

Prof. Dr. Boas Pucker

Python - Application examples

Availability of slides

- All materials are freely available (CC BY) - after the lectures:
 - GitHub: <https://github.com/bpucker/teaching/tree/master/WBIO-A-07>
- Questions: Feel free to ask at any time
- Feedback, comments, or questions: [pucker\[a\]uni-bonn.de](mailto:pucker[a]uni-bonn.de)



My figures and content can be re-used in accordance with CC BY 4.0, but this might not apply to all images/logos.

Reverse complement of nucleotide sequence

What happens here?

```
1 def revcomp( seq ):           Sequence of bases e.g. ATGACATGA
2
3     seq = seq.lower()          Converts input to lower case: atgacatga
4
5     #key:value (=dictionary)
6     complement = { 'a':'t', 't':'a', 'c':'g', 'g':'c' }
7
8     new_seq = []                Get complement for each base
9
10    for nt in seq:
11        new_seq.append( complement[ nt ] )
12
13    #list[::-1] inverts list (last element becomes first)
14    new_seq = ''.join( new_seq[::-1] )
15
16    return new_seq               Inverts list (=reverse)
```

How to use dictionaries

- Values are accessible via keys
- Keys need to be unique
- Quick access to data based on key
- Higher memory occupation than lists/strings

```
my_dict = {"k1": "v1", "k2": {"x1": "y1"}, 5: ["one", "two", "three"], "hello world": "hello world" }
print(my_dict.keys()) #all keys of a dictionary
print(my_dict.values()) #all values of a dictionary
print(my_dict["k1"])
print(my_dict["k2"]["x1"])

dict_keys(['k1', 'k2', 5, 'hello world'])
dict_values(['v1', {'x1': 'y1'}, ['one', 'two', 'three'], 'hello world'])
v1
y1
```

Exercises - Part 6a

- 6.1) Write a function to get the reverse complement (upper case letters) of a DNA sequence given in upper case letters!
- 6.2) Write a function to translate a DNA sequence into amino acids (first frame only)!

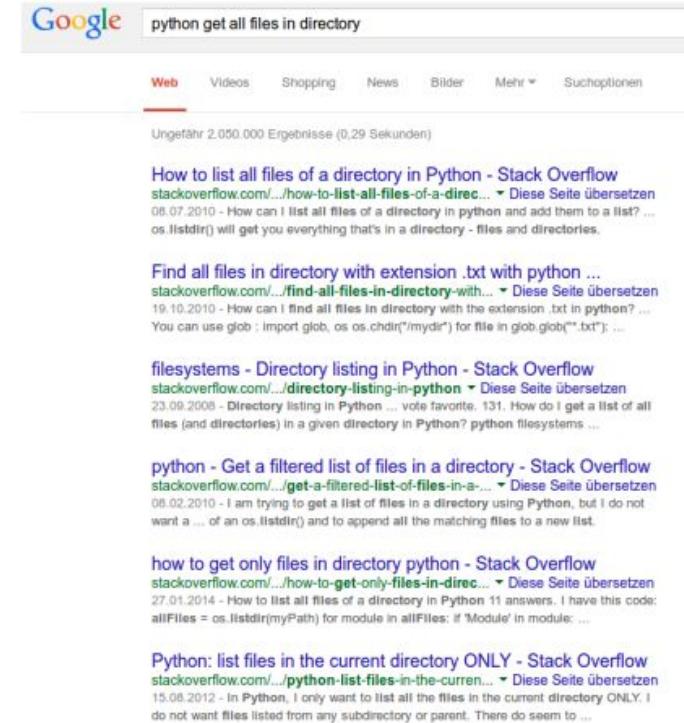
- 6.X1) Write a function to translate DNA sequences in all 6 frames into peptide sequences! The longest peptide sequence per DNA sequence should be returned!
- 6.X2) Write a function to grep a sequence from a FASTA file based on the name of this sequence!

How to approach a bioinfo challenge?

