Microarray Gene Expression Analysis with Python

Ahmed Moustafa

October 5, 2023

Table of contents

1 Importing required libraries **2**

2 Loading the data **2**

3 Checking the data behavior **3**
3.1 Transforming 4

4 Exploring the data **5**
4.1 Boxplot 5
4.1.1 Using Matplotlib 5
4.1.2 Using Pandas 7
4.1.3 Using Seaborn 8
4.2 Hierarchical clustering 8

5 Biological signifiance **9**
5.1 Slicing the dataset by condition 9
5.2 Gene-wise mean expression 10
5.3 Scatter plot using Matplotlib 11
5.4 Calculating the fold-change 12
5.5 Scatter plot using Seaborn 13
5.6 Historgram of the fold-change 13
5.6.1 Using Matplotlib 13
5.6.2 Using Seaborn 14

6 Statistical significance **15**
6.1 Calculating t-test p-value 15

6.2 Historgram of the p-value . . . 16
6.2.1 Using Matplotlib . . . 16
6.2.2 Using Seaborn . . . 16

7 Biological & statistical signifiance **17**
7.1 Volcano Plot (p-value vs. fold-change) . . . 17
7.1.1 Using Matplotlib . . . 17
7.1.2 Using Seaborn . . . 18

8 Differentially expressed genes (DEGs) **19**
8.1 Genes with significant fold-change . . . 19
8.2 Genes with significant p-value . . . 19
8.3 Genes with significant fold-change & significant p-value . . . 19
8.4 Heatmap . . . 20

1 Importing required libraries

```
import pandas as pd
import numpy as np
from scipy import stats
from scipy.cluster import hierarchy
import matplotlib.pyplot as plt
import seaborn as sns
```

2 Loading the data

```
data = pd.read_table("https://media.githubusercontent.com/media/ahmedmoustafa/gene-express
data.head()
```

```
/Users/ahmed/Library/Python/3.11/lib/python/site-packages/IPython/core/formatters.py:342: Fu

In future versions `DataFrame.to_latex` is expected to utilise the base implementation of `St
```

ID	KO1	KO2	KO3	WT1	WT2	WT3
1415670_at	6531.0	5562.8	6822.4	7732.1	7191.2	7551.9
1415671_at	11486.3	10542.7	10641.4	10408.2	9484.5	7650.2
1415672_at	14339.2	13526.1	14444.7	12936.6	13841.7	13285.7
1415673_at	3156.8	2219.5	3264.4	2374.2	2201.8	2525.3
1415674_a_at	4002.0	3306.9	3777.0	3760.6	3137.0	2911.5

```
number_of_genes = data.shape[0]
number_of_genes
```

```
45101
```

```
ids = data.index # The ids of the genes are the names of the rows
ids
```

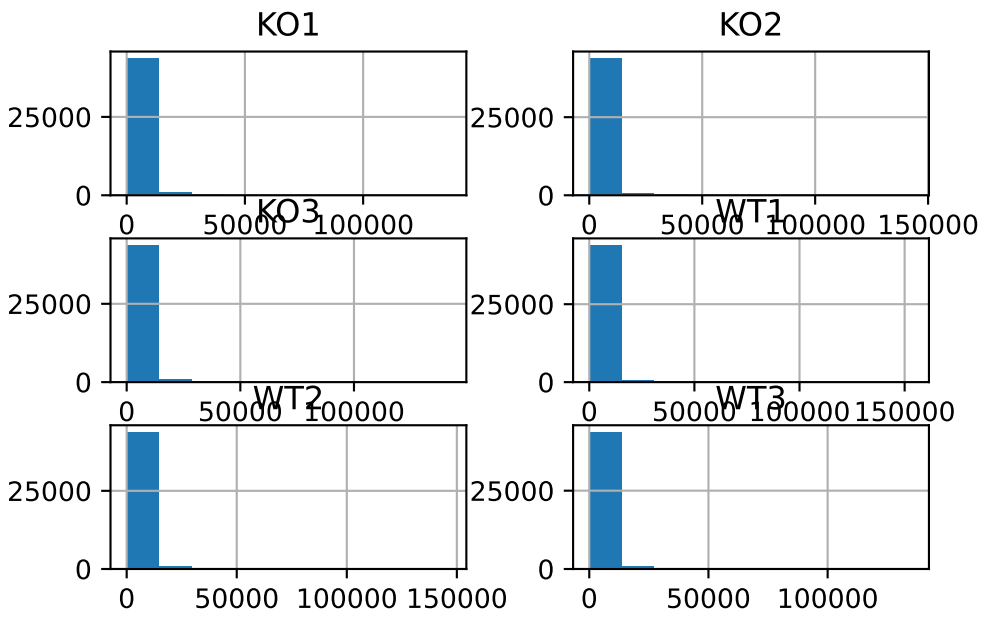
```
Index(['1415670_at', '1415671_at', '1415672_at', '1415673_at', '1415674_a_at',
       '1415675_at', '1415676_a_at', '1415677_at', '1415678_at', '1415679_at',
       ...
       'AFFX-r2-P1-cre-5_at', 'AFFX-ThrX-3_at', 'AFFX-ThrX-5_at',
       'AFFX-ThrX-M_at', 'AFFX-TransRecMur/X57349_3_at',
       'AFFX-TransRecMur/X57349_5_at', 'AFFX-TransRecMur/X57349_M_at',
       'AFFX-TrpnX-3_at', 'AFFX-TrpnX-5_at', 'AFFX-TrpnX-M_at'],
      dtype='object', name='ID', length=45101)
```

3 Checking the data behavior

Check the behavior of the data (e.g., normal?, skewed?)

```
data.hist()
```

```
array([[<Axes: title={'center': 'KO1'}>, <Axes: title={'center': 'KO2'}>],
       [<Axes: title={'center': 'KO3'}>, <Axes: title={'center': 'WT1'}>],
       [<Axes: title={'center': 'WT2'}>, <Axes: title={'center': 'WT3'}>]],
      dtype=object)
```

