MSCBIO 2070/02-710: Computational Genomics, Spring 2022

HW3: Genomics in the Real World

Version: 1
Due: 11:59 EST, March 30, 2022 by Gradescope

Topics in this assignment:

1. ChIP seq analysis
2. Linkage disequilibrium
3. Hardy-Weinberg equilibrium

All data for this assignment is available for download here.

What to hand in.

- One write-up (**in pdf format**) addressing each of following questions (this will be an assignment on Gradescope).
- All source code (this will be a separate assignment on Gradescope). Parts of your code will be autograded, and other parts may be graded by hand. Please make clear in your code (i.e. with comments) which parts you used for which plots/analysis.

It is highly recommended that you typeset your write-up. Illegible handwriting will not be graded.

1. **[24 points] ChIP-seq analysis.**

We provide you some ChIP-seq peak data in `peak.bed` that contains the chromosome, start and end positions of the ChiP-seq peaks from an experiment. This is real transcription factor ChIP-Seq data from a cancer cell line, and your goal is to figure out if you can distinguish what transcription factor the experiment was performed with. In doing so, you will gain familiarity with some common genomics tools. Most of the parts in this question are pretty open-ended; as long as your answers are well-supported, you will get credit.

Data Processing Steps

Using `peak.bed`, please complete the following tasks.

- Using your tool of choice, extract the first 100 lines of this file to a new file `"trimmed.bed"`.
- Load `"trimmed.bed"` in the UCSC Genome Browser (https://genome.ucsc.edu) as a custom track (use hg19 as the assembly). Get the genomic sequence for each of the intervals through UCSC Genome Browser. You can do that by going to Table Browser in UCSC Genome Browser, select custom track and load `"trimmed.bed"`, select output format as **sequence** and get output. Figure **??** shows how to do this.

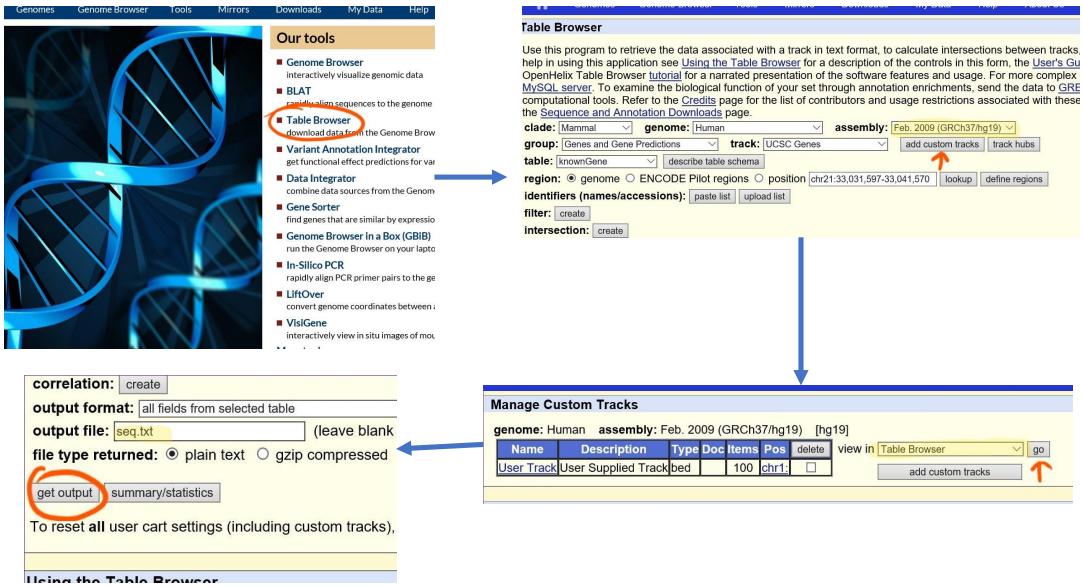

Figure 1: Use UCSC Table Browser to get the sequence.

