1. Сравнение реплик инструментом hicrep.

```
hicrep /home/kononkova/hic_data_hse/lib1.dm3.mapq_30.1000.mcool
/home/kononkova/hic_data_hse/lib2.dm3.mapq_30.1000.mcool outputSCC_lib.txt --binSize 10000
--h 1 --dBPMax 5000000
hicrep /home/kononkova/hic_data_hse/Kc167_rep1.dm3.mapq_30.1000.mcool
/home/kononkova/hic_data_hse/Kc167_rep2.dm3.mapq_30.1000.mcool outputSCC_Kc167.txt --
binSize 10000 --h 1 --dBPMax 5000000
hicrep /home/kononkova/hic_data_hse/lib1.dm3.mapq_30.1000.mcool
/home/kononkova/hic_data_hse/Kc167_rep2.dm3.mapq_30.1000.mcool outputSCC_lib_Kc_1.txt --
binSize 10000 --h 1 --dBPMax 5000000
hicrep /home/kononkova/hic_data_hse/Kc167_rep1.dm3.mapq_30.1000.mcool
/home/kononkova/hic_data_hse/lib2.dm3.mapq_30.1000.mcool outputSCC_lib_Kc_2.txt --binSize
10000 --h 1 --dBPMax 5000000
hicrep /home/kononkova/hic_data_hse/lib1.dm3.mapq_30.1000.mcool
/home/kononkova/hic_data_hse/Kc167_rep1.dm3.mapq_30.1000.mcool outputSCC_lib_Kc_3.txt --
binSize 10000 --h 1 --dBPMax 5000000
hicrep /home/kononkova/hic_data_hse/lib2.dm3.mapq_30.1000.mcool
/home/kononkova/hic_data_hse/Kc167_rep2.dm3.mapq_30.1000.mcool outputSCC_lib_Kc_4.txt --
binSize 10000 --h 1 --dBPMax 5000000
```

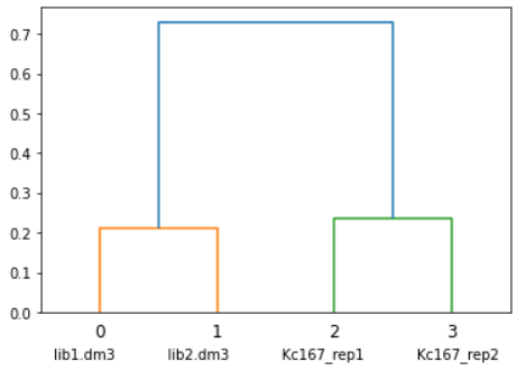
Для расчётов я воспользуюсь python.

В [1]:
```
import pandas as pd
import numpy as np
from scipy.cluster import hierarchy
import matplotlib.pyplot as plt
```

В [2]:
```
scc_list = ('outputSCC_lib.txt',
            'outputSCC_Kc167.txt',
            'outputSCC_lib_Kc_1.txt',
            'outputSCC_lib_Kc_2.txt',
            'outputSCC_lib_Kc_3.txt',
            'outputSCC_lib_Kc_4.txt')
scc_dist = []

for i in range(len(scc_list)):
    with open(scc_list[i], 'r') as f:
        lib = f.readlines()
    scc = [float(x.strip()) for x in lib[2:-1]]
    scc = [i for x, i in enumerate(scc) if x in [x - 1 for x in [1, 3, 5, 7, 9, 12]]]
    scc_dist.append(np.mean(scc))

print(scc_dist)
```

```
[0.85045980851829, 0.8368286767714815, 0.5409096974046415, 0.5684901964584212, 0.5727378813725648, 0.5404668147418871]
```

В [3]:
```
names = ('lib1.dm3', 'lib2.dm3', 'Kc167_rep1', 'Kc167_rep2')
dist_matrix = np.array([[1, scc_dist[0], scc_dist[4], scc_dist[2]],
                        [scc_dist[0], 1,scc_dist[3], scc_dist[5]],
                        [scc_dist[4], scc_dist[3], 1, scc_dist[1]],
                        [scc_dist[2], scc_dist[5], scc_dist[1], 1]])
dm = hierarchy.linkage(dist_matrix, 'single')
dn = hierarchy.dendrogram(dm)
plt.xlabel("            ".join(names)) # пришлось сделать так, подписи не хотели рисоваться красиво
plt.show()
```



Чего и следовало ожидать, реплики нервной и эмбриональной ткани разделились попарно.

