muon.meth and muon.atac

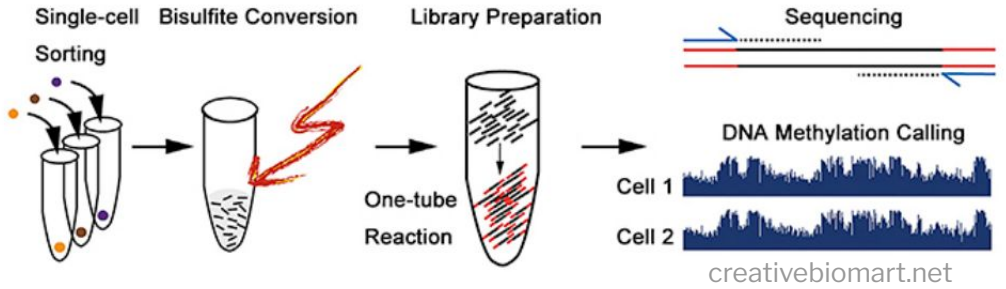
Ideas and concepts

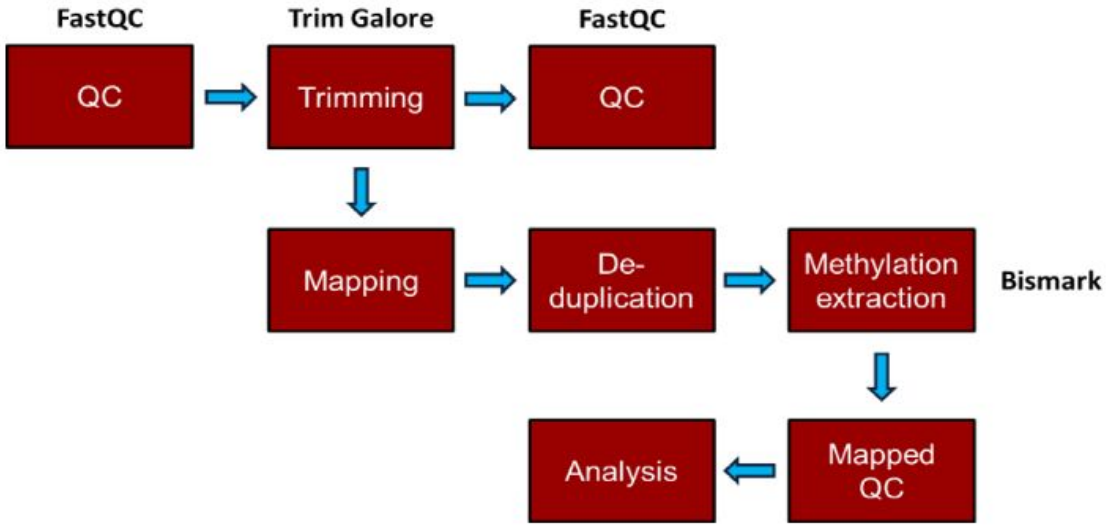
Max Frank
PhD Student
Stegle Group - European Molecular Biology Laboratory
25th Jan 2022, Omicstech Meeting

EMBL

Single cell whole-genome bisulfite sequencing

- Bisulfite converts unmethylated C to U
- Align to bisulfite converted Genome special aligner (e.g. Bismark)
- Properties of scWGBS data
  - Binomial readout for each cytosine
  - sparse (0.1 - 10% coverage)
  - Explicit zeros vs. missing values
  - Different cytosine contexts can be analyzed (CpG, CHH, CHG)





EMBL

Common starting point for methylation analysis - bedGraph

- Bismark methylation extractor outputs
- One file per cell
- Can be sorted and indexed
- Read methods to
  - Create count matrix for provided features (windows, promoters, TSS, etc.)
  - Random Access to C level info

<chromosome> <position> <strand> <count methylated> <count non-methylated> <C-context> <trinucleotide context>

tumor_sample_final.CpG_report.txt

4	10163	+	0	0	CG	CGC
4	10164	-	12	1	CG	CGT
4	10206	+	6	1	CG	CGG
4	10207	-	17	3	CG	CGA
4	10232	+	17	0	CG	CGT
4	10233	-	16	1	CG	CGG
4	10278	+	33	2	CG	CGT
4	10279	-	25	2	CG	CGT
4	10296	+	39	1	CG	CGC
4	10297	-	26	3	CG	CGC

