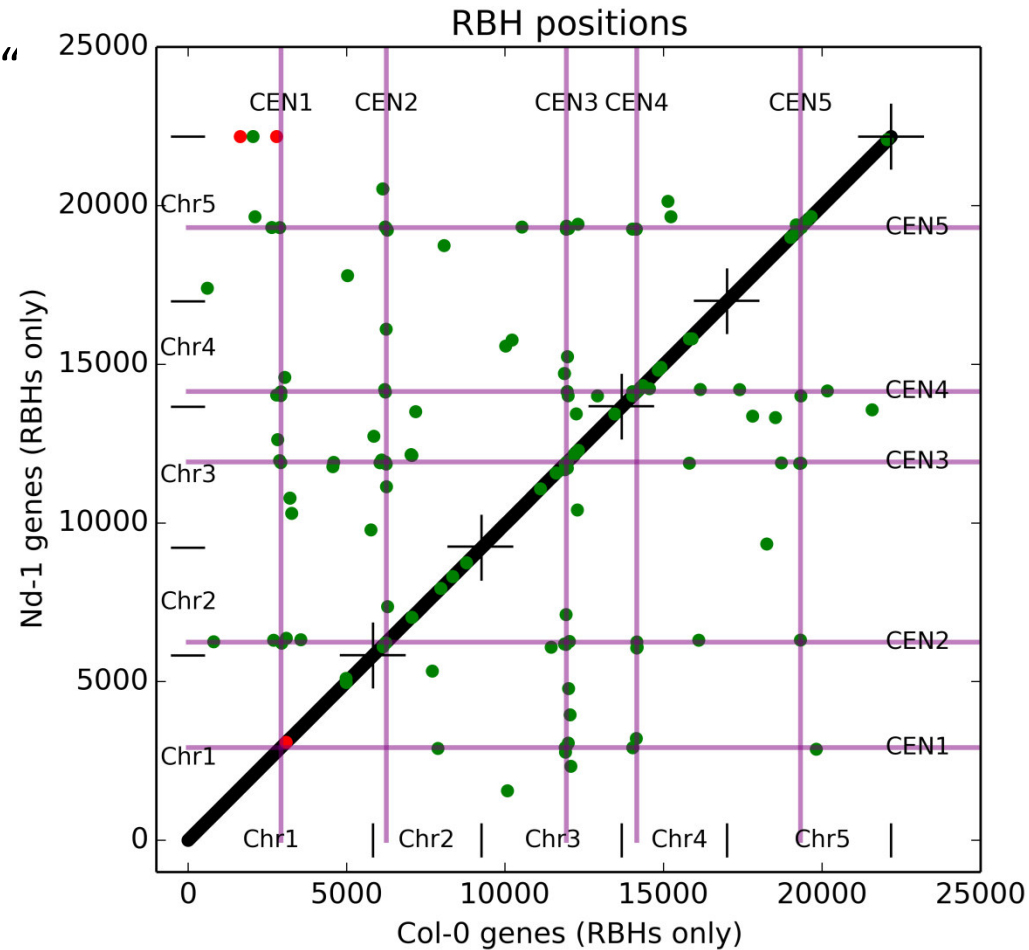
#specifies a specific python version

- Author
- Version
- Imports
- Usage

```
1  ### Boas Pucker ###
2  ### bpucker@cebitec.uni-bielefeld.de ###
3  ### v0.2 ###
4
5  __usage__ = """
6      python construct_RNA_seq_coverage_file.py\n
7      --in <BAM_FILE>\n
8      --out <OUTPUT_FILE>\n
9
10     --bam_is_sorted <PREVENTS_EXTRA_SORTING_OF_BAM_FILE>\n
11
12     feature requests and bug reports: bpucker@cebitec.uni-bielefeld.de
13     """
14
15  __cite__ = """ Pucker & Brockington, 2018: https://doi.org/10.1186/s12864-018-5360-z """
16
17
18  import os, sys
19
20  # --- end of imports --- #
21
22  def main( arguments ):
```