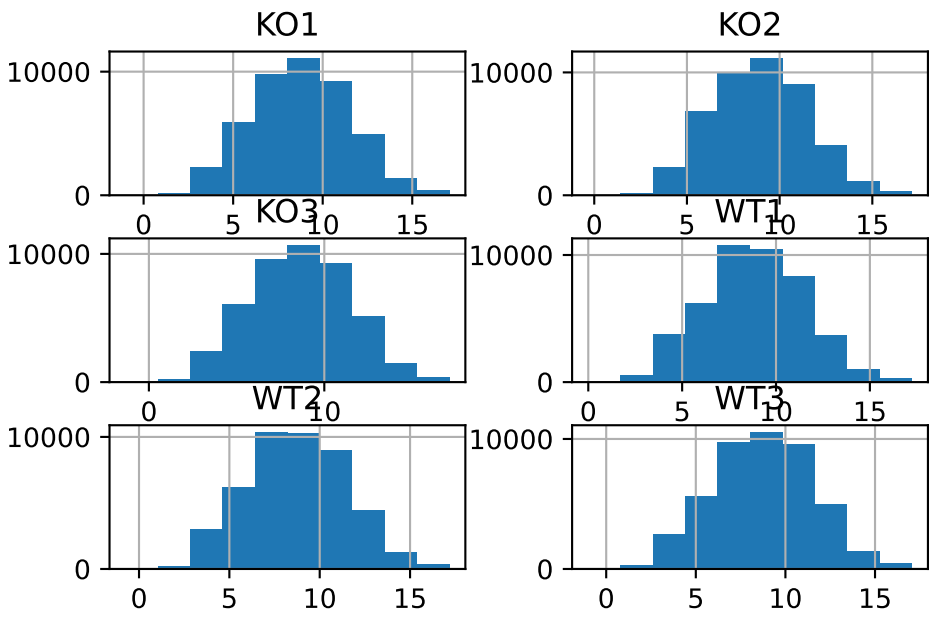


3.1 Transforming

*log*2 transformation

```
data2 = np.log2(data)
data2.head()
data2.hist()
```

```
array([[<Axes: title={'center': 'KO1'}>, <Axes: title={'center': 'KO2'}>],
       [<Axes: title={'center': 'KO3'}>, <Axes: title={'center': 'WT1'}>],
       [<Axes: title={'center': 'WT2'}>, <Axes: title={'center': 'WT3'}>]],
      dtype=object)
```



4 Exploring the data

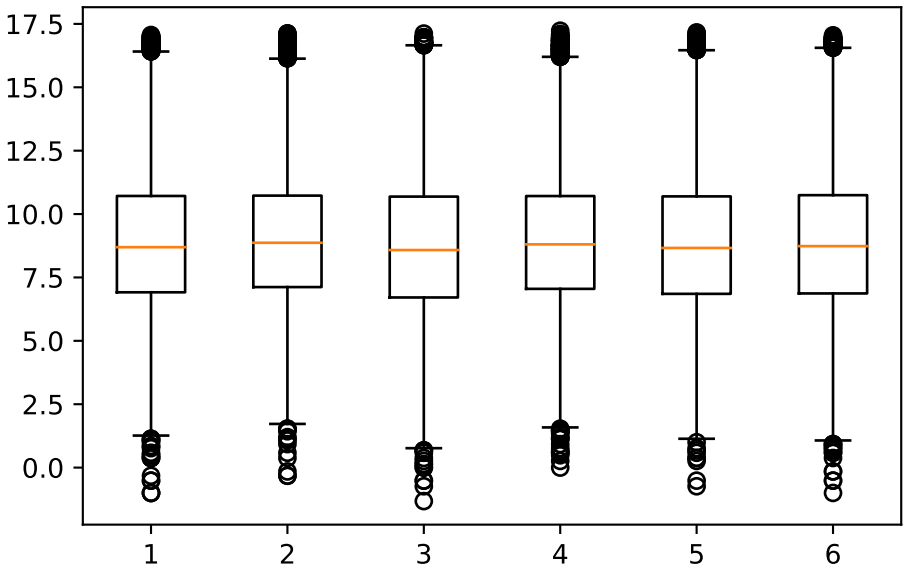
4.1 Boxplot

4.1.1 Using Matplotlib

```
plt.boxplot(data2)
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x13c410790>,
  <matplotlib.lines.Line2D at 0x13cc1ec50>,
  <matplotlib.lines.Line2D at 0x13cc88e50>,
  <matplotlib.lines.Line2D at 0x13cc8a750>,
  <matplotlib.lines.Line2D at 0x13cd16bd0>,
  <matplotlib.lines.Line2D at 0x13cd17150>,
  <matplotlib.lines.Line2D at 0x13ccb4610>,
  <matplotlib.lines.Line2D at 0x13ccb5c90>,
  <matplotlib.lines.Line2D at 0x13cd05ad0>,
  <matplotlib.lines.Line2D at 0x13cd04050>,
  <matplotlib.lines.Line2D at 0x13cd5fb10>,
  <matplotlib.lines.Line2D at 0x13cd5ee90>],
 'caps': [<matplotlib.lines.Line2D at 0x13cc1edd0>,
```