- Identify the problem that needs to be solved
- Split the problem into smallest possible parts
- Solutions for small parts of the problem might be available already
- Precise description of the problem is required for online search
- Best hit often leads to stack overflow:
 - Problem is described by the asking person
 - Multiple solutions are suggested by the community
 - Community votes to identify the best solution
 - Green marking highlights the answer that solved the problem

Example

- Problem: get paths of all files in a certain directory
- Search expression: ‘python get all files in directory’



Google python get all files in directory

Web Videos Shopping News Bilder Mehr ▾ Suchoptionen

Ungefähr 2.050.000 Ergebnisse (0,29 Sekunden)

How to list all files of a directory in Python - Stack Overflow
stackoverflow.com.../how-to-list-all-files-of-a-direc... ▾ Diese Seite übersetzen
08.07.2010 - How can I list all files of a directory in python and add them to a list? ...
os.listdir() will get you everything that's in a directory - files and directories.

Find all files in directory with extension .txt with python ...
stackoverflow.com.../find-all-files-in-directory-with... ▾ Diese Seite übersetzen
19.10.2010 - How can I find all files in directory with the extension .txt in python? ...
You can use glob : Import glob, os os.chdir("/mydir") for file in glob.glob("**.txt"); ...

filesystems - Directory listing in Python - Stack Overflow
stackoverflow.com.../directory-listing-in-python ▾ Diese Seite übersetzen
23.09.2008 - Directory listing in Python ... vote favorite. 131. How do i get a list of all files (and directories) in a given directory in Python? python filesystems ...

python - Get a filtered list of files in a directory - Stack Overflow
stackoverflow.com.../get-a-filtered-list-of-files-in-a... ▾ Diese Seite übersetzen
08.02.2010 - I am trying to get a list of files in a directory using Python, but I do not want a ... of an os.listdir() and to append all the matching files to a new list.

how to get only files in directory python - Stack Overflow
stackoverflow.com.../how-to-get-only-files-in-direc... ▾ Diese Seite übersetzen
27.01.2014 - How to list all files of a directory In Python 11 answers. I have this code: allFiles = os.listdir(myPath) for module in allFiles: if 'Module' In module: ...

Python: list files in the current directory ONLY - Stack Overflow
stackoverflow.com.../python-list-files-in-the-curren... ▾ Diese Seite übersetzen
15.08.2012 - In Python, I only want to list all the files in the current directory ONLY. I do not want files listed from any subdirectory or parent. There do seem ...

Best hit on stackoverflow

533 user like
this answer

This answers
solved the
question

14 Answers active oldest votes

You can use `glob`:

```
import glob, os
os.chdir("/mydir")
for file in glob.glob("*.txt"):
    print(file)
```

or simply `os.listdir`:

```
import os
for file in os.listdir("/mydir"):
    if file.endswith(".txt"):
        print(file)
```

or if you want to traverse directory, use `os.walk`:

```
import os
for root, dirs, files in os.walk("/mydir"):
    for file in files:
        if file.endswith(".txt"):
            print(os.path.join(root, file))
```

share Improve this answer edited Apr 22 at 17:44 answered Oct 19 '10 at 1:12

 Tarantula 4,794 ● 3 ● 22 ● 39  ghostdog74 102K ● 17 ● 116 ● 188

1 Using solution #2, How would you create a file or list with that info? – Merlin Oct 19 '10 at 3:46

35 @ghostdog74: In my opinion it would more appropriate to write `for file in f` than for `for files in f` since what is in the variable is a single filename. Even better would be to change the `f` to `files` and then the for loops could become `for file in files`. – martineau Oct 28 '10 at 14:18

18 @computermagyver: No, `file` is not a reserved word, just the name of a predefined function, so it's quite possible to use it as a variable name in your own code. Although it's true that generally one should avoid collisions like that, `file` is a special case because there's hardly ever any need to use it, so it is often consider an exception to that guideline. If you don't want to do that, PEP8 recommends appending a single underscore to such names, i.e. `file_`, which you'd have to agree is still quite readable. – martineau Oct 14 '12 at 19:04

