The ChIP-Seq Distillery

User Manual

May 2015

Victor Hanson-Smith

victorhansonsmith@gmail.com

# Introduction

Chromatin immunoprecipitation followed by DNA sequencing, also known as ChIP-seq, is a popular experimental technique to determine genomic locations that are enriched for binding by a particular DNA-binding protein. ChIP-seq is typically used to study DNA-binding transcription regulator proteins. A common experimental approach is to perform ChIP-seq in different conditions – perhaps different cells strains, species, or environmental conditions – and then compare the effects on DNA-binding of a particular protein.

ChIP-seq methods have become well established for mapping sequenced reads to their reference genome [cite XX], and finding “peaks” with significantly enriched binding [cite XX]. However, it remains less established how binding data should be compare between replicates, across different conditions, and across species.

The ChIP-Seq Distillery automates the multi-step protocol of aligning reads to a genome, calling peaks, and then comparing those peaks between replicates or species.

Table of Contents

Introduction 2

Download and Install 4

Usage 5

Output Files 6

Troubleshooting 7

# Download and Install

The ChIP-Seq Distillery is available here: https://github.com/vhsvhs/SeqTools

APRES is written in **Python** and **R**. It uses a few commonly-used Python libraries: **scipy**, **numpy**, and **matplotlib**. You made need to install these libraries separately.

# Input Files

1. Annotation file
2. Pillars File
3. DB file (optional, will create new if no DB)

## The Annotation File

The annotation file describes your FASTQ reads. This file associates reads with known genomes, replicate IDs, and media conditions. The annotation file indicates which reads are from a untagged control experiment, and which reads are the tagged experimental observations.

The annotation file should contain a text table, described as follows. The first line should be a header row with the following columns

* sample\_tube\_name
* library\_name
* index
* file\_name
* strain
* species
  + Current Options: cdub, calb, ctrop
  + Hybrid species should be species1\_species2, e.g. “cdub\_calb”
* tr
* tag – “YES” or “NO”
* media
* condition
* replicate – an integer number
* comment

# We need to build a better

# Usage

The ChIP-Seq Distillery (CSD) is a Python script that is run from the command-line. CSD may require several hours to finish, depending on the size of your ChIP-seq project. During its run, CSD will print useful updates to the terminal.

## Command:

python run\_distill.py

## Required Parameters:

--annopath <PATH>

--datadir <PATH>

--outdir <PATH>

--dbpath <PATH>

--project\_name <NAME>

--genome\_list <PATH>

## Optional Parameters:

--use\_mpi 1

--mismatch\_thresh 1000000

--chrom\_filter <KEYWORD>

--jump <INTEGER>

--stop <INTEGER>

# The Analysis Pipeline

1. Map Reads to Genomes (Bowtie)

1.1. Check Bowtie Output

2. Extract Reads from Bowtie Output

2.1. Get Hybrid Pairs

2.3. Find Hybrid Unique Reads

2.31. Pring Reads Histograms

2.4. Write Filtered SAM files

3. Write Sorted BAM files

3.1. Check BAMS

3.2. BAM to BedGraph

3.3. Check BedGraph files

4. Run Peak Calling (MACS2)

4.1 Check Peaks

5. Calculate FE

5.1. Check FE

6. BED to WIG

6.2. Check WIG

7. Setup Visualizations

8. Create Visualizations

i. Build Visualization Database

ii. Import Pillars

iii. Resolve Alias Gene IDs

iv. Import Summits (BED) and FE (BDG)

v. Correlate Replicates, Groups, and Species

vi. Plot Replicates, Groups, and Species

**Notes from Meeting May 6th, 2015:**

Parameters to add to run\_distill.py

--mismatch\_threshold X (for post-Bowtie)

--eliminate\_multialign (skips those reads with multiple alignment positions, based on the XS flag in SAM)

--min\_qval X (for MACS2)

--peak\_style <any intergenic, peak intergenic, any exonic>

--rep\_comp\_style <INTERSECTION/UNION>

1. BOWTIE - Read mismatch threshold
   1. Eugenio: You can only play with the mismatches in the seed of the alignment, not the entire alignment.
   2. Eugenio: Eliminate reads that map to multiple locations
      1. Do this use “–q 1” in samtools view
   3. Eugenio: remove PCR artifacts, i.e. reads that are identical
      1. MACS2 does this, but the duplicates appear in the WIG file
      2. “samtools rmdup” will do this.
         1. These will create “square” edges in the peaks
2. MACS2 – shiftsize –effective genome size
   1. Let’s compute genome size from the genome list, rather than hardcoding the genome size.
   2. Let’s do a run where we double the genome size.
   3. –slocal versus –llocal
   4. Let’s build a false-positive discovery rate estimate into the pipeline.
      1. Control versus control?
      2. Mean and median Q-value for control vs. control.
      3. Let’s compare the distributions of Q-values for control versus control, and the distribution of Q-values for experimental versus control. Then do a statistical test to see if the distributions are significantly different.
3. Peaks to Genes
   1. Add an option –peak\_style for rejecting/keeping peaks.
   2. Use all the intergenic region? Versus use just the 1KB upstream from a gene.
   3. Add a histogram of peak distance to nearest gene.
4. Correlating Replicates
   1. Maybe add a criterion to evaluate overlap of peaks between species.
5. Matt: add an option to try different correlation steps
6. Eugenio: fix the PNG visualztion to color yellow last.
7. Matt: think about a way to compare between different transcription factors

Datasets:

Isabel’s NDT80 is an overexpression dataset, Eugenio’s.

Rather than doing all-by-all combinations, vary one at at time.

**Genome size – double it, and try halving it.**

Matt: use wor4 – use one where it worked well, and one where it worked less well. – Is there information in the less-well experiment that we’re missing?

Sandy: Maybe the regular expressed Ndt80 data contains some cryptic information that’s in the overexpressed Ndt80.

# Output Files

Asdfadsf

# Troubleshooting

asdfasdf