<chromosome> <position> <strand> <rate> <count methylated> <count non-methylated>

tumor_sample_final.CpG_report.merged_CpG_evidence.cov

4	10163	10165	92.307692	12	1
4	10206	10208	85.185185	23	4
4	10232	10234	97.058824	33	1
4	10278	10280	93.548387	58	4
4	10296	10298	94.202899	65	4
4	10310	10312	93.150685	68	5
4	10322	10324	91.176471	62	6
4	10324	10326	94.117647	64	4
4	10350	10352	92.647059	63	5
4	10356	10358	92.957746	66	5

EMBL

muon.meth: IO

- Access to C level data
  - tabix index Bismark output bedGraph files (similar to ATAC)
  - Convenience function to index files
  - Retrieval function of all C in region to DataFrame/pyRanges
- Efficient construction of count matrix with specified features
  - Read in one cell at a time
  - Find overlaps with Features
  - Sum methylated/unmethylated counts per features
  - Add to sparse matrix (lil matrix for construction?)
  - convert to csr/csc
  - calculate methylation rate per feature (m-value, beta-value)

From Bismark output to count matrix

```python
def aggregate_metcounts(cdf):
    """Sum the methylated and unmethylated counts"""
    cdfg = cdf.groupby(["Start", "End"], as_index=False).agg(dict(Met="sum", Unmet="sum"))#.reset_index()
    cdfg.insert(0, column="Chromosome", value="chr1")
    return(cdfg)

def bismark_to_feature_counts(features, file):
    """ Convert bismark cpg counts to features counts"""
    print(file)
    cpg = read_bismark(file)
    tm = features.join(cpg, how="left")
    tmg = tm.apply(aggregate_metcounts)
    return tmg

def create_count_matrices(features: pr.PyRanges, files):
    Xmet = sp.lil_matrix((len(files), len(features)), dtype=np.int8)
    Xunmet = sp.lil_matrix((len(files), len(features)), dtype=np.int8)

    for i, file in enumerate(files):
        tmg = bismark_to_feature_counts(features, file)
        met = tmg.Met.to_numpy(dtype=np.int8) + 1
        Xmet[i,:] = met
        unmet = tmg.Unmet.to_numpy(dtype=np.int8) + 1
        Xunmet[i,:] = unmet

    Xmet.tocsr()
    Xunmet.tocsr()
    return Xmet, Xunmet
```

Fetch genomic features - Ensembl and Ucsc Rest API

```
import muon.meth as meth
u = meth.qu.Ucsc()
u.list_genomes()
```