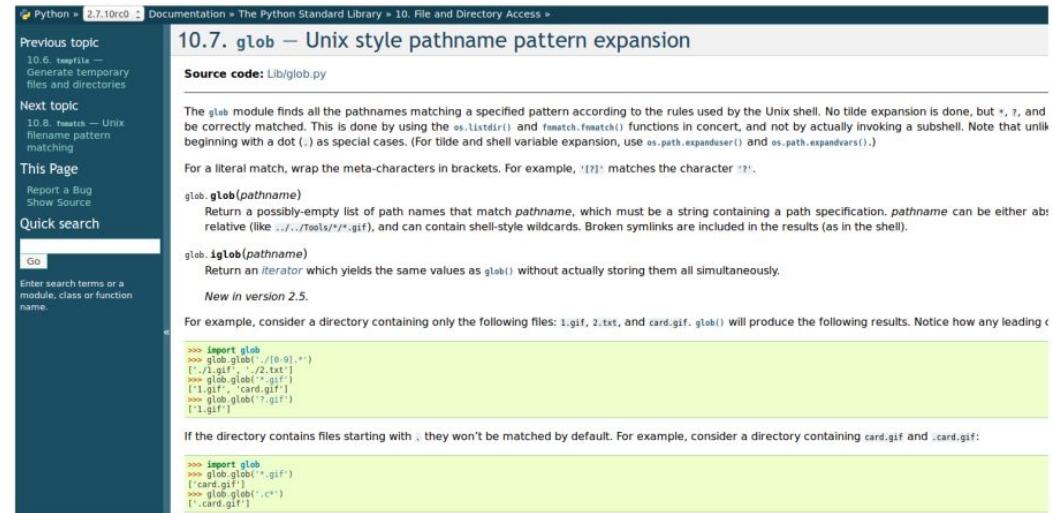
2 Thanks, martineau, you're absolutely right. I jumped too quickly to conclusions. – computermagyver Oct 15 '12 at 19:53

1 Really cool answer, you could replace `r,d,f` by `r_,f` to avoid unused variable declaration. – AsTeR Mar 8 '13 at 20:16

Three different
ways to solve this
problem are
described

Official Python documentation

- Systematic documentation of all functions in a module with all possible arguments
- Sometimes examples are given



The screenshot shows a web browser displaying the Python 2.7.10rc0 documentation for the `glob` module. The URL is `http://docs.python.org/2.7/library/glob.html`. The page title is "10.7. `glob` – Unix style pathname pattern expansion". The left sidebar contains navigation links for "Previous topic" (tempfile), "This Page" (glob), and "Next topic" (fnmatch). Below the sidebar is a "Quick search" input field and a "Go" button. The main content area starts with a "Source code" link to `Lib/glob.py`. It then describes the `glob.glob(pathname)` function, which returns a list of path names matching the specified pattern. It also describes the `glob.iglob(pathname)` function, which returns an iterator for the same purpose. A note indicates that new functionality was added in version 2.5. Below this, there are two code examples. The first example shows how to use `glob.glob` to find files like `1.gif`, `2.txt`, and `card.gif`. The second example shows how to use `glob.glob` to find files starting with a dot, such as `.card.gif`.

```
>>> import glob
>>> glob.glob('/*.*')
['1.gif', '2.txt']
>>> glob.glob('*.*')
['1.gif', 'card.gif']
>>> glob.glob('*.gif')
['1.gif']

If the directory contains files starting with . they won't be matched by default. For example, consider a directory containing card.gif and .card.gif:
```

```
>>> import glob
>>> glob.glob('.*')
['.card.gif']
>>> glob.glob('*.')
['.card.gif']
```

Linux (Ubuntu)



- Ubuntu is an operating system with a graphical user interface
- Excellent environment to perform bioinformatics
- Offers a powerful terminal and comes with comprehensive support
- Processing of large data sets is more efficient via command line tools
- Dual boot system with Windows is possible
- Configuration of USB stick for Ubuntu is possible to explore opportunities

How to run BLAST?

- Running at the NCBI website does not give you full control over all parameters
- Python can be used to run a local search:

```
blastn \
-query <query_file> \
-subject <subject_file> \
-out <output_file> \
-outfmt 6 \
-evalue 0.01 \
-word_size 4
```

1.	qseqid	query (e.g., gene) sequence id
2.	sseqid	subject (e.g., reference genome) sequence id
3.	pident	percentage of identical matches
4.	length	alignment length
5.	mismatch	number of mismatches
6.	gapopen	number of gap openings
7.	qstart	start of alignment in query
8.	qend	end of alignment in query
9.	sstart	start of alignment in subject
10.	send	end of alignment in subject
11.	evalue	expect value
12.	bitscore	bit score

How to execute processes via shell?