matplotlib

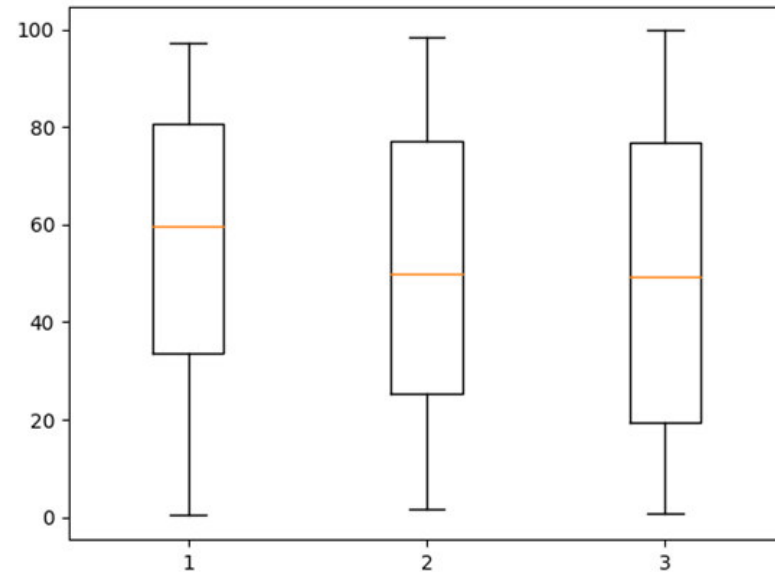
- „import matplotlib.pyplot as plt“
- Visualization of complex data
- Automatic generation of plots
- Amazing customization options



(Pucker *et al.*, 2016)

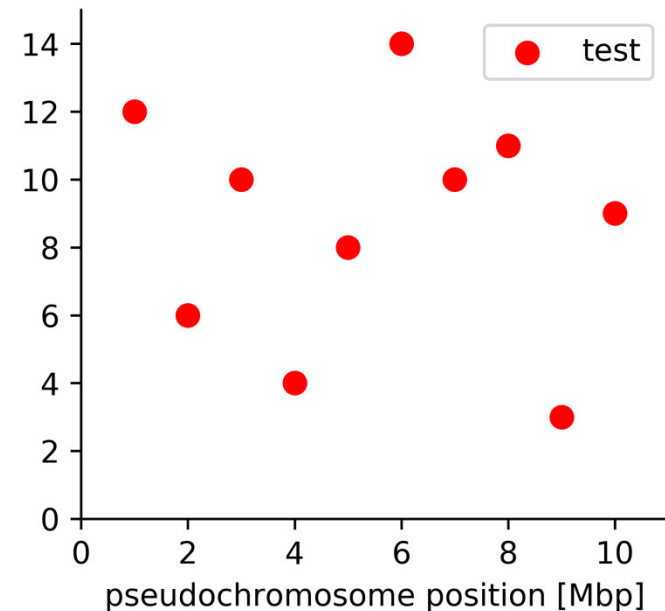
Box plot

```
1 import matplotlib.pyplot as plt
2 import numpy as np
3
4 d1 = np.random.rand(50) * 100 #generate random numbers
5 d2 = np.random.rand(50) * 100
6 d3 = np.random.rand(50) * 100
7
8 data = [d1, d2, d3] # multiple box plots on one figure
9
10 plt.figure()
11 plt.boxplot(data)
12 plt.show()
```



Scatter plot

```
1 import matplotlib.pyplot as plt
2
3 fig, ax = plt.subplots( figsize=( 10, 4 ) ) #defining size of plot
4
5 x_values = [ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 ]
6 y_values = [ 12, 6, 10, 4, 8, 15, 10, 11, 3, 9 ]
7
8 ax.scatter( x_values, y_values, color="red", s=10, marker="o", label="test" )
9 #setting color, marker size, marker shape and label of this group
10
11 ax.legend( numpoints=1 )
12 #each group is represented by only one marker in the legend (default=3)
13
14 ax.set_xlim( 0, 11 ) #set range of x-axis
15 ax.set_ylim( 0, 15 ) #set range of y-axis
16
17 ax.set_xlabel( "pseudochromosome position [Mbp]" )
18
19 ax.spines["top"].set_visible(False) #remove lines and ticks
20 ax.spines["right"].set_visible(False) #remove lines and ticks
21
22 plt.subplots_adjust(left=0.05, right=0.99, top=0.97, bottom=0.12)
23 #adjust size of plot within figure
24
25 plt.show()
26 fig.savefig( "my_plot.png", dpi=600 ) #write figure into output file
27 plt.close( "all" ) #destroy created figures (cleaning up)
```