- Use MEME (http://meme-suite.org/tools/meme) to call enriched motifs based on the output sequence. Use the default setting in MEME but only consider output motif with width from 5 to 9 (set under "Advanced options"). Note that you may still need to adjust the size of the input file to keep the number of characters within 60,000 in the file. You may want to provide your email if you want a notification for when your results are ready from MEME. This may take some time, depending on the server load.
- Using each of your hits from MEME, feed them to TOMTOM by clicking the 'submit/download' arrow to the right of your MEME results. Use all default options for TOMTOM.

Questions

(a) (2 points) Question – Screenshot the output returned by MEME. Based on the motif logos, which one is the most promising hit?

Solution

(b) (6 points) Question – What is the top hit obtained from TomTom for each of the motifs from MEME? Based on this do you agree with your previous assessment? Note that the E-value reported by TOMTOM is the Bonferroni-corrected p-value.

Solution

(c) (8 points) Question – Use the original peak.bed file and use GREAT (http://great.stanford.edu/public/html/) with default settings to identify enriched biological processes and pathways. **Be sure to use GREAT version 3.0.0, with species hg19.** Screenshot the top five Go Biological Process hits. How do these pathways compare to your expectations based on your MEME results? Would you consider these pathways statistically significant? Hint: explain what the important enrichment statistics mean.

Solution

(d) (2 points) Question – Now take a quick look at the hits in the category MSigDB Preturbation category. Explain what type of data is shown here. How much might these hits depend on what type of cancer cells the experiment was done with?

Solution

(e) (6 points) Question – Suppose that you were interested in determining what type of cancer the provided data was collected from. Based on your GREAT analyses, how useful is the provided TF ChIP-Seq data for this purpose? Justify this with some explanation: this will require that you understand what the GO terms are, and what processes your TF(s) are involved with. Then outline a follow up high throughput sequencing experiment from those mentioned in class that might help you with this new objective.

Solution

2. **[50 points] Linkage Disequilibrium**

In this problem we will use genotype sequences for the first 5,000 SNPS in mouse chromosome 1, derived from a case study involving 1,063 mice; these data are contained in the file `chromosome_1_phased_first_5000_snps.vcf` (you can read more about this file format here). Usually, after sequencing each sample's genome, you would have to phase the sequences (assign each of two alleles at an SNP site to the maternal or paternal chromosome) using a tool like Beagle. However, we have already done this for you, so you are provided as input the phased haplotype data.

Note: Your code is graded by an autograder. You have been given a code skeleton in `linkage_disequilibrium.py`. You must complete 6 functions (`split_phased_snps`, `compute_allele_frequencies`, `compute_haplotype_frequencies`, `estimate_D`, `estimate_D_prime` and `compute_r_squared`). At the end, the script should be able to run with the following command:

```
python linkage_disequilibrium.py
```

The formats of arguments and return values of each function are explicitly described in the skeleton code. For the hyperparameters, please use the **default** values specified in the skeleton code.

(a) (15 points) Complete the `split_phased_snps`, `compute_allele_frequencies` and `compute_haplotype_frequencies` functions.

The `split_phased_snps` function takes as input a Pandas DataFrame of dimensions `num_snps` $\times$ `num_samples` which contains phased haplotype data - that is, each element in the array is a string of the form

`"maternal_chromosome_allele | paternal_chromosome_allele"`

Here, a `0` indicates that the allele is the reference allele, whereas a `1` indicates that the allele is the alternate allele. For example, if an entry in the dataframe is `0 | 1`, the reference allele at that position is `G` and the alternate allele at that position is `A`, then according to the entry the maternal chromosome allele is `G` and the paternal chromosome allele is `A`. The output of this function should be a Numpy array of dimensions `2` $\times$ `num_snps` $\times$ `num_samples`, where the last dimension holds the data for the maternal and paternal chromosome separately.

The `compute_allele_frequencies` function should calculate the frequency of the reference and alternate alleles for each SNP, so the output should be a 2-tuple of Numpy arrays, each with length `num_snps`.