- Running shell commands through the subprocess module:

```
p = subprocess.Popen( arg='ls -lh', shell=True )  
p.communicate()
```

- Can be used to run everything via Python

- Python waits until the command is completed

- Example:

```
import subprocess  
p = subprocess.Popen( arg="mkdir test && cd test && ls -lh", shell=True )  
p.communicate()
```

How to process BLAST results?

```
1  def load_BLAST_results( input_file ):
2      """! @brief load all BLAST results from file """
3
4      data = []
5      with open( input_file, "r" ) as f:
6          line = f.readline()
7          while line:
8              parts = line.strip().split('\t')
9              data.append( { 'query': parts[0],
10                  'subject': parts[1],
11                  'query_start': int( parts[6] ),
12                  'query_end': int( parts[7] ),
13                  'score': float( parts[-1] )
14              } )
15          line = f.readline()
16      return data
```

AT1G01010	NdCChr1.g1.t1	100.00	429	0	0	1	429	1	429	0.0	895	
AT1G01010	NdCChr4.g18734.t1	32.84	469	247	15	1	428	2	443	4e-56	194	
AT1G01010	NdCChr1.g127.t1	34.23	336	157	10	1	330	1	278	1e-41	152	
AT1G01010	NdCChr1.g128.t1	32.38	349	157	14	1	331	1	288	4e-39	146	
AT1G01010	NdCChr4.g18730.t1	39.33	178	95	5	1	175	2	169	1e-32	126	
AT1G01010	NdCChr3.g12773.t1	39.63	164	89	2	1	162	1	156	1e-28	115	
AT1G01010	NdCChr4.g22969.t1	40.74	162	79	5	5	159	11	162	3e-28	117	
AT1G01010	NdCChr4.g18733.t1	40.00	165	90	5	1	162	2	160	4e-28	115	
AT1G01010	NdCChr3.g17122.t1	42.31	156	74	6	5	153	15	161	4e-27	112	

Exercises - Part6b

- 6.6) Collect the best CHS BLAST result per contig from the CHS_vs_Digitalis.txt file.
- 6.7) Count the number of BLAST hits that show a similarity >80%, an alignment length >200, and an e-value $<10^{-10}$.

How to organize a Python script

- Make a script recognize that it needs to run with Python:

```
#!/usr/bin/env python3
```

- Other information to include:
 - Author
 - Version
 - Usage
 - Imports

```
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>
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 ):
```

How to pass arguments to a Python script?

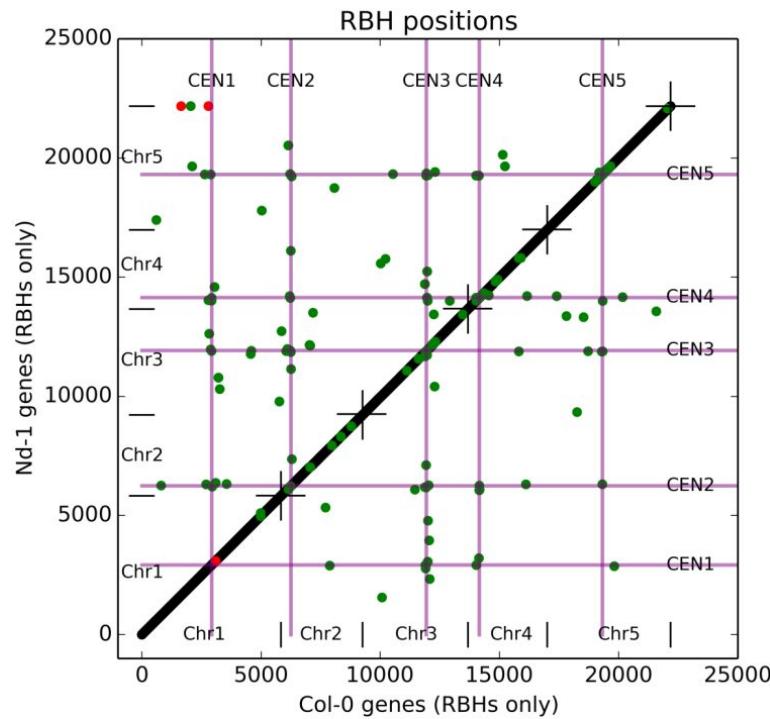
```
1 __usage__ = """ how to run the script and list of arguments """
2
3 def main( arguments ):
4     """! @brief run everything """
5
6     fasta_file = arguments[ arguments.index( '--fasta' )+1 ]
7     gff3_file = arguments[ arguments.index( '--gff3' )+1 ]
8     species = arguments[ arguments.index( '--species' )+1 ]
9     output_dir = arguments[ arguments.index( '--tmp' )+1 ]
10    hints_file = arguments[ arguments.index( '--hints' )+1 ]
11
12    if '--cutoff' in arguments:
13        cutoff = int( arguments[ arguments.index( '--cutoff' )+1 ] )
14    else:
15        cutoff = 1
16
17    #everything happens here
18
19
20    if '--fasta' in sys.argv and '--gff3' in sys.argv and '--species' in sys.argv and '--tmp' in sys.argv and '--hints' in sys.argv:
21        main( sys.argv )
22    else:
23        sys.exit( __usage__ )
```