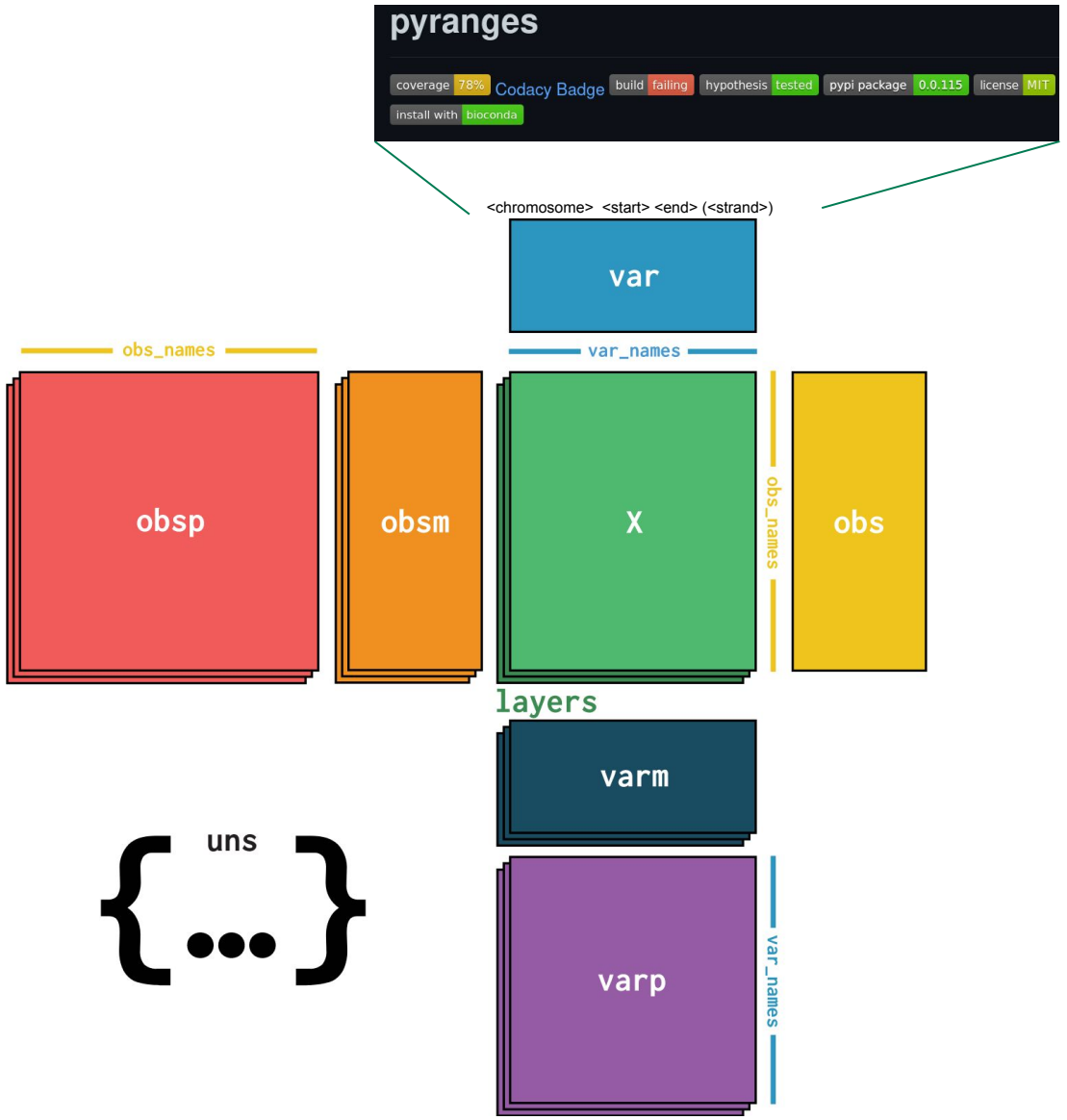
	organism	scientificName
ailMel1	Panda	Ailuropoda melanoleuca
allMis1	American alligator	Alligator mississippiensis
anoCar1	Lizard	Anolis carolinensis
anoCar2	Lizard	Anolis carolinensis
anoGam1	A. gambiae	Anopheles gambiae
...	...	...
xenTro10	X. tropicalis	Xenopus tropicalis
xenTro2	X. tropicalis	Xenopus tropicalis
xenTro3	X. tropicalis	Xenopus tropicalis
xenTro7	X. tropicalis	Xenopus tropicalis
xenTro9	X. tropicalis	Xenopus tropicalis

```
chromosomes = u.get_chromosome_sizes(genome="hg38")
chromosomes.head()
```

	length	coord_system
chr1	248956422	chromosome
chr21	46709983	chromosome
chr22	50818468	chromosome
chr19	58617616	chromosome
chr20	64444167	chromosome

muon.meth design: RangeData

- Data container: RangeData
  - thin wrapper around AnnData
  - .var is dataframe with mandatory columns <chromosome> <start> <end>
  - can be converted to pyranges
- pyranges allows for range manipulations
  - join
  - intersect
  - overlaps



EMBL

PyRanges are created on the fly and not stored

```python
class RangeData(anndata.AnnData):
    def __init__(self, *args, **kwargs):
        super(RangeData, self).__init__(*args, **kwargs)
        self._varranges = None

    @property
    def varranges(self):
        if self._varranges:
            return self._varranges
        else:
            df = self.var.copy()
            df["var_index"] = df.index
            ranges = pr.PyRanges(df) # This can be optimized, if we force var to be sorted
            self._varranges = ranges
            return ranges
```

