"Analyzing ChIP-seq data"

Practical BIOL3014: 10 marks
Workshop BINF7000: 20 marks

Below are a number of practical exercises that aim to guide you through issues related to processing genome-wide data, and analyzing ChIP-seq peaks for motifs.

Complete all exercises and record your results in your report as you go along. MS Word is a good choice for writing your report since you can easily paste the plots you are going to be making into it. Make sure you "cut-and-paste" into your report the exact commands/lines of code that you execute to create each plot. That will make your "experiments" repeatable, and hence, scientific. The report is submitted as a PDF for marking via the course web site—all parts in one document.

EXERCISES

BIOL3014: Complete all exercises except those marked with *. Bonus marks for *, but total capped at 10 marks.
BINF7000: Complete all exercises including those marked with *.

Part 1 contains exercises a, b, c and d*
Part 2 contains exercise a, b*, c, d, e, f, g, h, i*
Part 3 contains exercises a*, b*, c*

10 non-asterisked exercises are worth 1 mark each
6 asterisked exercises are worth 10 marks total

Part 1: whole-genome data

The portal for the UCSC Genome Browser (http://genome.ucsc.edu) contains the reference sequence and working draft assemblies for a large collection of genomes. We will use the 2009 assembly for human, also referred to as hg19 (see http://hgdownload.soe.ucsc.edu/downloads.html).

The human genome is large (3Gb) so we will use a compressed format known as '2bit' where each base is represented by 2 'bits'. If you are on campus, download the genome from the local server. Caution: the file is about 800MB.

http://bioinf.scmb.uq.edu.au/pubsvn/share/hg19.2bit

Fire up Python and use the module `seqdata` to have a closer look.

```
>>> import seqdata as sqd
>>> hg19 = sqd.TwoBitFile('hg19.2bit')
>>> for key in hg19:
```

```
        print key
```

To Python `hg19` is a dictionary. The for-loop will show you the main pieces that make up the genome. No prizes for guessing what they correspond to… You can access each chromosome separately, but be careful not to print the actual sequence unless you provide a 'genome location index' or 'range', as exemplified below. You now have every single base in the human (reference) genome at your fingertips!

```
>>> hg19['chrX']
    Out[1]: <seqdata.TwoBitSequence at 0x10cde7d90>
>>> len(hg19['chrX'])
    Out[1]: 155270560
>>> hg19['chrX'][1000000:1000060]
    Out[1]:
'AAAcagctacttggaaggctgaagcaggaggattgtttgagtctaggagtttgaggctgc'
```

The Encyclopedia of DNA Elements (ENCODE; http://genome.ucsc.edu/ENCODE/) Consortium is an international collaboration of research groups funded by the National Human Genome Research Institute (NHGRI). The goal of ENCODE is to build a comprehensive parts list of functional elements in the human genome, including elements that act at the protein and RNA levels, and regulatory elements that control cells and circumstances in which a gene is active.

ChIP-seq has been used extensively by ENCODE to determine where transcription factors bind, and where chromatin modifications and other major protein-DNA binding events occur. Let us look up some of these when responding to a few questions below.

a. Histone-3 Lysine-4 tri-methylation (H3K4me3) is known as an active promoter mark. Histone-3 Lysine-27 acetylation (H3K27ac) is regarded as an enhancer mark. i) How many genes in human embryonic stem cells (hESC) are marked with H3K27ac in the first 1000 base pairs upstream of the transcription start site? ii) What is the expected distance from a H3K4me3 mark and the closest gene? iii) What is the expected distance from a H3K27ac mark?

   To answer these questions you need hg19, the locations of all genes and H3K4me3 ChIP-seq data for hESC.

   You already have hg19. The UCSC Genome has a 'Table browser' (select menu "Tables", check the appropriate options—default should do it (knownGene), select BED-browser extensible data output format, click "get output", select "1" bp upstream, and download file. (`hum_TSS.bed` is already in your data directory. Familiarise yourself with the format.)

   Next, from the ENCODE page, go to and view the glorious Experiment matrix. Continue to the ChIP-seq experiment matrix, and download the 'broadPeak' files for the histone modifications. (`H3K4me3.broadPeak` and `H3K27ac.broadPeak` are in your directory.)

