data_tools Documentation

Release 0.0.1

Nicolas Palacio

May 23, 2018

CONTENTS:

CHAPTER

ONE

MODULE PLOTS

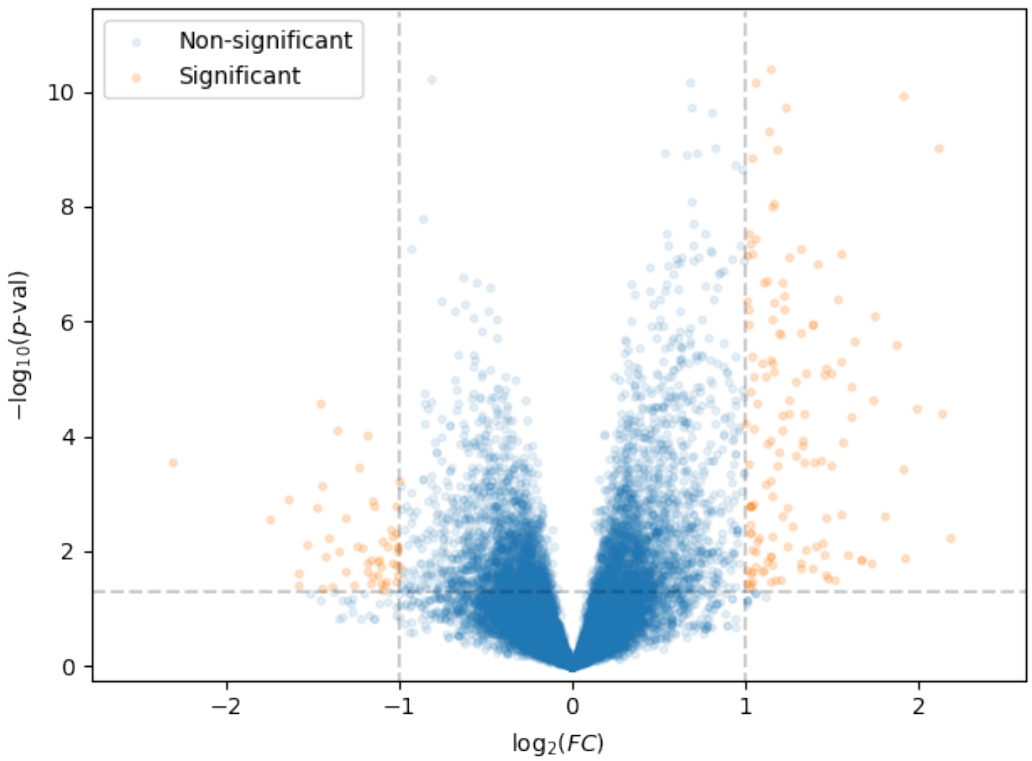
Plotting functions module.

data_tools.plots.**volcano**(*logfc*, *logpval*, *thr_pval=0.05*, *thr_fc=2.0*, *c=('C0', 'C1')*, *legend=True*, *title=None*, *filename=None*, *figsize=None*)

Generates a volcano plot from the differential expression data provided.

- **Arguments:**
  - logfc [list]: Or any iterable type. Contains the log (usually base 2) fold-change values. Must have the same length as logpval.
  - logpval [list]: Or any iterable type. Contains the -log p-values (usually base 10). Must have the same length as logfc.
  - thr_pval [float]: Optional, 0.05 by default. Specifies the p-value (non log-transformed) threshold to consider a measurement as significantly differentially expressed.
  - thr_fc [float]: Optional, 2. by default. Specifies the FC (non log-transformed) threshold to consider a measurement as significantly differentially expressed.
  - c [tuple]: Optional, ('C0', 'C1') by default (matplotlib default colors). Any iterable containing two color arguments tolerated by matplotlib (e.g.: ['r', 'b'] for red and blue). First one is used for non-significant points, second for the significant ones.
  - legend [bool]: Optional, True by default. Indicates wether to show the plot legend or not.
  - title [str]: Optional, None by default. Defines the plot title.
  - filename [str]: Optional, None by default. If passed, indicates the file name or path where to store the figure. Format must be specified (e.g.: .png, .pdf, etc)
  - figsize [tuple]: Optional, None by default (default matplotlib size). Any iterable containing two values denoting the figure size (in inches) as [width, height].
- **Returns:**
  - [*matplotlib.figure.Figure*]: Figure object containing the volcano plot.
- **Examples:**

```
>>> volcano(my_log_fc, my_log_pval)
```