RangeData allows to do genomic operations on the matrix

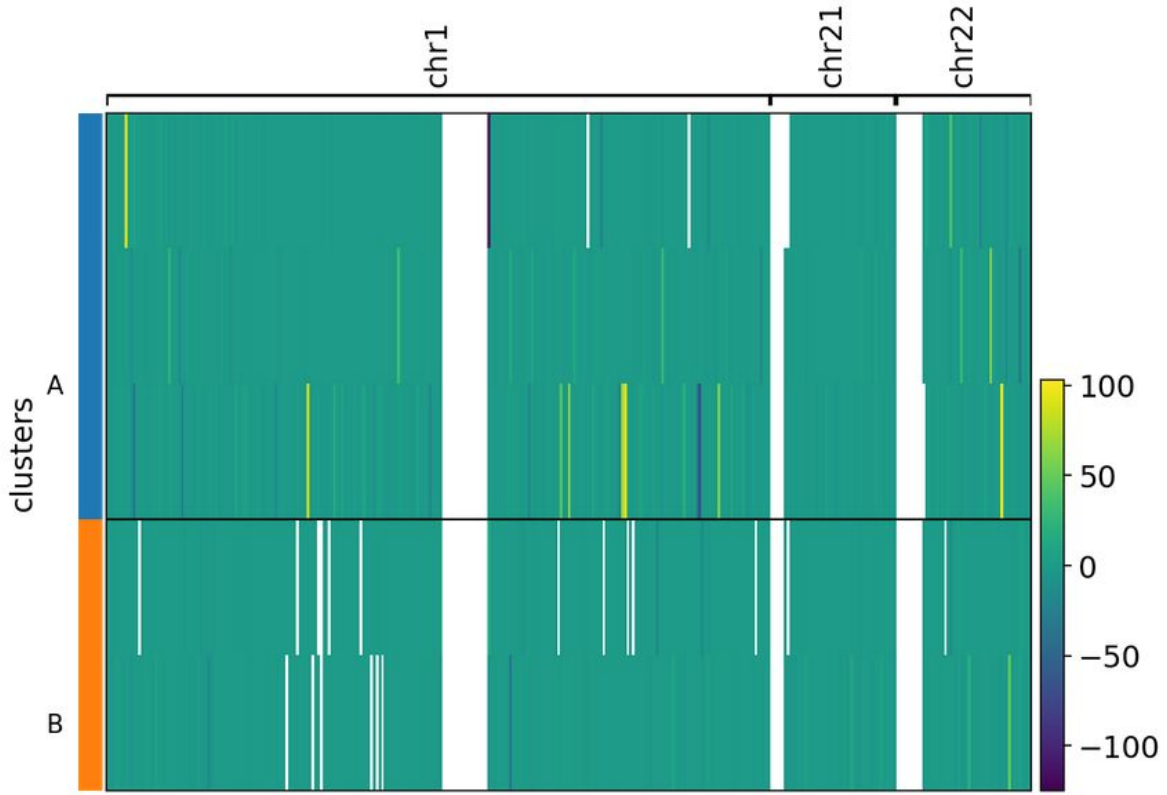
```python
def subset_by_intersection(self, ranges: pr.PyRanges):
    """Subset RangeData object by intersecting varranges with other ranges"""

    subs_ranges = self.varranges.intersect(ranges)
    sub = rad[:, subs.var_index]
    sub._varranges = subs_ranges
    return sub

def sum_by_range(self, ranges: pr.PyRanges, how: str="left"):
    """Add counts that fall within ranges to produce new RangeData"""

    new_ranges = ranges.join(self.varranges, how=how).cluster(slack=-1)
    g = new_ranges.as_df().groupby("Cluster", sort=False)
    m = sp.lil_matrix((len(g), self.n_obs))
    X = self.X.tocsc()
    for i, (name, group) in enumerate(g):
        idx = group.var_index.astype(int)
        s = X[:,idx].sum(axis=1)
        m[i,:] = s.flatten()
    return m.T.tocsr()
```


EMBL

Visualization: Genome overview with windows

```
tiles = pr.genomicfeatures.tile_genome(chromosomes.iloc[:3,:].to_dict()["length"], tile_size=1000000)
rad = rad.sum_by_range(tiles)
```