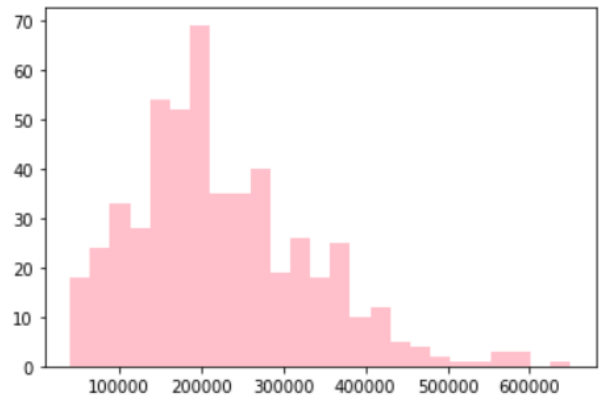
2. TADs calling.

Запускаю hicFindTADs со стандартными параметрами.

```
hicFindTADs -m /home/kononkova/hic_data_hse/lib_1_and_2.dm3.mapq_30.1000.mcool::/
resolutions/10000 --outPrefix TADs --correctForMultipleTesting fdr --chromosomes chr2L
chr2R chr3L chr3R chrX
```

```
tads_default = pd.read_csv('tads/TADs_domains.bed', sep='\t', header=None)
tads_default_len = [x - y for x, y in zip(tads_default[2].tolist(), tads_default[1].tolist())]
print('mean length of TADs', np.mean(tads_default_len, dtype=int))
plt.hist(tads_default_len, bins=25, color='pink')
plt.show()
```

```
mean length of TADs 224575
```

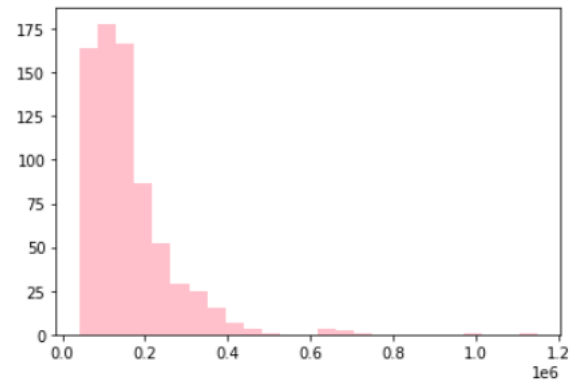


Я сделала несколько запусков с разными значениями –minDepth (30k, 100k, 200k) и –maxDepth (50k, 100k, 250k, 500k, 1M). Команды аналогичны этой:

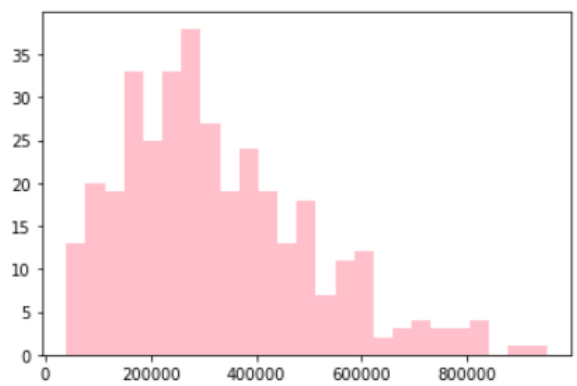
```
hicFindTADs -m /home/kononkova/hic_data_hse/lib_1_and_2.dm3.mapq_30.1000.mcool::/
resolutions/10000 --outPrefix TAD_30k_100k --correctForMultipleTesting fdr --chromosomes
chr2L chr2R chr3L chr3R chrX --minDepth 30000 --maxDepth 100000
```

При небольших значениях minDepth средняя длина ТАДов уменьшается и маленьких становится сильно больше. Но разброс значений при этом большой. С увеличением minDepth ТАДы становятся больше и разброс уменьшается.

```
-minDepth 30k -maxDepth 100k
mean length of TADs 157921
```