Command line examples

```
python transeq.py \
--in Eucommia_ulmoides.cds.fasta \
--out Eucommia_ulmoides.pep.fasta
```

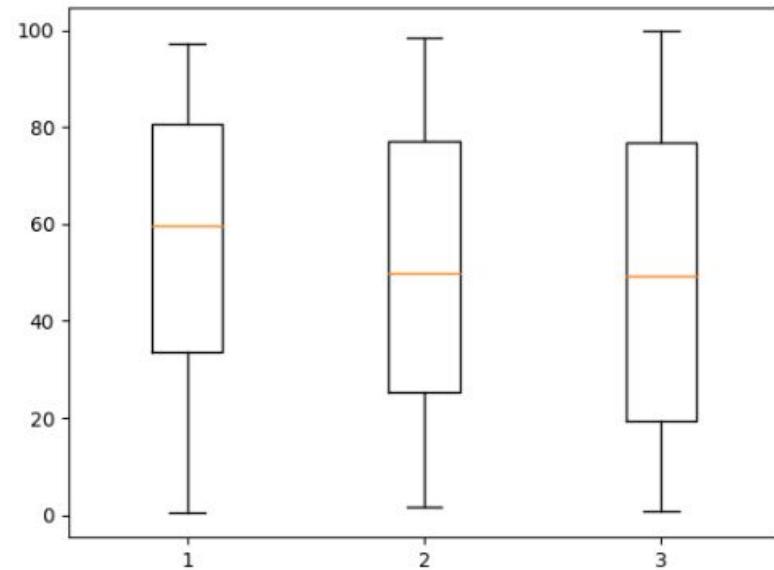
```
python3 construct_anno.py \
--out ./Eucommia_ulmoides_anno/ \
--in ./Eucommia_ulmoides.pep.fasta \
--ref ./Araport11_genes.201606.pep.repr_MOD.fasta \
--anno ./araport11_annotation.updated.txt
```

- Importing matplotlib:
`import matplotlib.pyplot as plt`
- Visualization of complex data
- Automatic generation of plots
- Unlimited customization options



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()
```

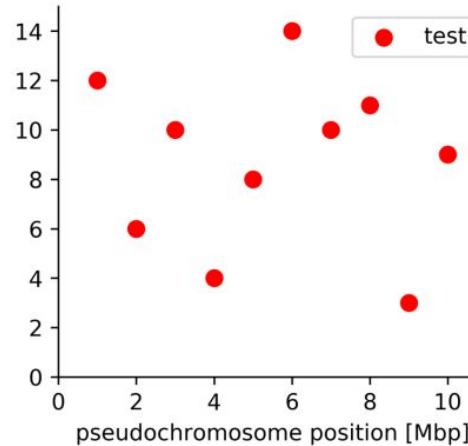


Scatter plot

