THE MISSING GUIDE

The BIOSTAR HANDBOOK

BIOINFORMATICS DATA ANALYSIS GUIDE

RNA-SEQ
ANANLYSIS

GENE
EXPRESSION

NGS
SEQUENCING

BLAST
SEARCH

GENOME
ANALYSIS

Table of Contents

Installing alignment tools

Installing short read aligners

BBMap/BBTools suite

Installing Variation calling tools

RNA-Seq analysis tools

The Biostar Handbook: A Beginner's Guide to Bioinformatics

Life scientists have been working for over fifty years to decode the information contained in DNA. Until recently, this effort has been hampered by tools and methodologies insufficient to the task. Continued advances in digital information processing are now making large-scale analyses of biological information possible. The new field of bioinformatics has emerged to apply these digital tools to questions life scientists have been asking for decades. To date, the results have been promising.

The Biostar Handbook introduces readers to bioinformatics. This new scientific discipline is positioned at the intersection of biology, computer science, and statistical data analysis. It is dedicated to the digital processing of genomic information.

The Handbook has been developed, improved, and refined over the last five years in a research university setting. It is currently used as course material in an accredited PhD training program. The contents of this book have provided the analytical foundation to hundreds of students, many of whom have become full-time bioinformaticians and work at some of the most innovative companies in the world.

Is the Handbook Available?

The handbook was released on Dec. 7th, 2016.

How will I access the Handbook?

The introductory chapters are available to the public. To read the rest of the Handbook, you will need to be logged in.

The Handbook is currently being developed and used as part of the BMMB 852: Applied Bioinformatics graduate course at Penn State. Please contact the instructor if you are enrolled in this course but have not yet received an email with your login information.

Where does the material covered in the Handbook come from?

The Handbook is based on the materials we've used to develop and teach bioinformatics and programming courses to life scientists in a research university setting. We've augmented these materials with insights gathered from a website we created and maintain called Biostars: Bioinformatics Explained.

Is the Handbook available in other formats?

We offer the Handbook in the following formats:

- Downloadable as a PDF
- Downloadable as an EBook.

What is a Biostar?

It's not a what. It's a who! And it could be you.

Just about every modern life science research project depends on large-scale data analysis. Moreover, it's the results of these data analyses that propel new scientific discoveries. Life science research projects thrive or wither by the insights of data analyses. Consequently, bioinformaticians are the stars of the show.

But make no mistake: it is a high-pressure, high-reward job where an individual's skills, knowledge, and determination are each needed to carry their team to success. This Handbook was carefully designed to give you all the skills and knowledge needed to become the star of the life science show, but the determination is up to you.

Science needs you. Be a Biostar!

How do I use the Handbook?

First of all do note the search box on the top left. Once you are more familiar with site the search becomes the simplest way to find what you are looking for.

What is currently covered in the book?

The Handbook is divided into the following sections. We cover both the foundations and their applications to realistic data analysis scenarios.

1. Bioinformatics foundations

- Data formats and repositories.
- Sequence alignments.
- Data visualization.
- Unix command line usage.

2. Bioinformatics data analysis protocols

- Genome variation and SNP calling.
- RNA-seq and gene expression analysis
- Genome Assembly (coming in 2017)
- Metagenomics (coming in 2017)
- ChIP-Seq analysis (coming in 2017)

3. Software tool usage

- Using short read aligners
- Using quality control tools
- Manipulating sequence data

The table of contents on the left allows you to jump to the corresponding sections.

Should all life scientists understand how bioinformatics operates?

Yes!

The results of bioinformatic analyses are relevant for most areas of study inside the life sciences. Even if a scientist isn't performing the analysis themselves, they need to be familiar with how bioinformatics operates so they can accurately interpret and incorporate the findings of bioinformaticians into their work.

All scientists informing their research with bioinformatic insights should understand how it works by studying its principles, methods, and limitations—the majority of which is available for you in this Handbook.

We believe that this book is of great utility even for those that don't plan to run the analysis themselves.

Was the book designed to be read from top to bottom?

This book follows a curricula that teaches practical data analysis for life scientists. We gradually introduce concepts and chapters tend to build on information covered before. For newcomers following top bottom might be the best approach. Yet not all chapters need to be followed in order -- readers may jump ahead to any topic of interest.

Is there a theme to the book?

The book explains most concepts through the task of analyzing the genomic data obtained from the 2014 Ebola virus outbreak in Africa. The data representing 99 sequenced Ebola virus genomes published in the scientific article Genomic surveillance elucidates Ebola virus origin and transmission during the 2014 outbreak is used to demonstrate the data processing and data analysis tasks that a scientist might need to undertake.

What type of computer is required?

All tools and methods presented in this book have been tested and will run on all three major operating systems: **MacOS**, **Linux** and **Windows 10**. See the Computer Setup page.

For best results Windows 10 users will need to join the Windows Insider program (a free service offered by Microsoft) that will allow them to install the newest release of "Bash Unix for Windows."

Is there data distributed with the book?

Yes, we have a separate data site at http://data.biostarhandbook.com. Various chapters will refer to content distributed from this site.