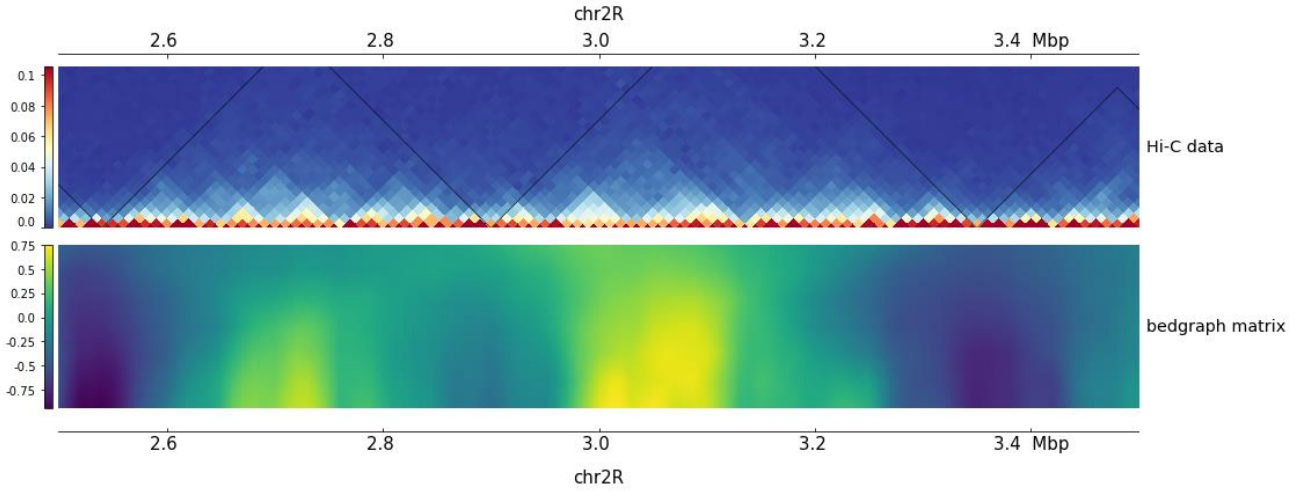

```
-minDepth 200k -maxDepth 1M
mean length of TADs 325071
```



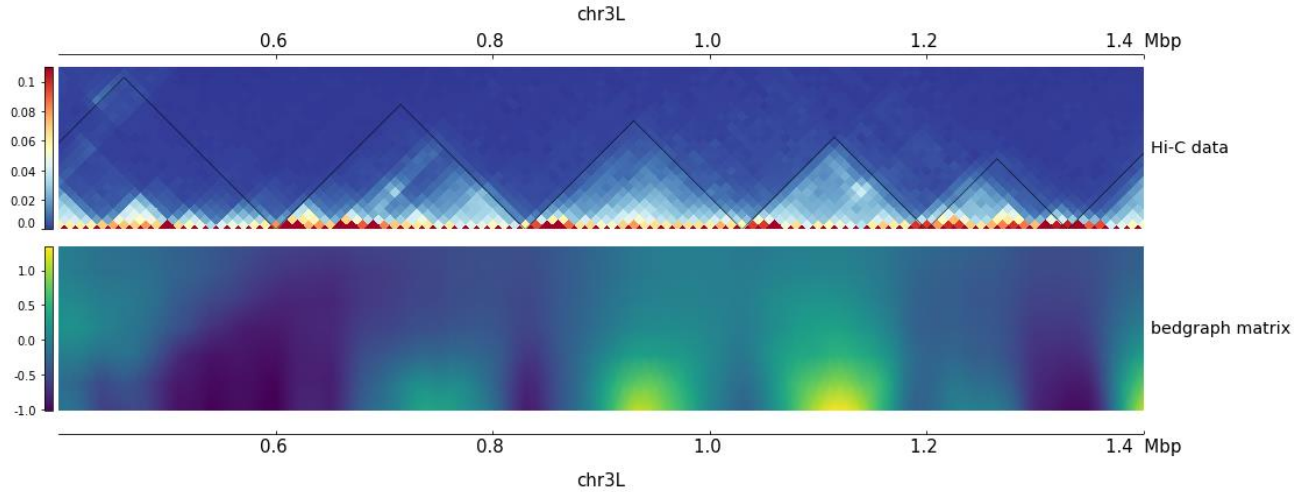
3. Визуализация.