Once the data files are in place, you can use `seqdata` to load and use them, like so

```
>>> tss = sqd.BedFile('hum_TSS.bed')
>>> len(tss)
82960
>>> print tss[0]
('chr1', 10872, 10873)
>>> h3k4me3 = sqd.BedFile('H3K4me3.broadPeak')
>>> len(h3k4me3)
33270
>>> h3k4me3.closest(tss[0])
(701258, <seqdata.BedEntry instance at 0x10bb1e710>)
>>> print h3k4me3.closest(tss[0])[1]
('chr1', 713131, 713668)
```

With your two files now available as lists in Python, you should be able to answer the questions above, specifically to determine and plot the distance distributions. Compare those between H3K4me3 and H3K27ac and confirm that they are displaying what you would expect regarding their associations with promoters and enhancers. (Some example code below, but you may have to do some changes. Note that we log-transform the distances, and avoid log(0) by adding "1" to each distance.)

```
>>> import math
>>> d_h3k4me3 = []
>>> for mark in h3k4me3:
       d_h3k4me3.append(math.log10(tss.closest(mark)[0]+1))
>>> d_h3k27ac = []
>>> for mark in h3k27ac:
       d_h3k27ac.append(math.log10(tss.closest(mark)[0]+1))

>>> import numpy as np
>>> import matplotlib.pyplot as plt
>>> plt.figure(1, figsize=(8,8))
>>> ax = plt.axes([0.1, 0.1, 0.8, 0.8])
>>> n, bins, patches = plt.hist(d_h3k27ac, 100, normed=1, facecolor='r')
>>> n, bins, patches = plt.hist(d_h3k4me3, 100, normed=1, facecolor='g')
>>> plt.show()
```

Answer the questions i – iii.

b. * What does a 'broadPeak' file contain generally? Take a look inside and explain the information (including its source). You can find out more by browsing ENCODE (http://genome.ucsc.edu/FAQ/FAQformat.html).

Provide explanation.

c. CEBPB is a transcription factor. Construct a FASTA file, containing all the DNA sequences with CEBPB binding sites, so that someone, maybe you, can do transcription

factor binding motif discovery with it.

Download ChIP-seq data from ENCODE for it in human ES cells. (Pick the narrowPeak file, also found in your directory as `CEBPB.narrowPeak`.)

```
>>> cebpb = sqd.BedFile('CEBPB.narrowPeak')
>>> import sequence as seq
>>> import symbol as sym
>>> seqs = []
>>> for b in cebpb:
      try:
           seqstr = hg19[b.chrom][b.chromStart:b.chromEnd]
           newseq = seq.Sequence(seqstr, sym.DNA_Alphabet, str(b))
           if len(newseq) >= 100:
                seqs.append(newseq)
      except:
           print 'Failed to map', b.chrom, b.chromStart, b.chromEnd

>>> seq.writeFastaFile('cebpb.fa', seqs)
```

In fact, this saves only the sequences that have 100 nucleotides or more. Report the number of sequences that had 100 nucleotides (or more).

Report number.

d. * Is CEBPB more likely to be associated with promoters or enhancers? To answer this, you need to measure distances again, and substantiate your conclusions with plots of distances between CEBPB binding sites and promoter and enhancer associated histone marks.

Answer question.

Part 2: Transcription factor binding motifs

You are now going to analyze some sequence data taken from ChIP-seq data (as illustrated in part 1) and try to determine what motif describes the direct DNA binding by the transcription factor (TF) that was "pulled-down" in the immunoprecipitation experiment. The sequence data you are provided with is the genomic sequence around each of the "peak" regions declared by the "peak-calling" algorithm. Your mission is to perform a type of "motif enrichment analysis".

a. ChIP-seq nucleotide sequence data

Inspect the ChIP-seq peak region dataset (`chip-seq-peaks.fa`). It should contain sequences that are all exactly 100 bp long.

Start Python and load the sequence file after importing the module as described above:

```
>>> seqs = seq.readFastaFile('chip-seq-peaks.fa', sym.DNA_Alphabet)
```

Report how many sequences there are.

b. Plotting the average score of a PWM at each position across all sequences

We have prepared a module `wordcount` with a method `scanMotifReport` to compute the average score of a given motif at each position in a set of equal-length sequences. Go through the code, make sure you understand every step of it, and comment each line to describe the logic. You will need to peek in `sequence.py` where the code for PWM scoring is kept.

Choose a motif at random from `JASPAR_matrices.txt` (open the file and read the names of the motifs) and make the "average score density" plot for it. (Use this in *addition* to that referred to below.)

```
>>> from wordcount import *
>>> scanMotifReport(seqs, 'MA0456.1', 'JASPAR_matrices.txt')
```

Put the commented code and the plot in your report.

c. Smoothing the score across all sequences

It may be hard to see if the motif's scores are concentrated in a particular region of the ChIP-seq peaks. One way to address this problem is to "smooth" the plot.