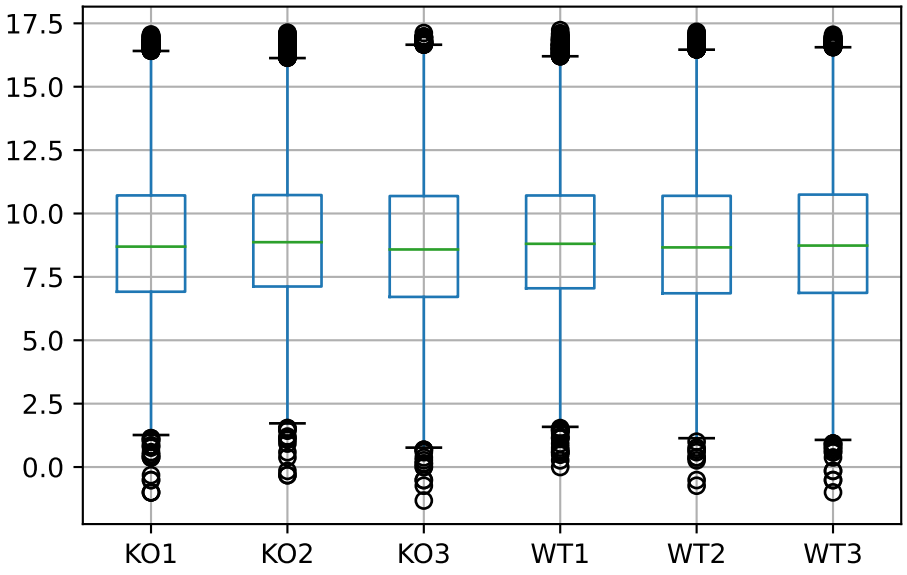
```
 <matplotlib.lines.Line2D at 0x13cc1d7d0>,
 <matplotlib.lines.Line2D at 0x13cc8b090>,
 <matplotlib.lines.Line2D at 0x13cc89d50>,
 <matplotlib.lines.Line2D at 0x13cd154d0>,
 <matplotlib.lines.Line2D at 0x13cd14ed0>,
 <matplotlib.lines.Line2D at 0x13ccb5f90>,
 <matplotlib.lines.Line2D at 0x13ccb7f90>,
 <matplotlib.lines.Line2D at 0x13cd05610>,
 <matplotlib.lines.Line2D at 0x13cd05c50>,
 <matplotlib.lines.Line2D at 0x13cd5d590>,
 <matplotlib.lines.Line2D at 0x13cd5fcd0>],
'boxes': [<matplotlib.lines.Line2D at 0x13c411710>,
 <matplotlib.lines.Line2D at 0x13cc88910>,
 <matplotlib.lines.Line2D at 0x13cd16750>,
 <matplotlib.lines.Line2D at 0x13cd16710>,
 <matplotlib.lines.Line2D at 0x13ccb6dd0>,
 <matplotlib.lines.Line2D at 0x13cd04690>],
'medians': [<matplotlib.lines.Line2D at 0x13c4622d0>,
 <matplotlib.lines.Line2D at 0x13cc89bd0>,
 <matplotlib.lines.Line2D at 0x13cd14fd0>,
 <matplotlib.lines.Line2D at 0x13ccb4450>,
 <matplotlib.lines.Line2D at 0x13cd06c90>,
 <matplotlib.lines.Line2D at 0x13cd5ced0>],
'fliers': [<matplotlib.lines.Line2D at 0x13c3234d0>,
 <matplotlib.lines.Line2D at 0x13cc1df90>,
 <matplotlib.lines.Line2D at 0x13cc1f190>,
 <matplotlib.lines.Line2D at 0x13db2d2d0>,
 <matplotlib.lines.Line2D at 0x13cd05590>,
 <matplotlib.lines.Line2D at 0x13ccb4c10>],
'means': []}
```