Я выбрала два региона на разных хромосомах. И попробовала визуализировать ещё bedgraph matrix, получившиеся после hicFindTADs.

```
hicPlotTADs --tracks tracks.ini --region chr2R:2500100-3500100 -o drosophila_tads_2R_bm.png
```

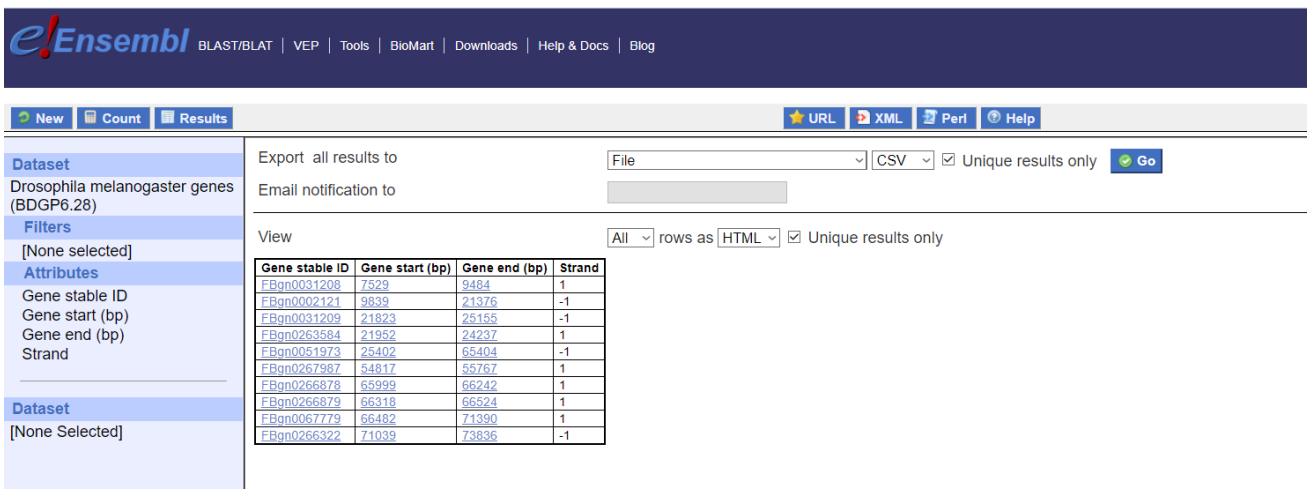



```
hicPlotTADs --tracks tracks.ini --region chr3L:400050-1400050 -o drosophila_tads_3L_bm.png
```



4. Взаимосвязь между границами ТАДов и экспрессией.

Информацию о генах *Drosophila melanogaster* я скачала с Ensembl. Далее также считала в python.



Gene stable ID	Gene start (bp)	Gene end (bp)	Strand
FBgn0031208	7529	9484	1
FBgn0002121	9839	21376	-1
FBgn0031209	21823	25155	-1
FBgn0263584	21952	24237	1
FBgn0051973	25402	65404	-1
FBgn0267987	54817	55767	1
FBgn0266878	65999	66242	1
FBgn0266879	66318	66524	1
FBgn0067779	66482	71390	1
FBgn0266322	71039	73836	-1

Координаты в файле *boundaries.bed* содержат координаты границ ТАДов. В то время как *domains.bed* содержит координаты самих ТАДов. Я решила, что если начало или конец гена попадает в промежуток между началом и концом границы ТАДа, будем считать, что он относится к границе. Если не попадает в этот промежуток – ген находится в ТАДе. Для поиска по всем генам получилось, что больше генов находится в граничных регионах.