Non-significant
Significant
10
8
6
4
2
0
−2
−1
0
1
2
$-\log_{10}(p\text{-val})$
$\log_2(FC)$

CHAPTER

TWO

DATA_TOOLS.SETS

Set operations module.

data_tools.sets.**in_all**(*x*, *N*)

Checks if a vector x is present in all sets contained in a list N.

- **Arguments:**
  - x [tuple]: Or any hashable type as long as is the same contained in the sets of N.
  - N [list]: Or any iterable type containing [set] objects.
- **Returns:**
  - [bool]: True if x is found in all sets of N, False otherwise.
- **Examples:**

```
>>> N = [{(0, 0), (0, 1)}, # <- set A
...       {(0, 0), (1, 1), (1, 0)}] # <- set B
>>> x = (0, 0)
>>> in_all(x, N)
True
>>> y = (0, 1)
>>> in_all(y, N)
False
```

data_tools.sets.**bit_or**(*a*, *b*)

Returns the bit operation OR between two bit-strings a and b. NOTE: a and b must have the same size.

- **Arguments:**
  - a [tuple]: Or any iterable type.
  - b [tuple]: Or any iterable type.
- **Returns:**
  - [tuple]: OR operation between a and b element-wise.
- **Examples:**

```
>>> a, b = (0, 0, 1), (1, 0, 1)
>>> bit_or(a, b)
(1, 0, 1)
```

data_tools.sets.**multi_union**(*N*)

Returns the union set of all sets contained in a list N.

- **Arguments:**

  - N [list]: Or any iterable type containing [set] objects.

- **Returns:**
  - [set]: The union of all sets contained in N.

- **Examples:**

```
>>> A = {1, 3, 5}
>>> B = {0, 1, 2}
>>> C = {0, 2, 5}
>>> multi_union([A, B, C])
set([0, 1, 2, 3, 5])
```

data_tools.sets.**find_min**(*A*)

Finds and returns the subset of vectors whose sum is minimum from a given set A.

- **Arguments:**
  - A [set]: Set of vectors ([tuple] or any iterable).

- **Returns:**
  - [set]: Subset of vectors in A whose sum is minimum.

- **Examples:**

```
>>> A = {(0, 1, 1), (0, 1, 0), (1, 0, 0), (1, 1, 1)}
>>> find_min(A)
set([(0, 1, 0), (1, 0, 0)])
```

CHAPTER

THREE

DATA_TOOLS.STRINGS

String operations module.

data_tools.strings.**is_numeric**(*s*)

Determines if a string can be considered a numeric value. NaN is also considered, since it is float type.

- **Arguments:**
  - s [str]: String to be evaluated.
- **Returns:**
  - [bool]: True/False depending if the condition is satisfied.
- **Examples:**

```
>>> is_numeric('4')
True
>>> is_numeric('-3.2')
True
>>> is_numeric('number')
False
>>> is_numeric('NaN')
True
```

data_tools.strings.**join_str_lists**(*a*, *b*, *sep=''*)

Joins element-wise two lists (or any 1D iterable) of strings with a given separator (if provided). Length of the input lists must be equal.

- **Arguments:**
  - a [list]: Contains the first elements [str] of the joint strings.
  - b [list]: Contains the second elements [str] of the joint strings.
  - sep [str]: Optional '' (non separated) by default. Determines the separator between the joint strings.
- **Returns:**
  - [list]: List of the joint strings.
- **Example:**

```
>>> a = ['a', 'b']
>>> b = ['1', '2']
>>> join_str_lists(a, b, sep='_')
['a_1', 'b_2']
```

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

d

INDEX