4.1.2 Using Pandas

```
data2.boxplot()
```

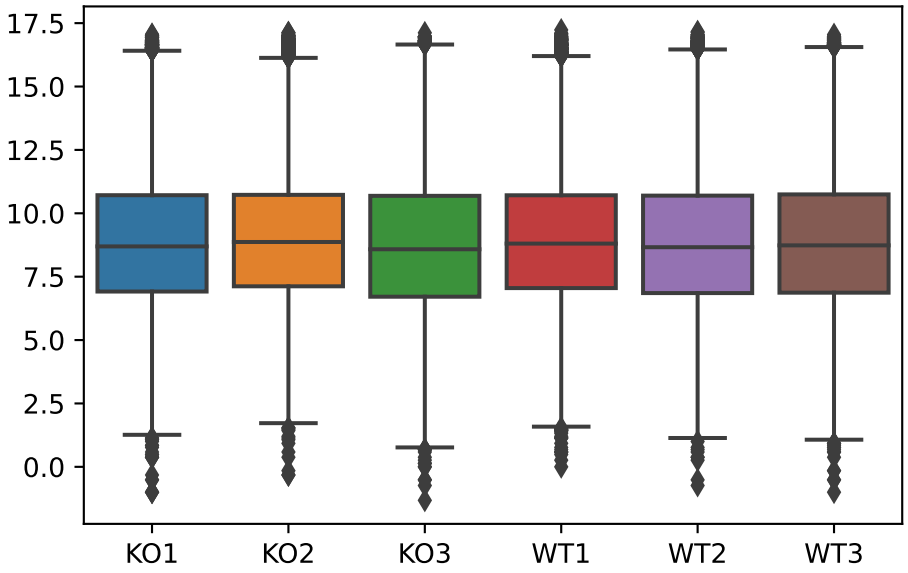
```
<Axes: >
```



4.1.3 Using Seaborn

```
sns.boxplot(data2)
```

```
<Axes: >
```

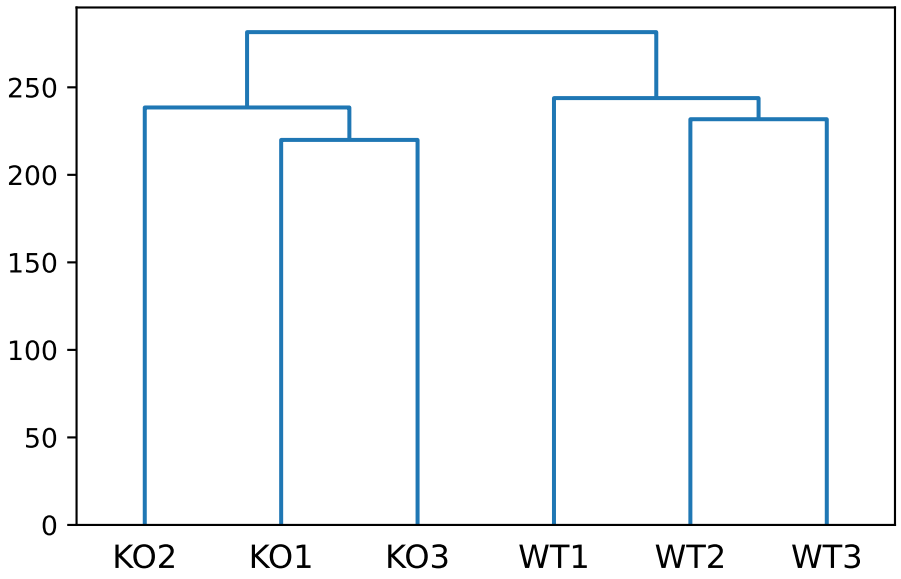


4.2 Hierarchical clustering

```
linkage_matrix = hierarchy.linkage(data2.T, method='ward')  # Transpose data with .T
hierarchy.dendrogram(linkage_matrix, labels = data2.columns)
```

```
{'icoord': [[15.0, 15.0, 25.0, 25.0],
  [5.0, 5.0, 20.0, 20.0],
  [45.0, 45.0, 55.0, 55.0],
  [35.0, 35.0, 50.0, 50.0],
  [12.5, 12.5, 42.5, 42.5]],
 'dcoord': [[0.0, 219.95424812831652, 219.95424812831652, 0.0],
  [0.0, 238.48396280958053, 238.48396280958053, 219.95424812831652],
  [0.0, 231.7745678766896, 231.7745678766896, 0.0],
  [0.0, 243.84222507735646, 243.84222507735646, 231.7745678766896],
  [238.48396280958053,
```

```
  281.5274405240738,
  281.5274405240738,
  243.84222507735646]],
'ivl': ['KO2', 'KO1', 'KO3', 'WT1', 'WT2', 'WT3'],
'leaves': [1, 0, 2, 3, 4, 5],
'color_list': ['C0', 'C0', 'C0', 'C0', 'C0'],
'leaves_color_list': ['C0', 'C0', 'C0', 'C0', 'C0', 'C0']}
```



5 Biological signifiance

5.1 Slicing the dataset by condition

```
ko = data2[['KO1', 'KO2', 'KO3']] # KO dataframe (KO1,KO2,KO3)
ko.head()
```

```
/Users/ahmed/Library/Python/3.11/lib/python/site-packages/IPython/core/formatters.py:342: Fu

In future versions `DataFrame.to_latex` is expected to utilise the base implementation of `St
```

ID	KO1	KO2	KO3
1415670_at	12.673088	12.441596	12.736064
1415671_at	13.487627	13.363957	13.377400
1415672_at	13.807677	13.723458	13.818253
1415673_at	11.624247	11.116019	11.672602
1415674_a_at	11.966505	11.691264	11.883025

```
wt = data2[['WT1', 'WT2', 'WT3']] # WT dataframe (WT1,WT2,WT3)
wt.head()
```

```
/Users/ahmed/Library/Python/3.11/lib/python/site-packages/IPython/core/formatters.py:342: Fu

In future versions `DataFrame.to_latex` is expected to utilise the base implementation of `St
```

ID	WT1	WT2	WT3
1415670_at	12.916645	12.812017	12.882624
1415671_at	13.345433	13.211356	12.901282
1415672_at	13.659171	13.756734	13.697587
1415673_at	11.213226	11.104468	11.302239
1415674_a_at	11.876747	11.615170	11.507547

5.2 Gene-wise mean expression

Note: the mean function can take the `axis` parameter to determine the direction of computing the mean, where:

- `axis=0` → vertical (by column), the default direction
- `axis=1` → horizontal (by row), the direction that we want in this case.

```
ko_means = ko.mean(axis=1) # Compute the means of the KO samples
ko_means.head()
```

ID	0
1415670_at	12.616916
1415671_at	13.409661
1415672_at	13.783129
1415673_at	11.470956
1415674_a_at	11.846931

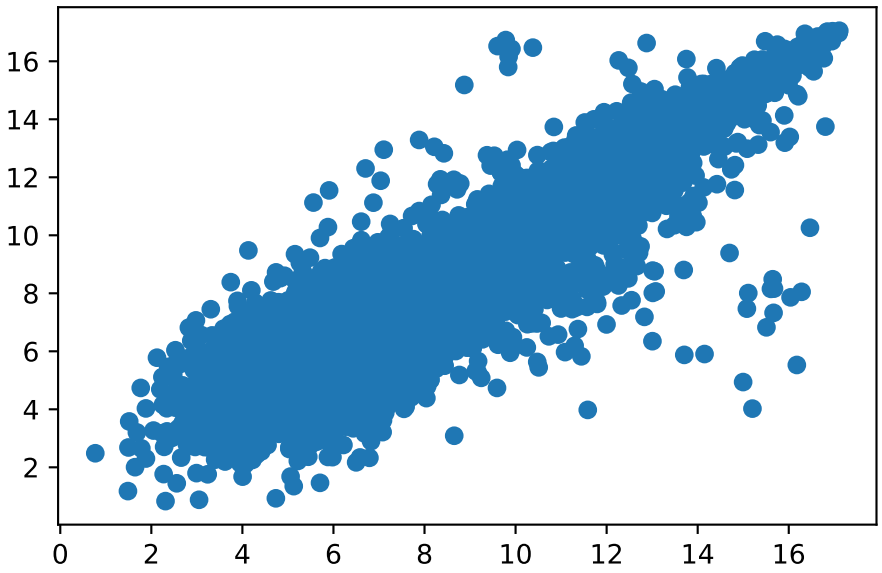
```
wt_means = wt.mean(axis=1) # Compute the means of the WT samples
wt_means.head()
```