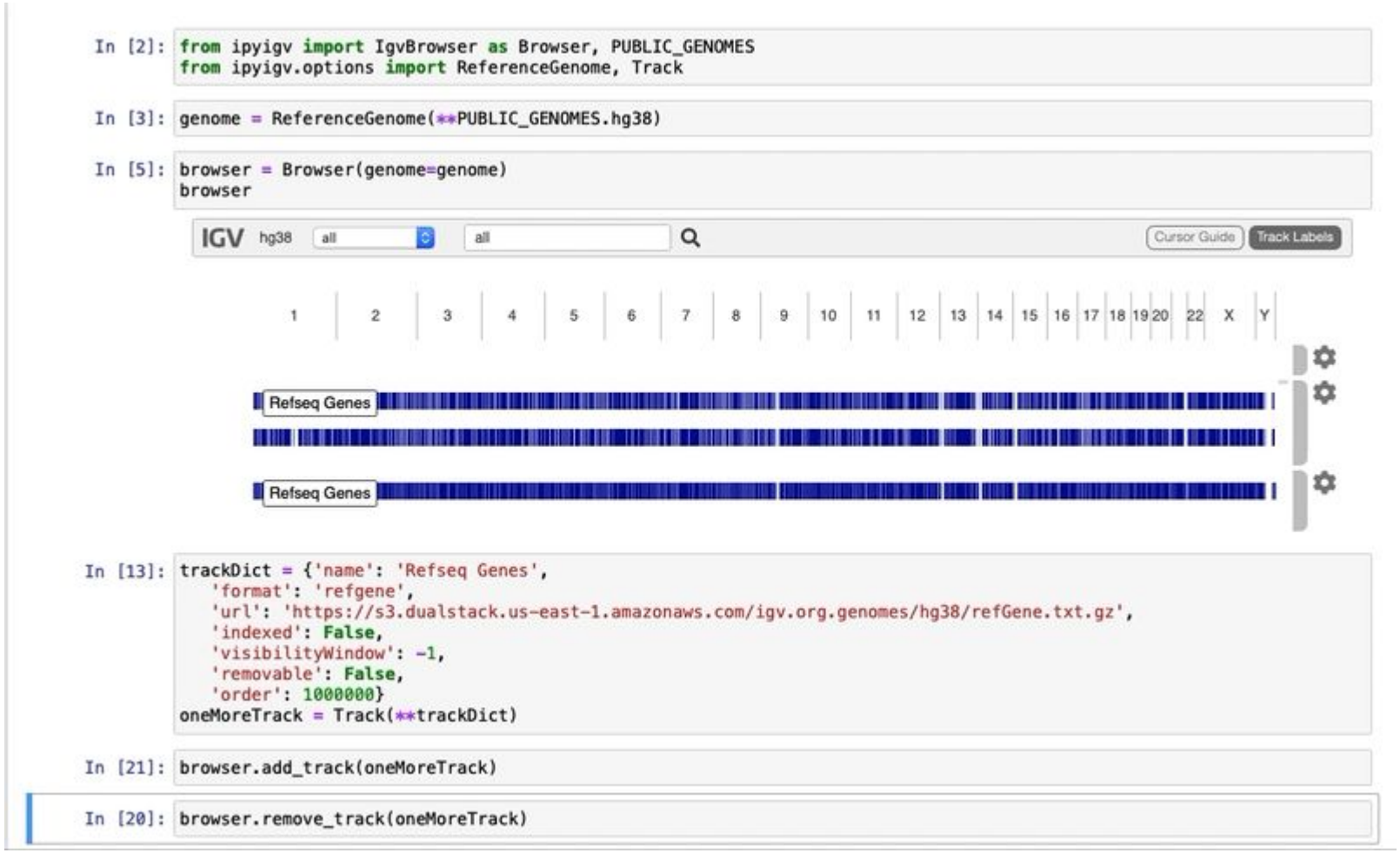
```
import scanpy as sc
sc.set_figure_params(figsize=(10,10))
varnames = rad.var.groupby("Chromosome").groups
sc.pl.heatmap(rad, var_names=varnames, groupby="clusters", show_gene_labels=False)
```