Add code to the function `scanMotifReport` to so that instead of plotting the average motif score at each position, it plots the "running average" in a window of size 11 = `hw` + 1 + `hw`, where `hw` = 5. In other words, at position `X`, it plots the average over the positions `[X-5,…,X-1, X, X+1,…,X+5]`.

Note that the first 5 positions will have no smoothed score, so you can set their scores to zero. The same is true with the last 5 positions.

Now run and put the resulting plots in your report for `hw` = 1, `hw` = 3 and `hw` = 10. Which is smoother? Is smoother better to see motif concentration? Add the code for the whole function to your report with your own comments that explain the logic of the function as per the earlier question AND the smoothing. Set `hw` = 5 before you continue.

Put commented code and answers to questions about smoothing in your report.

d. Try three motifs and compare

Add the MA9999 motif to your collection of motifs to try. Could one of these three motifs be the DNA-binding motif for the ChIP-ed TF? Which one, and why do you think so?

Display the score plots for all of the three motifs on the same graph.

Put the plot in your report with your answers to the above questions.

e. Defining what is biologically meaningful using basic statistics

We are now (over several exercises) going to develop a new method that quantifies the "enrichment" of a motif. While there is no need to write down the answers yet, think about the following questions: What do we "expect" the plot of a randomly chosen motif to look like? How can we decide if a plot is "central" enough? How can we decide which of two motifs has a "more central" (that is "better") plot?

The (one) answer is to think about a single ChIP-seq peak region. That is, think about one of the sequences in our input set.

i. If you were to scan a single sequence with a "random" motif what do you think the probability of the best match being at position *X*, if the sequence is *L* base-pairs long? (Give a simple formula. Hints: The statistical distribution that describes the probability of the best match to the motif being at position *X* is the uniform distribution. Also, ask yourself, how many *different* matches can there be?)

ii. If you scan each of the *N* sequences with a "random" motif, and mark the location of the best match, how many matches would you expect (on average) at position *X*? (Give a formula. Hint: If you figured out the previous answer, you will know.)

The statistical distribution that describes the number of matches at position *X* is called the binomial distribution. You will use it below. See Wikipedia to understand the basics (http://en.wikipedia.org/wiki/Binomial_distribution) before you move on.

Answer questions *i* and *ii* in your report.

f. For a given site, plot the number of sequences where this site is their *best* motif match

Study `scanMotifReport_new()` in `wordcount.py`. Instead of averaging the scores at a given position, this function counts the number of times the best match to the motif occurs at that position. To do this, it needs to choose one best match in each sequence. To do that, for a given sequence, it needs to find the (set of) positions with maximal score, and choose one of them. If a sequence doesn't have any hits (over the score threshold), nothing gets added to your array of counts.

When all of the sequences have been processed, it divides the counts in the count array by the total count (by the total number of sequences that have at least one hit), the result will be a probability distribution. That is, the value at position *X* in the array will be the probability of a single sequence having its best hit at position *X*. Go through the code in detail and add your own comments to each line. Paste the code with comments in your report. You can leave out the section marked "STATISTICS"—we'll come back to this.

`scanMotifReport_new()` plots the smoothed version of the probability distribution referred to above, calling it the "probability" and the *X*-axis "position of best site". Using a score threshold is important when counting hits, so include the score threshold in your plot label. Now run the new function on your randomly chosen motif.

```
>>> scanMotifReport_new(seqs, 'MA0456.1')
```

Save the plot for your randomly selected motif, and place the PNG file in your report. Compare with another transcription factor "Oct" and the mystery motif.

```
>>> scanMotifReport_new(seqs, 'MA0197.1'); scanMotifReport_new(seqs, 'MA9999')
```

i. Does the "site-probability" method of plotting make the difference in central enrichment between these two motifs clearer? Why?
ii. Which motif do you think is more "centrally enriched"?
iii. Which motif do you think is more likely to be the DNA-binding motif of mystery the transcription factor and why?

Put commented code in your report. Add plot. Answer questions *i – iii*.

g. Quantifying which motif is more (centrally) enriched

A centrally enriched motif will have more "best hits" in a central region, right? So, what if we count the number of best hits in each central region of size R=2, 4, 6, … etc.? Then we test if the number of best hits is more than we would expect by chance.

i. If the motif is "random", how many best hits (out of *N*) would you expect on average to fall in a region of length *R* in a sequence of length L? (Give a formula. Hint: Similar to previous formula.)
ii. What is the name of the distribution that describes the number of best hits in a central region of length *R*? (Hint: You looked it up earlier.)