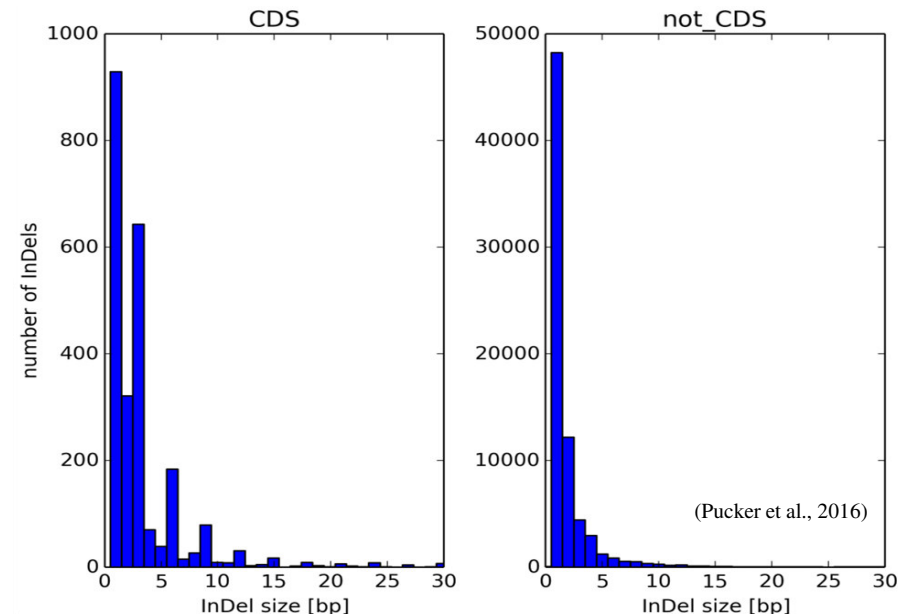


Histogram

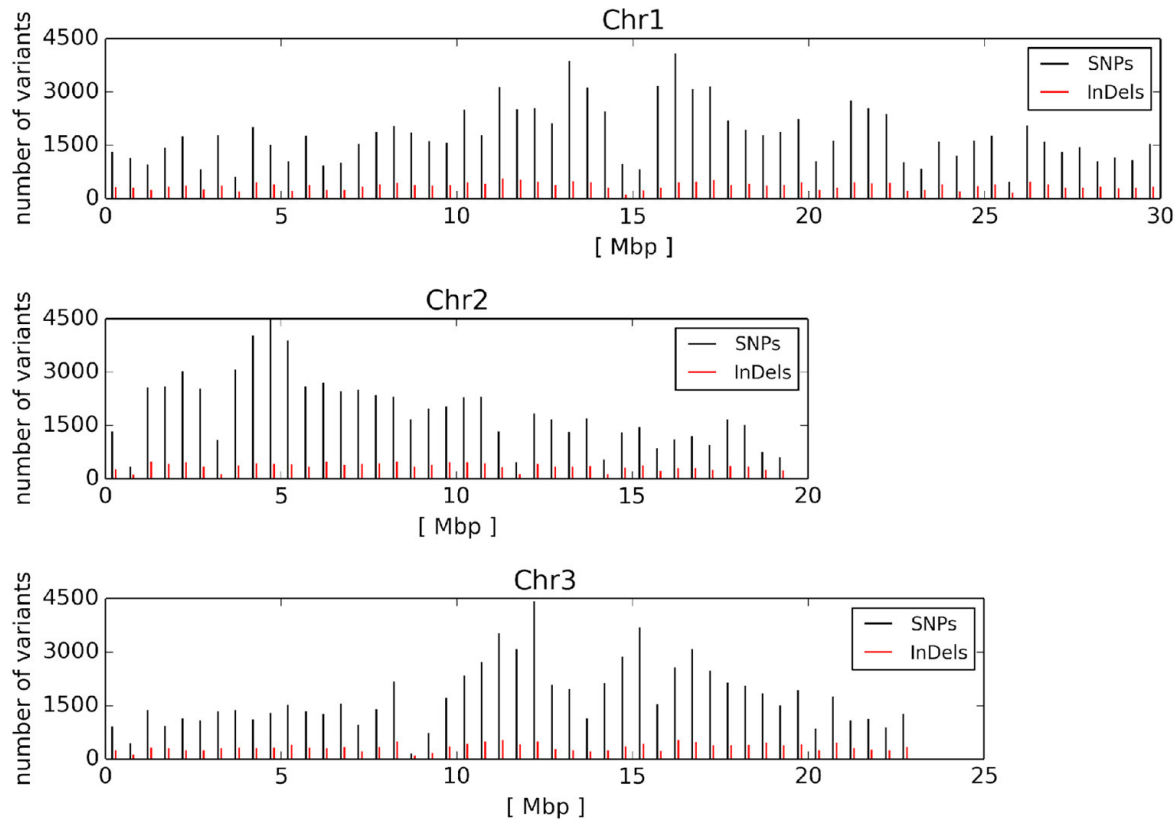
```

1 import matplotlib.pyplot as plt
2
3 # --- end of imports --- #
4
5 gene_space = [ 3, 3, 6, 6, 9, 9, 12, 3, 3, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
6               11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
7               12, 15, 18, 21, 24, 27, 30 ]
8 intergenic = [ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 1, 2, 3, 4, 5, 6, 7, 8, 9,
9               1, 2, 3, 4, 5, 6, 7, 1, 2, 3, 4, 1, 2, 1 ]
10
11
12 fig, ( ax1, ax2 ) = plt.subplots( 1, 2, sharey=False)
13 counts, bins, patches = ax1.hist( gene_space, bins=max( gene_space ), align="left" )
14 ax1.set_title( "CDS" )
15 ax1.set_xlim( 0, 30 )
16 ax1.set_xlabel( "InDel size [bp]" )
17 ax1.set_ylabel( "number of InDels" )
18
19 counts, bins, patches = ax2.hist( intergenic, bins=max( intergenic ), align="left" )
20 ax2.set_title( "not_CDS" )
21 ax2.set_xlim( 0, 30 )
22 ax2.set_xlabel( "InDel size [bp]" )
23 plt.subplots_adjust( wspace=0.3 ) #increase space between figures
24
25 plt.show()
26 fig.savefig( prefix + "InDel_size_distribution.png", dpi=300 )
27 plt.close('all')