Who is the Handbook for?

The Biostar Handbook provides training and practical instructions for students and scientists interested in data analysis methodologies of genome-related studies. Our goal is to enable readers to perform analyses on data obtained from high throughput DNA sequencing instruments.

All of the Handbook's content is designed to be simple, brief, and geared towards practical application.

Is bioinformatics challenging to learn?

Bioinformatics engages the distinct fields of biology, computer science, and statistical data analysis. Practitioners must navigate the various philosophies, terminologies, and research priorities of these three domains of science while keeping up with the ongoing advances of each.

Its position at the intersection of these fields might make bioinformatics more challenging than other scientific subdisciplines, but it also means that you're exploring the frontiers of scientific knowledge, and few things are more rewarding than that!

Can I learn bioinformatics from this book?

Yes!

The questions and answers in the Handbook have been carefully selected to provide you with steady, progressive, accumulating levels of knowledge. Think of each question/answer pair as a small, well-defined unit of instruction that builds on the previous ones.

- Reading this book will teach you what bioinformatics is all about.
- Running the code will teach you the skills you need to perform the analyses.

How long will it take me to learn bioinformatics from this book?

About **100** hours.

Of course, a more accurate answer depends on your background preparation, and each person's is different. Prior training in at least one of the three fields that bioinformatics builds upon (biology, computer science, and data analysis) is recommended. The time required to master all skills also depends on how you plan to use them. Solving larger and more complex data problems will require greater skills, which need more time to develop fully.

That being said, based on several years of evaluating trainees in the field, we have come to believe that an active student would be able to perform publication quality analyses after dedicating about **100** hours of study. This is what this book is really about -- to help you put those **100** hours to good use.

What is bioinformatics?

Bioinformatics is a new, computationally-oriented Life Science domain. Its primary goal is to make sense of the information stored within living organisms. Bioinformatics relies on and combines concepts and approaches from biology, computer science, and data analysis. Bioinformaticians evaluate and define their success primarily in terms of the new insights they produce about biological processes through digitally parsing **genomic** information.

> Bioinformatics is a data science that investigates how information is *stored within* and *processed by* living organisms.

How has bioinformatics changed?

In its early days—perhaps until the beginning of the 2000s—bioinformatics was synonymous with **sequence analysis**. Scientists typically obtained just a few DNA sequences, then analyzed them for various properties. Today, sequence analysis is still central to the work of bioinformaticians, but it has also grown well beyond it.

In the mid-2000s, the so-called *next-generation*, *high-throughput* sequencing instruments (such as the Illumina HiSeq) made it possible to measure the full genomic content of a cell in a single experimental run. With that, the quantity of data shot up immensely as scientists were able to capture a snapshot of *everything* that is DNA-related.

These new technologies have transformed bioinformatics into an entirely new field of *data science* that builds on the "classical bioinformatics" to process, investigate, and summarize massive data sets of extraordinary complexity.

What subfields of bioinformatics exist?

DNA sequencing was initially valued for revealing the DNA content of a cell. It may come as a surprise to many, however, that the greatest promise for the future of bioinformatics might lie in other applications. In general, most bioinformatics problems fall under one of four categories:

1. **Assembly**: establishing the nucleotide composition of genomes
2. **Resequencing**: identifying mutations and variations in genomes
3. **Classification**: determining the species composition of a population of organisms
4. **Quantification**: using DNA sequencing to measure the functional characteristics of a cell

The Human Genome Project fell squarely in the **assembly** category. Since its completion, scientists have assembled the genomes of thousands of others species. The genomes of many millions of species, however, remain completely unknown.

Studies that attempt to identify changes relative to known genomes fall into the **resequencing** field of study. DNA mutations and variants may cause phenotypic changes like emerging diseases, changing fitness, different survival rates, etc. For example, there are several ongoing efforts to compile all variants present in the human genome—these efforts would fall into the resequencing category. Thanks to the work of bioinformaticians, massive computing efforts are underway to produce clinically valuable information from the knowledge gained through resequencing.

Living micro-organisms surround us, and we coexist with them in complex collectives that can only survive by maintaining interdependent harmony. **Classifying** these mostly-unknown species of micro-organisms by their genetic material is a fast-growing subfield of bioinformatics.

Finally, and perhaps most unexpectedly, bioinformatics methods can help us better understand biological processes, like gene expressions, through **quantification**. In these protocols, the sequencing procedures are used to determine the relative abundances of various DNA fragments that were made to correlate with other biological processes

Over the decades biologists have become experts at manipulating DNA and are now able to co-opt the many naturally-occurring molecular processes to copy, translate, and reproduce DNA molecules and connect these actions to biological processes. Sequencing has opened a new window into this world, new methods and sequence manipulations are being continuously discovered. The various methods are typically named as *???-Seq* for example *RNA-Seq*, *Chip-Seq*, *RAD-Seq* to reflect on what phenomena is being captured/connected to sequencing. For example, RNA-Seq reveals the messenger RNA by turning it into DNA. Sequencing this construct allows for simultaneously measuring the expression levels of all genes of a cell.