ID	0
1415670_at	12.870428
1415671_at	13.152690
1415672_at	13.704497
1415673_at	11.206644
1415674_a_at	11.666488

5.3 Scatter plot using Matplotlib

```
plt.scatter(x = wt_means, y = ko_means)
```

```
<matplotlib.collections.PathCollection at 0x13d39c250>
```



5.4 Calculating the fold-change

```
fold_change = ko_means - wt_means # The difference between means
fold_change.head()
```

```
/Users/ahmed/Library/Python/3.11/lib/python/site-packages/IPython/core/formatters.py:342: Fu

In future versions `DataFrame.to_latex` is expected to utilise the base implementation of `St
```

ID	0
1415670_at	-0.253513
1415671_at	0.256971
1415672_at	0.078632
1415673_at	0.264312
1415674_a_at	0.180443

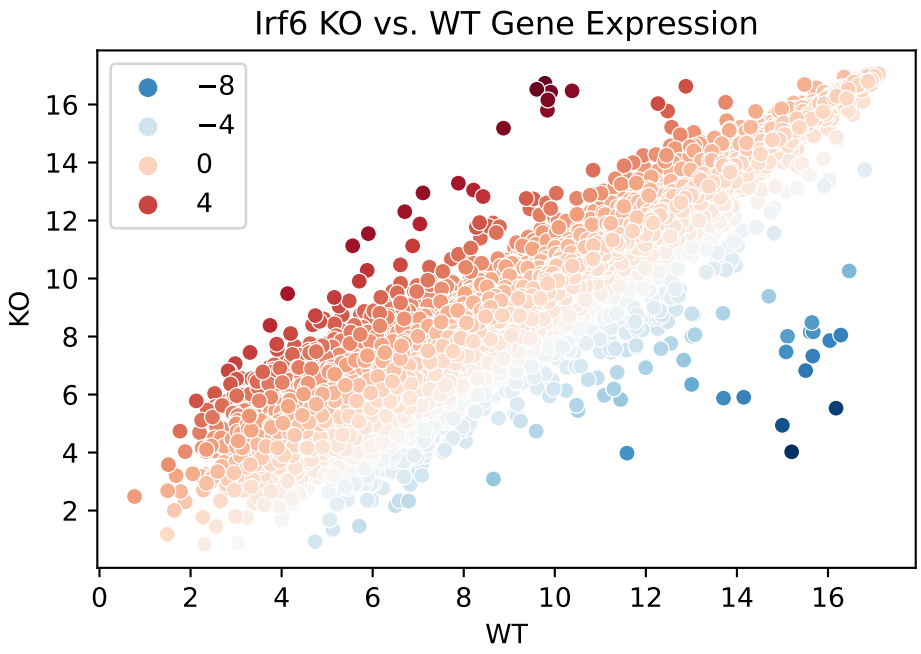
Note:

- **+ve** fold-change → **Up**-regulation ↑
- **-ve** fold-change → **Down**-regulation ↓

5.5 Scatter plot using Seaborn

```
sns.scatterplot(x = wt_means, y = ko_means, hue = fold_change, palette='RdBu_r')
plt.xlabel('WT')
plt.ylabel('KO')
plt.title('Irf6 KO vs. WT Gene Expression')
```

```
Text(0.5, 1.0, 'Irf6 KO vs. WT Gene Expression')
```



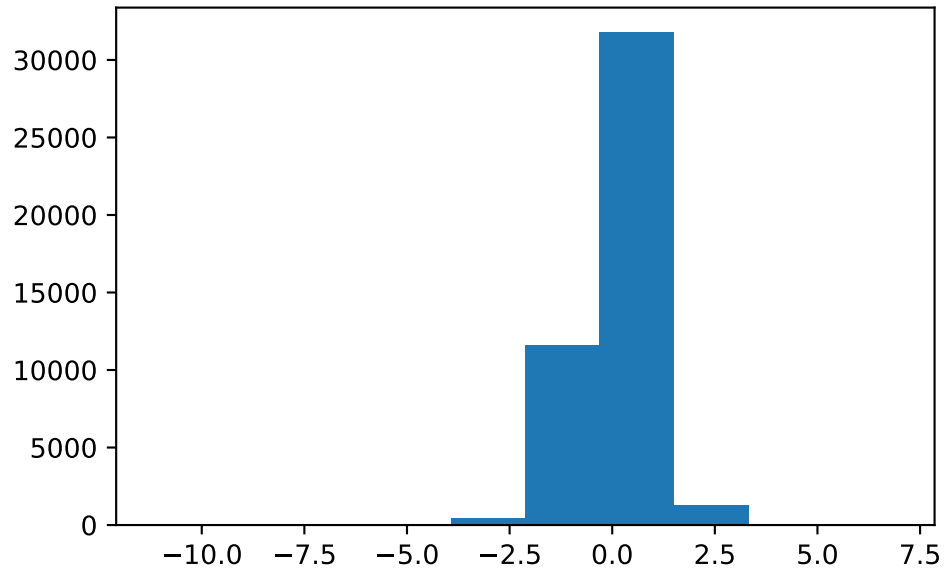
5.6 Historgram of the fold-change

5.6.1 Using Matplotlib

```
plt.hist (fold_change)
```

```
(array([3.0000e+00, 8.0000e+00, 6.0000e+00, 3.2000e+01, 4.0100e+02,
        1.1552e+04, 3.1788e+04, 1.2640e+03, 3.4000e+01, 1.3000e+01]),
 array([-11.17862753,  -9.36607657,  -7.55352561,  -5.74097464,
        -3.92842368,  -2.11587271,  -0.30332175,   1.50922922,
```