The `compute_haplotype_frequencies` function should calculate the frequency of each of four haplotypes (`(1, 1)`, `(1, 0)`, `(0, 1)`, and `(0, 0)`) for each pair of SNPs; thus, the output should be a 4-tuple of Numpy arrays, each with dimensions `num_snps` $\times$ `num_snps`.

Next, we would like to compute the linkage disequilibrium between every pair of SNPs. Suppose that we have two SNPs, one at locus p and the other at locus q. Define two indicator variables $P \sim \text{Bernoulli}(p_1)$ and $Q \sim \text{Bernoulli}(q_1)$ which take the value of 1 when the respective SNP contains

the reference allele and a value of 0 when it contains the alternate allele. Furthermore, let's denote

$$\begin{aligned} p_0 &= 1 - p_1 \\ q_0 &= 1 - q_1 \\ p_{11} &= \mathbb{P}(P = 1, Q = 1) \\ p_{10} &= \mathbb{P}(P = 1, Q = 0) \\ p_{01} &= \mathbb{P}(P = 0, Q = 1) \\ p_{00} &= \mathbb{P}(P = 0, Q = 0) \end{aligned}$$

Essentially, the p_{ij} indicate the probability that $P = i$ **and** that $Q = j$. In practice, can estimate these probabilities empirically from our data.

Note that, assuming the SNPs are independently inherited, we should have that $p_{ij} = p_i \cdot p_j$. Thus, we can define the *linkage disequilibrium* (D) as the difference between these two quantities. For clarity, here is a table showing the relationships between these quantities:

	Q=1	Q=0
P=1	$p_1q_1 + D = p_{11}$	$p_1q_0 - D = p_{10}$
P=0	$p_0q_1 - D = p_{01}$	$p_0q_0 + D = p_{00}$

(b) (5 points) Naively, we could estimate $D = p_{11} - p_1q_1$; however, this would not make full use of all the haplotype frequencies that we have estimated. Show that

$$D = p_{11} \cdot p_{00} - p_{10} \cdot p_{01}$$

is also a valid estimator for D.

Solution

Next, we can implement the computation of D to compare the linkage disequilibrium of different SNPs. However, notice that D can take on negative values even though frequencies cannot be negative; this makes it difficult to compare D values. To address this, we can divide D by D_{max}, defined as follows:

$$\begin{aligned} D &= p_{11} \cdot p_{00} - p_{10} \cdot p_{01} \\ D' &= |D/D_{\max}|, \text{ where} \\ D_{\max} &= \begin{cases} \min(p_1q_2, p_2q_1), & D > 0 \\ \max(-p_1q_1, -p_2q_2), & D <= 0 \end{cases} \end{aligned}$$

This adjustment guarantees that D' lies in the range $[0, 1]$.

(c) (10 points) Implement `calculate_D` and `calculate_D_prime` as described above. Attach a heatmap displaying the square matrix of D' values, with brighter colors indicating a value near 1 and darker colors indicating values near 0. From the plot, you should be able to identify linkage blocks - that is, adjacent regions of the DNA that are inherited together and show complete linkage disequilibrium (there should be anywhere from $5-10$ of these of varying sizes, depending on how stringent of a criterion you use for defining them). How big is the biggest clear linkage block, in terms of the number of SNPs spanned (a rough estimate is enough)? How small is the

smallest clear linkage block?

Hint #1: Note that the D' matrix is symemtric, so you only need to calculate its values for either the upper or lower triangular values of the matrix.

Hint #2: Make sure to vectorize your calculations as much as possible to ensure a reasonable runtime. With an efficient implementation, your code should for this part shouldn't run for longer than 10 minutes. For debugging, you can try testing on just a subset of the SNPs.

Solution

Another common measure of the linkage disequilibrium between two loci is r^2, the square of Pearson's correlation coefficient. Pearson's correlation coefficient for two variables X and Y is defined as follows:

$$r(X,Y) = \frac{\text{cov}(X,Y)}{\sigma_X \sigma_Y}, \text{ where}$$
$$\text{cov}(X,Y) = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])]$$
$$\sigma_X^2 = \mathbb{E}[(X - \mathbb{E}[X])^2]$$

In other words, $\text{cov}(X,Y)$ is the covariance of X and Y and σ_X^2 is the variance of X.