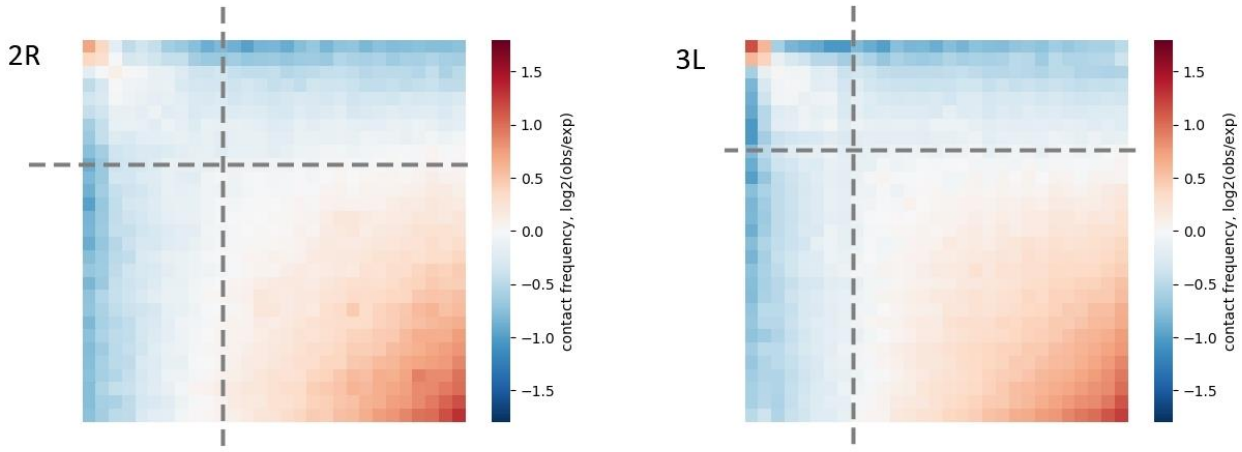
```
gene_start = drosophila['Gene start (bp)'].tolist()
gene_end = drosophila['Gene end (bp)'].tolist()

i = 0 # итератор генов
b = 0 # попадает на границу
x = 0 # подает внутрь ТАДа
for n in range(len(TADs_boundaries)):
    while gene_end[i] < TADs_boundaries[1][n]:
        i += 1
    if TADs_boundaries[1][n] < gene_start[i] < TADs_boundaries[2][n] or \
    TADs_boundaries[1][n] < gene_end[i] < TADs_boundaries[2][n]:
        while TADs_boundaries[1][n] < gene_start[i] < TADs_boundaries[2][n] or \
        TADs_boundaries[1][n] < gene_end[i] < TADs_boundaries[2][n]:
            b += 1
            i += 1
    elif TADs_boundaries[2][n] < gene_start[i] < TADs_boundaries[1][n+1] or \
    TADs_boundaries[2][n] < gene_end[i] < TADs_boundaries[1][n+1]:
        if n < len(TADs_boundaries):
            while TADs_boundaries[2][n] < gene_start[i] < TADs_boundaries[1][n+1] or \
            TADs_boundaries[2][n] < gene_end[i] < TADs_boundaries[1][n+1]:
                x += 1
                i += 1
    elif TADs_boundaries[2][n] < gene_start[i]:
        while n < len(TADs_boundaries):
            if TADs_boundaries[2][n] > gene_start[i]:
                break
            n += 1
            continue
        if n == len(TADs_boundaries):
            break

print('in boundaries:', b)
print('in TADs:', x)
```

```
in boundaries: 447
in TADs: 246
```

Аналогичный алгоритм я повторила на генах с максимальной экспрессией. В скачанном мной датасете с Ensembl нашлось 340 и 400 генов. И получилось, что на границы попадает 19 генов, а в ТАДы – 56. Следовательно, в хорошо экспрессированные гены чаще находятся в ТАДах (jupyter notebook прилагаю).

5. Компартменты.

Saddle plot для хромосом 2R и 3L. Эти графики показывают частоту контактов по длине хромосомы. У обеих хромосом картина получается очень схожая: одно плечо хромосомы сильно меньше другого.



6. Получение Hi-C карт из fastq-файлов.

Выход моей программы.

```
(base) anastasia_rastvorova@hpc09:~/hi-c/dstlr$ nextflow distiller.nf -params-file ./project.yml
N E X T F L O W  ~  version 20.10.0
Launching `distiller.nf` [amazing_mercator] - revision: af64431922
executor >  local (14)
[-        ] process > download_truncate_chunk_fastqs                                    -
[skipped  ] process > local_truncate_chunk_fastqs (library:replica_1 run:lane1)         [100%] 2 of 2, stored: 2
[-        ] process > fastqc                                                            -
[fa/45bb23] process > map_parse_sort_chunks (library:replica_2 run:lane1 chunk:0)       [100%] 2 of 2
[59/b77b68] process > merge_dedup_splitbam (library:replica_1)                          [100%] 2 of 2
[51/de220c] process > bin_zoom_library_pairs (library:replica_1 filter:no_filter)       [100%] 4 of 4
[b1/2ca64b] process > merge_zoom_library_group_coolers (library_group:all filter:mapq_30) [100%] 4 of 4
[09/20a595] process > merge_stats_libraries_into_groups (library_group:all)             [100%] 2 of 2
Completed at: 18-Dec-2020 22:23:45
Duration    : 13m 46s
CPU hours   : 1.8
Succeeded   : 14
```