Answer questions *i* and *ii* in your report.

h. Determine the significance of motif matches in the middle

You should now understand how to make your new function apply a statistical test to each central region of length *R*. Add code to your new function `scanMotifReport_new()` to apply the Binomial test to the counts of best hits in each central region of width *R*=2, 4, 6, … Make your function report the lowest *p*-value (across all values of *R*) and the value of *R* that gives this *p*-value.

All you have to do is add up the counts in each region of your count array. Then, apply the Binomial test (http://en.wikipedia.org/wiki/Binomial_test) using the function we have provided:

```
binomial.log_binomial_ncdf(trials, successes, p_success)
```

There is a skeleton of code to get you on the right track marked "STATISTICS". Print the width of best region and the *p*-value in the label of the plot. (Some code has been commented out that do this.)

Now, run your new function on your random motif and put the plot in your report. Paste in the code that you used in the STATISTICS section.

Add commented code and plot to your report.

i. * Identify the mystery transcription factor

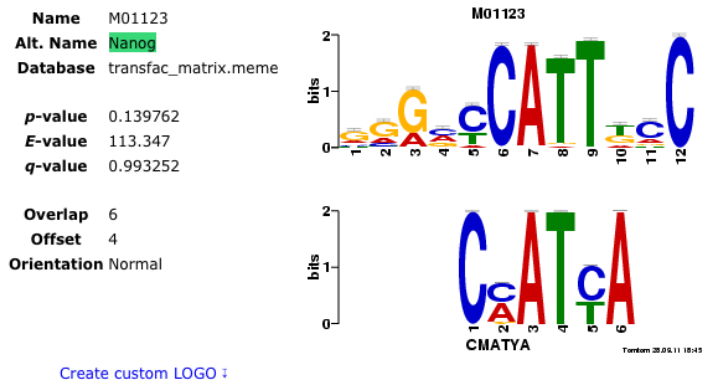
The ChIP-seq peak regions you have been using came from a ChIP-seq experiment that used antibodies to "Nanog". The motif was discovered by a motif discovery algorithm in these ChIP-seq peak regions. Run your new function on the published Nanog motif (M01247) and the "mystery" motif. Note that you may need to try different thresholds to the function.

```
>>> scanMotifReport_new(seqs, 'M01247'); scanMotifReport_new(seqs, 'MA9999
```

Answer these questions in your report:

i. Which motif (of the two) do you believe is the ChIP-ed transcription factor's binding motif?
ii. Give three things about the site-probability plot and statistical analysis that support your conclusion. Explain your reasoning.
iii. Which, in your opinion (and why) is more useful—the site-probability curve or the average score curve?

Here is a comparison (alignment) of the two motifs.



Answer questions *i – iii*.

==========

Part 3: Further research (all *)

a. * Add code to your function to correct for multiple testing.

In addition to the *p*-value, you should now print the *E*-value of the motif in its label on the plot.

i. Put the plot for the same two motifs in the report.
ii. Describe in your report how you compute the *E*-value of a motif: give the mathematical formula you use and explain what each of the variables in the formula means.

Respond to problems *i – ii*.

b. * Apply your new function to a new ChIP-seq data set

Re-use the FASTA file containing CEBPB binding sites that you created in part 1. It should have sequences that are exactly 100 sites long, centred on the middle of the original sequence. (You will have to write the code to do this.) This way, you can re-use the code that you completed in part 2.

i. In your report, state what motif in the JASPAR database (http://jaspar.genereg.net) should be enriched. Identify one that should not. Identify a third which is a close relative CEBPA.
ii. Run your new function on the three motifs and place a plot showing your results on the motifs in your report
iii. Comment on the location of the enrichment and discuss if there is additional information to be gained from the plot

Respond to problems *i – iii*. Include any code.

c. * Additional applications of your new function

i. Your new function can obviously be used to find the ChIP-ed TF in a ChIP-seq experiment. Could you also use it to find co-factors, i.e. proteins that may facilitate binding of the main factor, and/or co-modulate the effect of transcription? What steps would you take in this analysis—be specific.
ii. You have seen that local enrichment can detect the presence of signals at sites that have been identified through ChIP-seq using an anti-body of a transcription factor. In part 1, we explored histone modifications. Describe how you could use such signals to find transcription factor binding sites in the absence of ChIP-seq data.

Respond to problems *i – ii*.