EMBL

Genomic visualisations - interactive

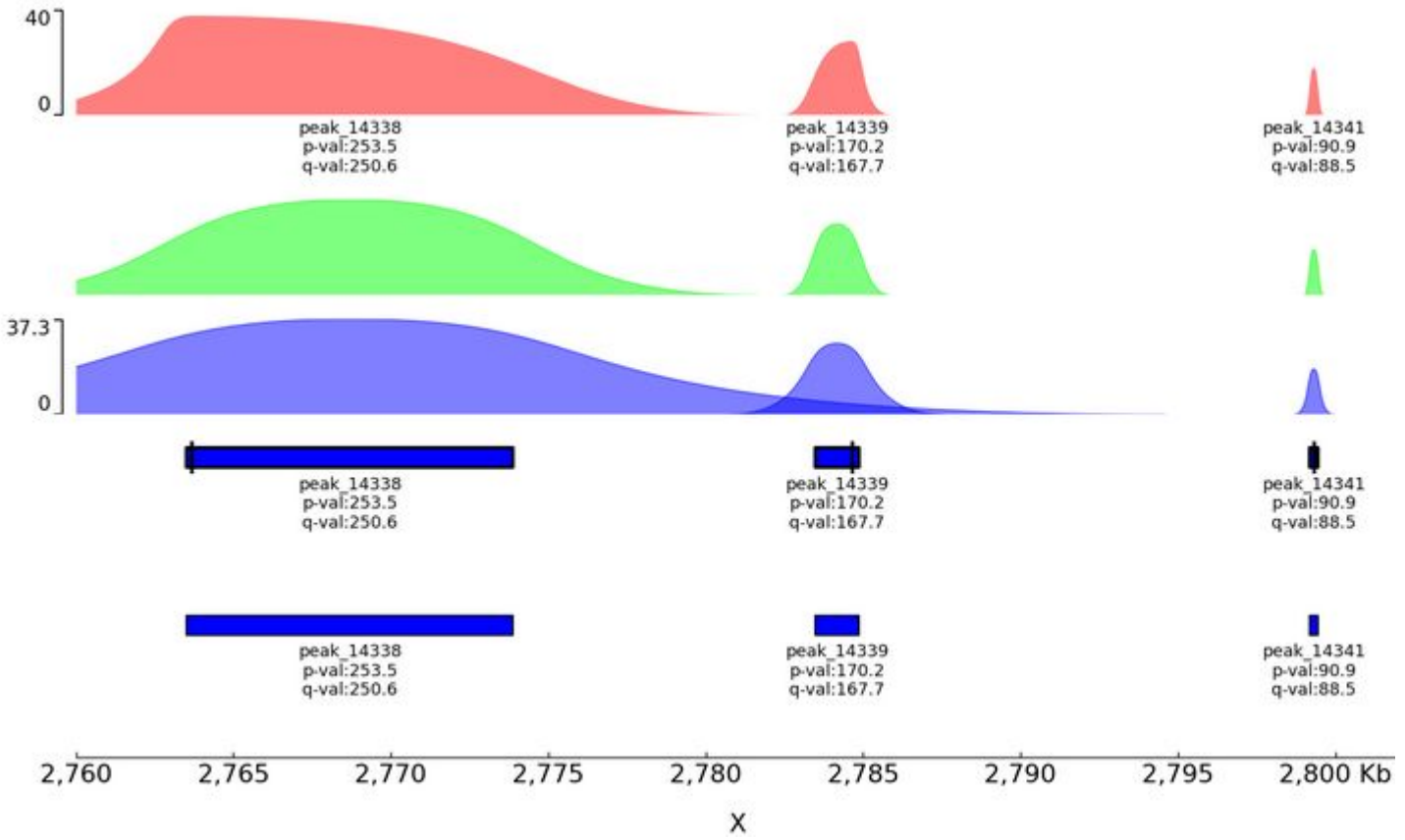
- **ipyigv** - jupyter plugin for igv.js
- load remote files
- serve local files via jupyter
- send in memory data via data URI
- Drawback: hard to add custom plots



EMBL

Genomic visualisations - static

- **pygenometracks -** matplotlib based genome track plotting
- Many track types available
  - bigwig
  - bed/gtf (many options)
  - bedgraph
  - epilogos
  - narrow peaks
  - etc.
- Drawback: no python api (only cli)
- Advantage: combine with any custom plot



EMBL

Todo

- Add differential tests
  - Fisher test
  - methylpy?
  - scMet?
- More Visualisations
  - Tracks plot with genomic Features alongside methylation (or ATAC coverage)
  - Averaged plots over features (e.g. TSS)
- Motif enrichment (overlaps with muon.atac)
- Finding gene-enhancer pairs (overlaps with muon.atac)