Посмотрим, что получилось.

```
cooler info results/coolers_library_group/BG3.dm3.no_filter.1000.mcool::/resolutions/10000
```

```
cooler info results/coolers_library_group/BG3.dm3.mapq_30.1000.mcool::/resolutions/10000
```

```
(base) anastasia_rastvorova@hpc09:~/hi-c/dstlr$ cooler info results/coolers_library_group/BG3.dm3.no_filter.1000.mcool::/resolutions/10000
{
    "bin-size": 10000,
    "bin-type": "fixed",
    "creation-date": "2020-12-18T22:19:53.145351",
    "format": "HDF5::Cooler",
    "format-url": "https://github.com/mirnylab/cooler",
    "format-version": 3,
    "generated-by": "cooler-0.8.10",
    "genome-assembly": "unknown",
    "nbins": 16880,
    "nchroms": 15,
    "nnz": 382553,
    "storage-mode": "symmetric-upper",
    "sum": 1055863
}
(base) anastasia_rastvorova@hpc09:~/hi-c/dstlr$ cooler info results/coolers_library_group/BG3.dm3.mapq_30.1000.mcool::/resolutions/10000
{
    "bin-size": 10000,
    "bin-type": "fixed",
    "creation-date": "2020-12-18T22:18:51.837230",
    "format": "HDF5::Cooler",
    "format-url": "https://github.com/mirnylab/cooler",
    "format-version": 3,
    "generated-by": "cooler-0.8.10",
    "genome-assembly": "unknown",
    "nbins": 16880,
    "nchroms": 15,
    "nnz": 366344,
    "storage-mode": "symmetric-upper",
    "sum": 1022730
}
```

Без фильтра (1055863), как понимаю, контактов получается больше, чем с фильтром (1022730).

Здесь же прилагаю, что у меня написано в *project.yml* (файл прилагаю).

```yaml
input:
  raw_reads_paths:
    replica_1:
      lane1:
       - /home/kononkova/hic_data_hse/SRR8195120_1.fastq.gz
       - /home/kononkova/hic_data_hse/SRR8195120_2.fastq.gz
    replica_2:
      lane1:
       - /home/kononkova/hic_data_hse/SRR8195121_1.fastq.gz
       - /home/kononkova/hic_data_hse/SRR8195121_2.fastq.gz

  library_groups:
    BG3:
     - replica_1
     - replica_2

  truncate_fastq_reads: 1000000

  genome:
    assembly_name: 'dm3'
    bwa_index_wildcard_path: '/mnt/local/vse2020/home/anastasia_rastvorova/hi-c/dm3.fa.gz .*'
    chrom_sizes_path:  '/mnt/local/vse2020/home/anastasia_rastvorova/hi-c/dm3.chrom.sizes.txt'

do_fastqc: False

map:
  chunksize: 0
  mapping_options: ''
  trim_options: ''
  use_custom_split: true

parse:
  make_pairsam: False
  drop_seq: False
  drop_readid: True
  keep_unparsed_bams: False
  parsing_options: '--add-columns mapq --walks-policy mask'

dedup:
  max_mismatch_bp: 1

bin:
  resolutions:
   - 1000000
   - 500000
   - 250000
   - 100000
   - 50000
   - 25000
   - 10000
   - 5000
   - 2000
   - 1000
  balance: true
  filters:
    no_filter: ''
    mapq_30: '(mapq1>=30) and (mapq2>=30)'

output:
  dirs:
    processed_fastqs: 'results/processed_fastqs/'
    mapped_parsed_sorted_chunks: 'results/mapped_parsed_sorted_chunks'
    fastqc: 'results/fastqc/'
    pairs_library: 'results/pairs_library'
    coolers_library: 'results/coolers_library/'
    coolers_library_group: 'results/coolers_library_group/'
    stats_library_group: 'results/stats_library_group/'
```