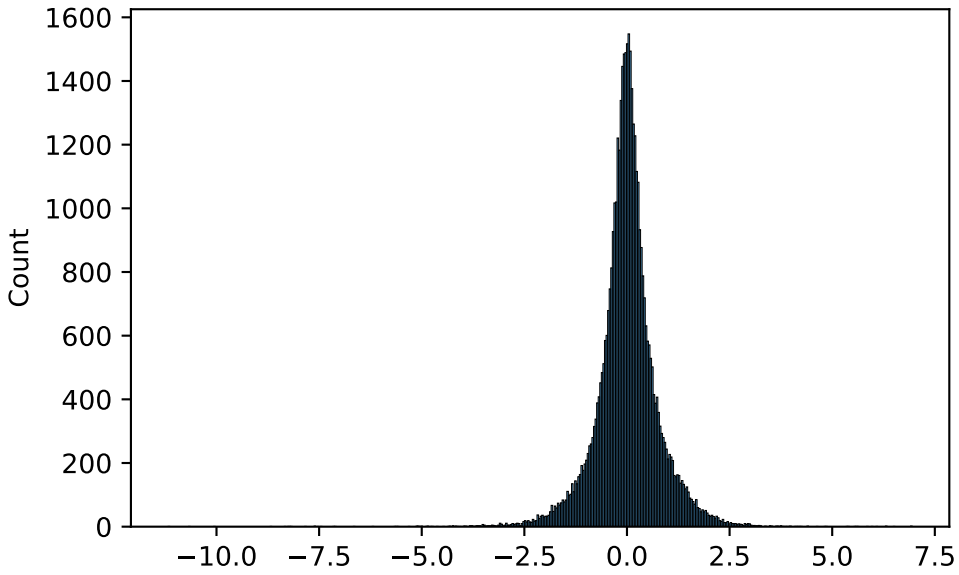
```
        3.32178018,   5.13433114,   6.94688211]),
<BarContainer object of 10 artists>)
```



5.6.2 Using Seaborn

```
sns.histplot (fold_change)
```

```
<Axes: ylabel='Count'>
```



6 Statistical significance

6.1 Calculating t-test *p*-value

(a p-value for each gene i.e., horizontally)

```
t_stat, p_value = stats.ttest_ind(ko, wt, axis=1)
t_stat_df = pd.DataFrame({'t_stat': t_stat, 'p_value': p_value})
t_stat_df.head()
```

```
/Users/ahmed/Library/Python/3.11/lib/python/site-packages/IPython/core/formatters.py:342: Fu

In future versions `DataFrame.to_latex` is expected to utilise the base implementation of `St
```

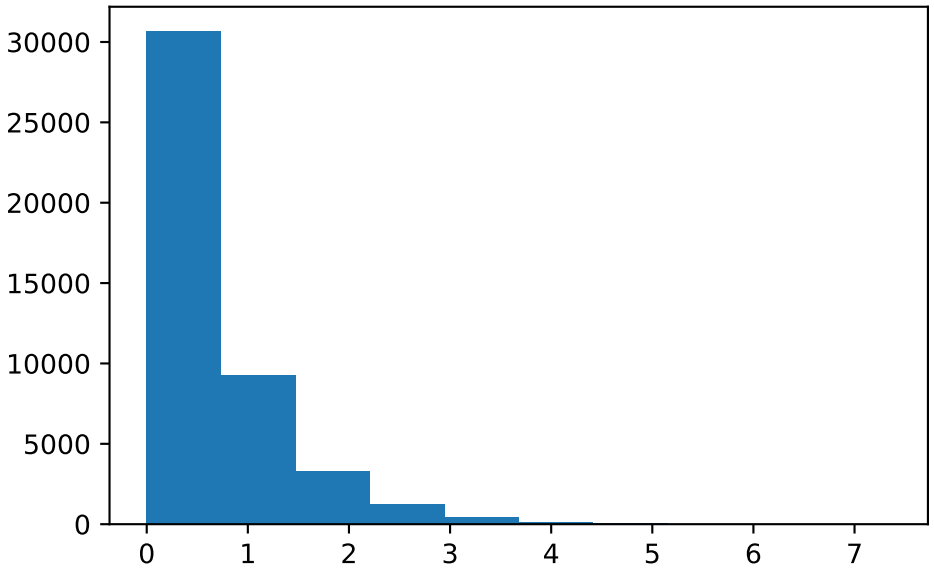
	t_stat	p_value
0	-2.677586	0.055367
1	1.872445	0.134450
2	1.904526	0.129561
3	1.413610	0.230364
4	1.321077	0.256980