```



'barplot' figure



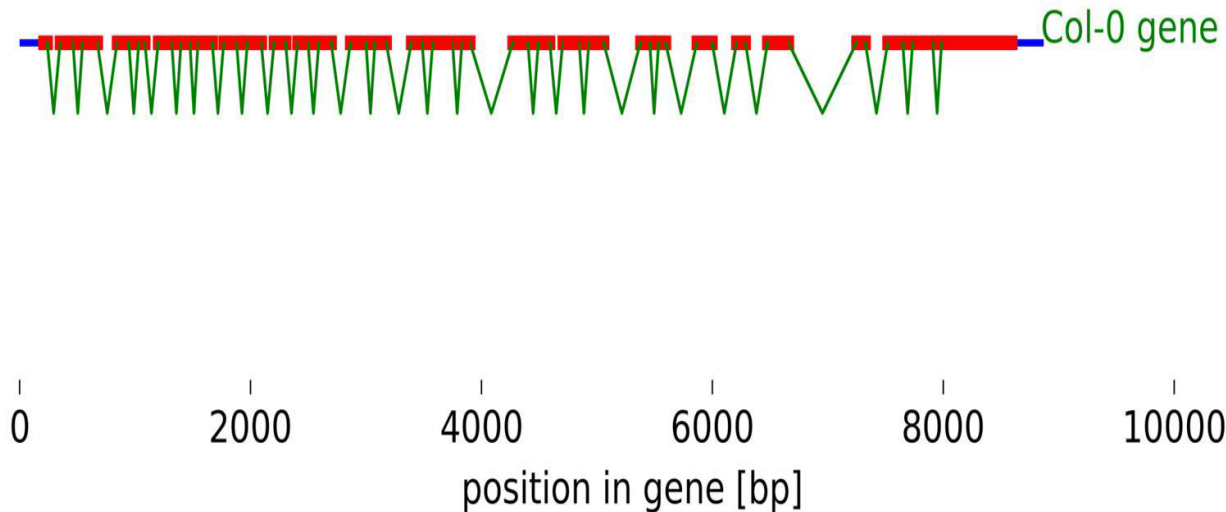
barplots.py generates
barplots at specific
positions by drawing a
normal line

(script is available in
course repository)

(Pucker et al., 2016)

Hanna Schilbert, Katharina Sielemann, Bianca Frommer, Daniela Holtgräwe -
Python Programming for Life Scientists

Gene structure plot

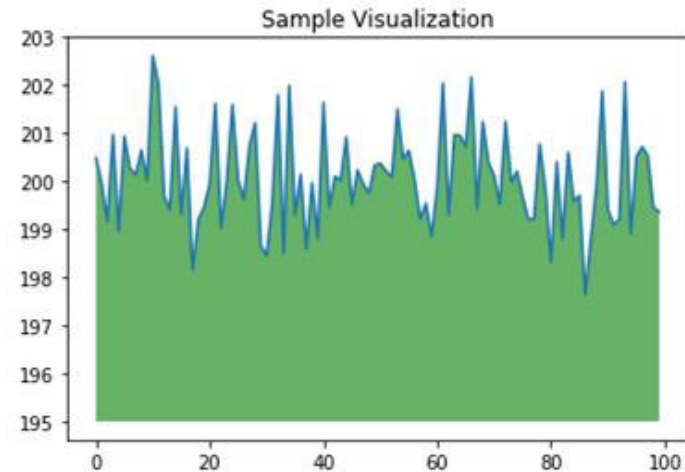


(Pucker et al., 2017)

`gene_structure_plot.py` generates visualizations
of gene/transcript structures based on GFF
annotations

Human dataset - Exercise 2

- Input: human_genes_alignment_identities.txt
- Visualize the data!



```
[ ] import numpy as np
    from matplotlib import pyplot as plt

    ys = 200 + np.random.randn(100)
    x = [x for x in range(len(ys))]

    plt.plot(x, ys, '-')
    plt.fill_between(x, ys, 195, where=(ys > 195), facecolor='g', alpha=0.6)

    plt.title("Sample Visualization")
    plt.show()
```