```

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

```

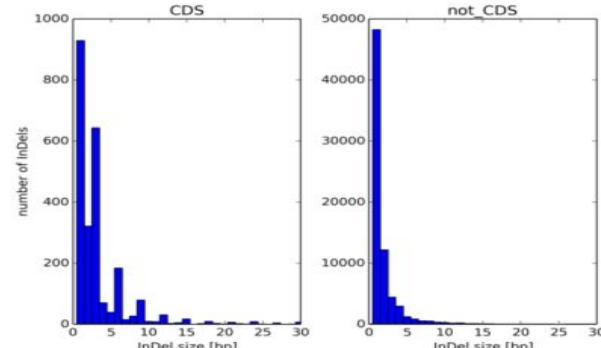


Histogram

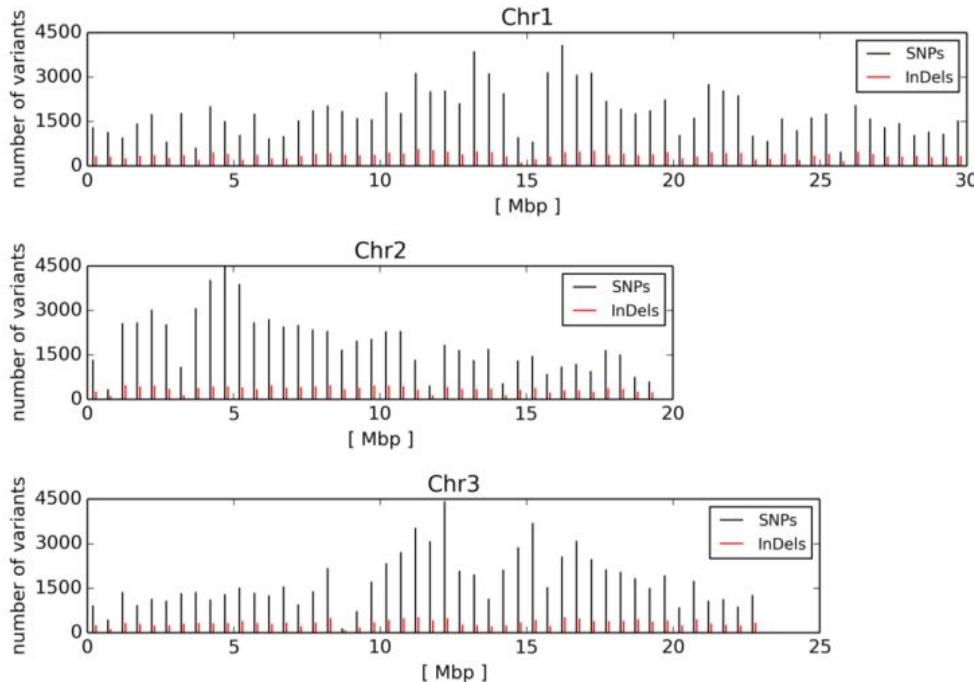
```

1 import matplotlib.pyplot as plt
2 # --- end of imports --- #
3
4 gene_space = [ 3, 3, 6, 6, 9, 9, 12, 3, 3, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
5   11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
6   12, 15, 18, 21, 24, 27, 30 ]
7
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

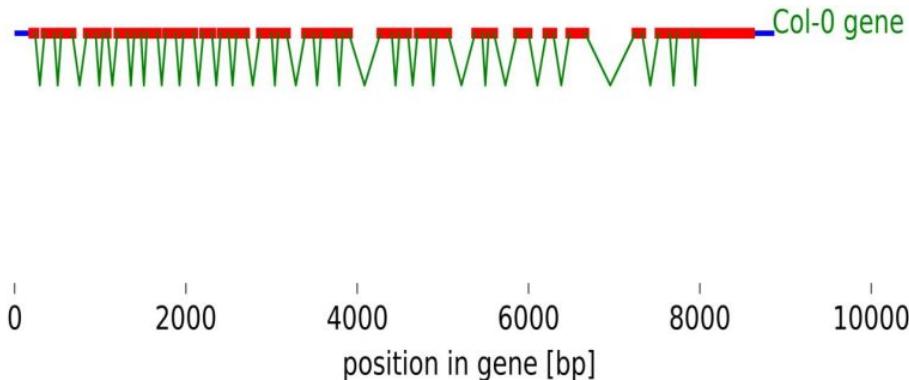


barplots.py generates
barplots at specific
positions by drawing a
normal line

(script is available in
course repository)

Gene structure plot

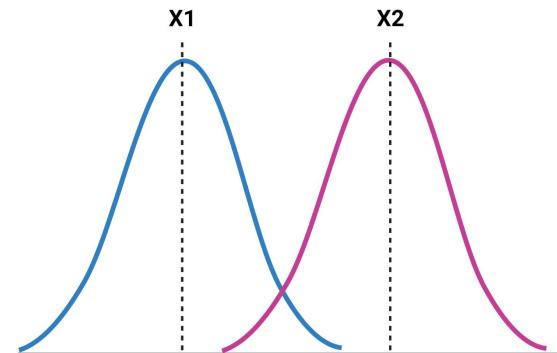
- `gene_structure_plot.py` generates visualizations of gene/transcript structures based on GFF annotations



Exercises - Part6c

- 6.8) Construct a figure to illustrate the order and orientation of genes in the gum gene cluster of *Xanthomonas campestris* pv. *campestris*!
- 6.9) Save this figure in different file formats (png, jpg, pdf, svg)!