6.2 Historgram of the *p*-value

6.2.1 Using Matplotlib

```
plt.hist (-np.log10(t_stat_df['p_value']))
```

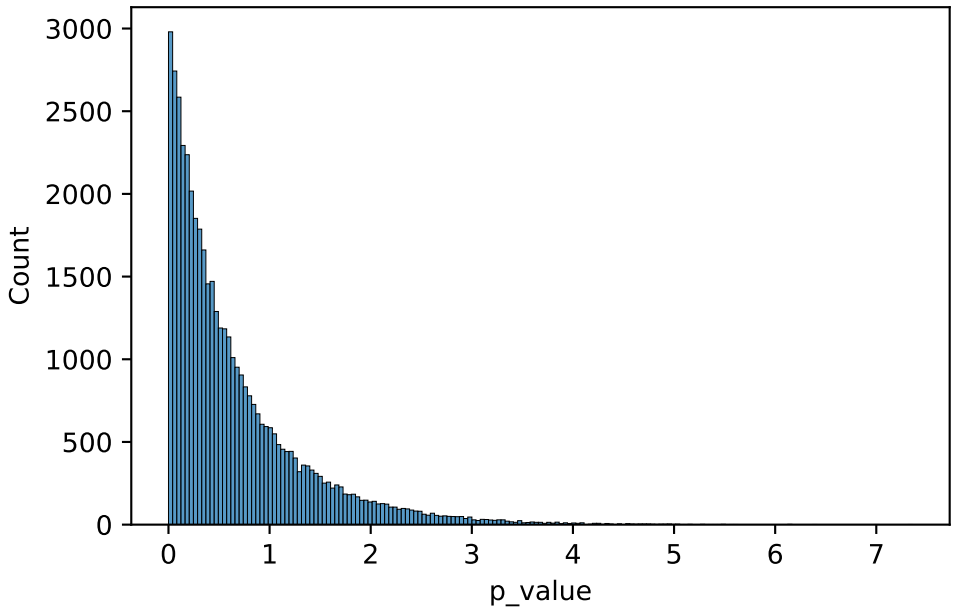
```
(array([3.0659e+04, 9.2820e+03, 3.2870e+03, 1.2330e+03, 4.3300e+02,
        1.3700e+02, 5.4000e+01, 1.1000e+01, 3.0000e+00, 2.0000e+00]),
 array([2.47714579e-06, 7.35796647e-01, 1.47159082e+00, 2.20738499e+00,
        2.94317915e+00, 3.67897332e+00, 4.41476749e+00, 5.15056166e+00,
        5.88635583e+00, 6.62215000e+00, 7.35794417e+00]),
 <BarContainer object of 10 artists>)
```



6.2.2 Using Seaborn

```
sns.histplot (-np.log10(t_stat_df['p_value']))
```

```
<Axes: xlabel='p_value', ylabel='Count'>
```



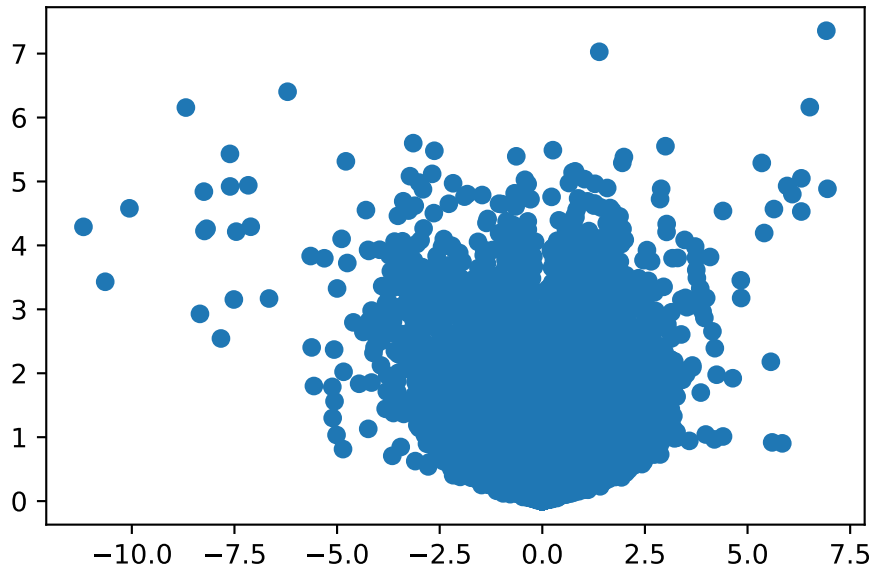
7 Biological & statistical signifiance

7.1 Volcano Plot (*p*-value vs. fold-change)

7.1.1 Using Matplotlib

```
plt.scatter (x = fold_change, y = -np.log10(t_stat_df['p_value']))
```

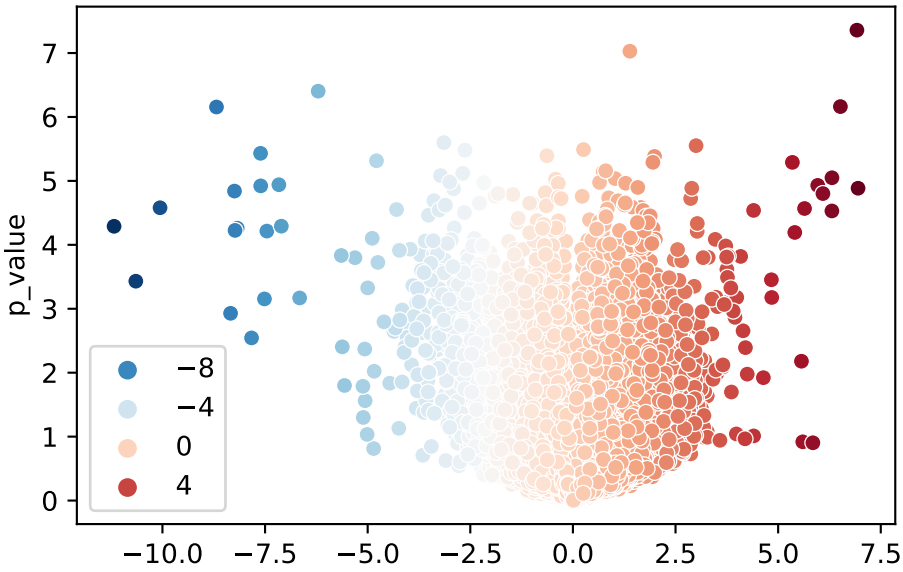
```
<matplotlib.collections.PathCollection at 0x13d2c4290>
```



7.1.2 Using Seaborn

```
sns.scatterplot (x = fold_change.values, y = -np.log10(t_stat_df['p_value']), hue = fold_c
```

```
<Axes: ylabel='p_value'>
```



8 Differentially expressed genes (DEGs)

8.1 Genes with significant fold-change

```
fold_change_cutoff = 2
np.sum(abs(fold_change.values) >= fold_change_cutoff)
```

```
1051
```

8.2 Genes with significant *p*-value

```
pvalue_cutoff = 0.001
np.sum(t_stat_df['p_value'] <= pvalue_cutoff)
```

```
575
```