(d) (10 points) Show that, for indicator variables $P \sim \text{Bernoulli}(p_1)$ and $Q \sim \text{Bernoulli}(q_1)$,

$$r(P,Q) \approx \frac{D}{\sqrt{p_1 p_2 q_1 q_2}}$$

where $D = p_{11} - p_1 q_1$ is the linkage disequilibrium as calculated in part (b). Note that we use the naive definition of D for this proof.

Hint: Recall that the variance of a Bernoulli variable X with parameter p is $\sigma_X^2 = p(1-p)$.

Solution

(e) (10 points) Using the formula that you derived in part (c), implement `calculate_r_squared`. Attach a heatmap of the square matrix of r^2 values, similar to part (d). Compare to the heatmap in (d); qualitatively, what similarities/differences do you notice?

Solution

3. **[26 points] Hardy-Weinberg Equilibrium**

In this problem, we will leverage the phasing of the haplotype data in order to derive aspects about the structure of the population that we are working with;. Here we will use all the SNPs from chromosome 19 for the same 1,063 mice that we used in the previous problem; these data are contained in the file `chromosome_19_phased_snps.vcf`

Note: Your code is graded by an autograder. You have been given a code skeleton in `population_structure.py`. You must complete 4 functions (`compute_effective_allele_frequencies`, `compute_genotype_counts`, `calculate_chi_squared_statistic`, `detect_snps_under_selection`). At the end, the script should be able to run with the following command:

```
python hardy_weinberg.py
```

The formats of arguments and return values of each function are explicitly described in the skeleton code. For the hyperparameters, please use the **default** values specified in the skeleton code.

First, we will use the Hardy-Weinberg test. The Hardy-Weinberg test evaluates whether or not a genomic locus violates the Hardy-Weinberg equilibrium. The statistic can be written as follows

$$\chi^2 = \sum_{i=1}^{\#\text{ of categories}} \frac{(\text{observed count}_i - \text{expected count}_i)^2}{\text{expected count}_i}$$

For our application, we have three categories at each locus (homozygous dominant, heterozygous and homozygous recessive). After computing the χ^2 statistic, we can compare it to a threshold based on our significance level and the degrees-of-freedom parameter to detect SNPs under selection.

(a) (10 points) Implement the `compute_effective_allele_frequencies` and `compute_genotype_counts` functions. These will be used for computing the χ^2 statistic in part (b).

Note: For this problem, you will calculate the effective allele frequencies in a slightly different way from how you calculated them in the previous problem. In particular, if you have that c_{11} is the number of `"0" | "0"` genotypes observed, c_{12} is the number of `"0" | "1"` genotypes observed, c_{21} is the number of `"1" | "0"` genotypes observed and c_{00} is the number of `"1" | "1"`, then the frequency of the reference allele is $\frac{2c_{11}+(c_{21}+c_{12})}{2(c_{11}+c_{21}+c_{12}+c_{22})}$; the complement holds for the frequency of the alternate.

(b) (11 points) Implement the `calculate_chi_squared_statistic` and `detect_snps_under_selection` functions. Use these functions to identify SNPs that are actively under selection. Using a significance level = 0.01 and degrees-of-freedom = 1, what fraction of SNPs from the sample violate Hardy-Weinberg equilibrium?

Hint: Use `scipy.stats.distributions.chi2`'s `ppf` method to calculate the correct threshold for significance.

Solution

(c) (5 points) For a χ^2 test, we must first specify the degrees-of-freedom parameter. This parameter is the number of freely varying factors in our data, and it governs the shape of the χ^2 distribution. In applications to genotypic frequencies, despite the fact that there are three different categories, we usually use a value of 1 for the degrees-of-freedom parameter. Why is this?

Solution