- Compare observed sample against the expected distribution
- Check if two samples are derived from same distribution
- H_0 = samples were taken from same distribution
- H_0 can only be rejected or kept due to insufficient evidence against it
- H_0 can NEVER be confirmed



Shapiro-Wilk test

- Testing data set for normal distribution
- Important for decision about potential tests

```
from scipy import stats  
x = [1, 2, 3, 3, 3, 2, 1]  
stats.shapiro(x)
```

Correlation

- Pearson correlation coefficient is suitable for data following a normal distribution
- Spearman correlation coefficient is better if data distribution is unknown

```
from scipy import stats
x = [1,2,3,4,5]
y = [2,4,6,8,10]
r,p = stats.pearsonr(x,y)
r,p = stats.spearmanr(x,y)
```

- Samples need to show normal distribution
- Comparison of one sample against a reference value
- Comparison of two samples:
 - Paired samples (ttest_rel)
 - Unpaired samples (ttest_ind)

```
from scipy import stats
x = [1,2,3,4,5]
y = [4,6,8,10,11]
t,p = stats.ttest_ind(x,y) #independent samples
t,p = stats.ttest_rel(x,y) #paired samples
```

W-test

- Wilcoxon (W) test compares two paired samples
- Normal distribution is not required

```
from scipy import stats  
x = [1,2,3,4,5]  
y = [4,6,8,10,11]  
w,p = stats.wilcoxon(x,y)
```

- Comparison of unpaired samples
- Normal distribution is not required

```
from scipy import stats  
x = [1,2,3,4,5]  
y = [4,6,8,10,11]  
w,p = stats.mannwhitneyu(x,y)
```

Chi square test

- Comparison of an observation against an expectation
- Comparisons of two observations

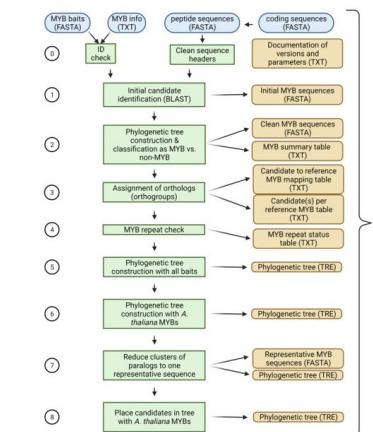
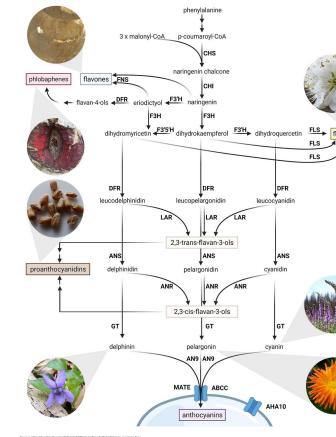
```
from scipy import stats
obs = [1,2,3,4,5]
exp = [4,6,8,10,11]
x, p = stats.chisquare(obs,exp)
```

Exercises - Part6c (UNKNOWN_DATA.ods)

- 6.10) Construct a suitable visualization!
- 6.11) Analyze distribution and trends!
- 6.12) Apply statistical test to investigate difference!

Looking for more Python opportunities?

- Plant Biotechnology and Bioinformatics group is working on:
 - Plant genomics
 - Plant transcriptomics (RNA-seq)
 - Specialized plant metabolites
 - Synthetic biology
 - Big data comparative studies
 - Tool development
 - Data reuse



- Details: <https://www.pbb.uni-bonn.de/>
- Web server: <https://pbb-tools.de/>

Time for questions!

References

- Nd-1 genome assembly (Pucker *et al.*, 2016)
 - <https://doi.org/10.1371/journal.pone.0164321>
- Non-canonical splice sites (Pucker *et al.*, 2017)
 - <https://doi.org/10.1186/s13104-017-2985-y>
- *Croton tiglium* transcriptome assembly (Haak *et al.*, 2018)
 - <https://doi.org/10.3389/fmolb.2018.00062>
- Genome-wide non-canonical splice sites in plants (Pucker & Brockington, 2018)
 - <https://doi.org/10.1186/s12864-018-5360-z>
- Chromosome-level Nd-1 genome assembly (Pucker *et al.*, 2019)
 - <https://doi.org/10.1371/journal.pone.0216233>
- NAVIP (Baasner *et al.*, 2025)
 - <https://doi.org/10.1371/journal.pcbi.1012732>