8.3 Genes with significant fold-change & significant *p*-value

```
np.sum((abs(fold_change.values) >= fold_change_cutoff) & (t_stat_df['p_value'] <= pvalue_c
```

```
163
```

```
filtered = data2.reset_index().loc[(abs(fold_change.values) >= fold_change_cutoff) & (t_st
filtered.set_index('ID', inplace=True)
filtered.shape
filtered.head()
```

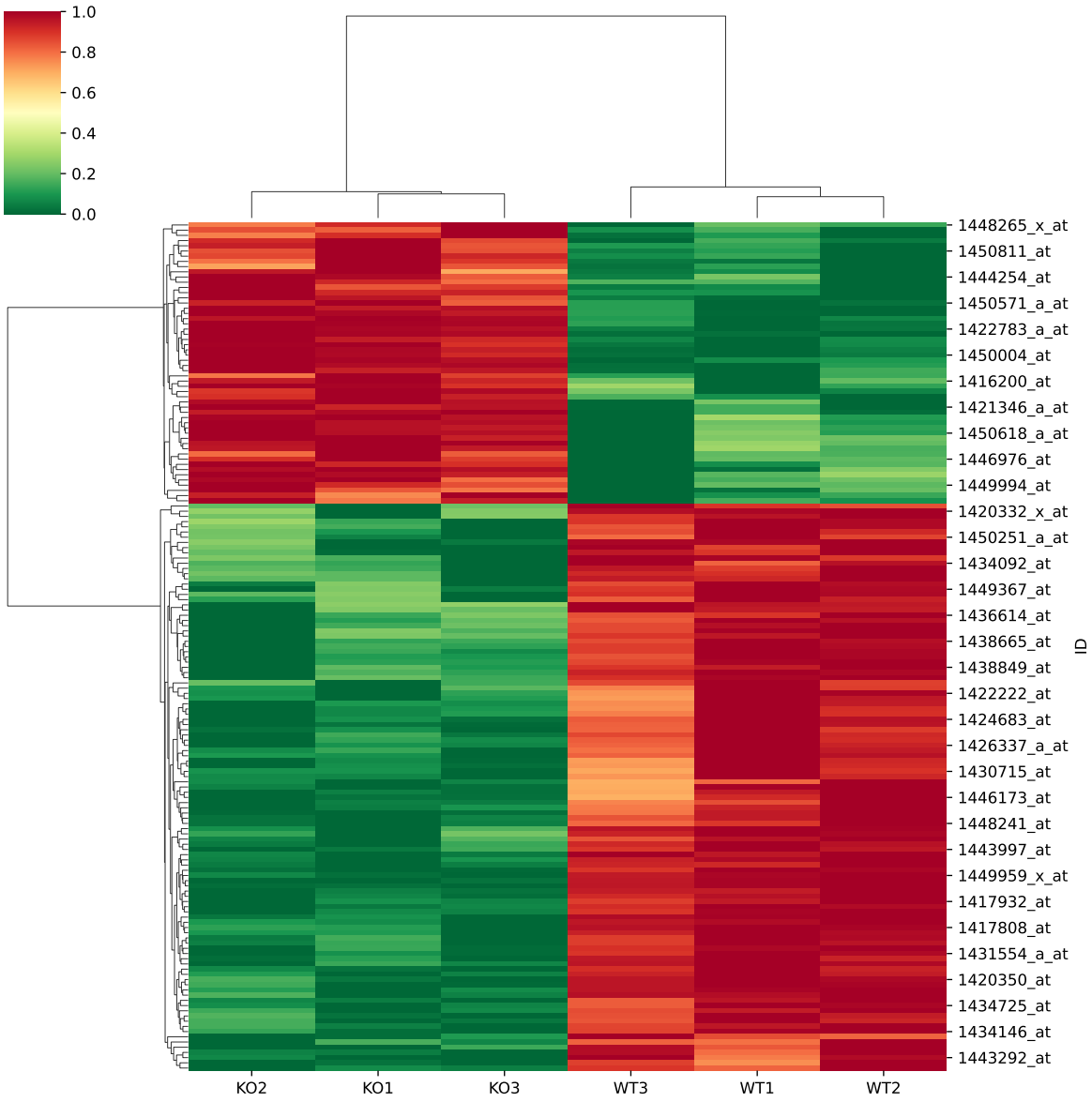
```
/Users/ahmed/Library/Python/3.11/lib/python/site-packages/IPython/core/formatters.py:342: Fu

In future versions `DataFrame.to_latex` is expected to utilise the base implementation of `St
```

ID	KO1	KO2	KO3	WT1	WT2	WT3
1416200_at	13.312004	12.973357	12.868456	7.404290	8.558803	8.683696
1416236_a_at	14.148397	14.039236	14.130007	12.236044	12.022402	11.495056
1417808_at	5.321928	5.442943	4.053111	15.169780	15.070087	14.753274
1417932_at	10.602884	10.257152	10.496055	13.984454	14.203294	13.720960
1418050_at	10.622052	10.975490	10.795066	12.865134	13.012048	12.658122

8.4 Heatmap

```
sns.clustermap(filtered, cmap='RdYlGn_r', standard_scale = 0)
```


1.0
0.8
0.6
0.4
0.2
0.0
1448265_x_at
1450811_at
1444254_at
1450571_a_at
1422783_a_at
1450004_at
1416200_at
1421346_a_at
1450618_a_at
1446976_at
1449994_at
1420332_x_at
1450251_a_at
1434092_at
1449367_at
1436614_at
1438665_at
1438849_at
1422222_at
1424683_at
1426337_a_at
1430715_at
1446173_at
1448241_at
1443997_at
1449959_x_at
1417932_at
1417808_at
1431554_a_at
1420350_at
1434725_at
1434146_at
1443292_at
ID
KO2
KO1
KO3
WT3
WT1
WT2