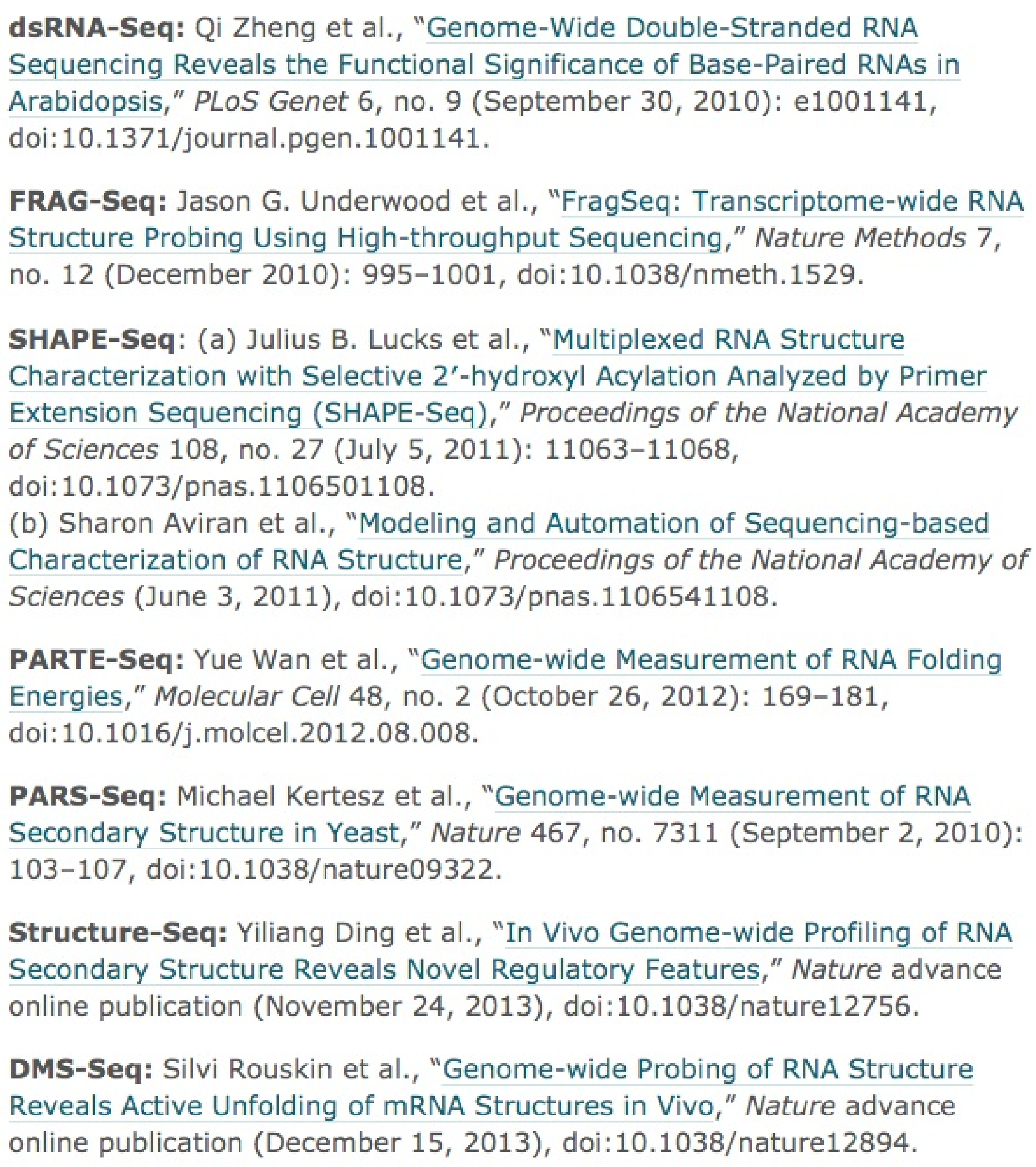
Is there a list of functional assays used in bioinformatics?

In the Life Sciences, an **assay** is an investigative procedure used to *assess* or measure the presence, amount, or function of some target (like a DNA fragment). Dr. Lior Pachter, professor of Mathematics at Caltech, maintains a list of "functional genomics" assay technologies on the page called Star-Seq.

All of these techniques fall into the **quantification** category. Each assay uses DNA sequencing to quantify another measure, and many are examples of connecting DNA abundances to various biological processes.

Notably, the list now contains nearly 100 technologies. Many people, us included, believe that these applications of sequencing are of greater importance and impact than identifying the base composition of genomes.

Below are some examples of the assay technologies on Dr. Pachter's list:



dsRNA-Seq: Qi Zheng et al., "Genome-Wide Double-Stranded RNA Sequencing Reveals the Functional Significance of Base-Paired RNAs in Arabidopsis," *PLoS Genet* 6, no. 9 (September 30, 2010): e1001141, doi:10.1371/journal.pgen.1001141.

FRAG-Seq: Jason G. Underwood et al., "FragSeq: Transcriptome-wide RNA Structure Probing Using High-throughput Sequencing," *Nature Methods* 7, no. 12 (December 2010): 995–1001, doi:10.1038/nmeth.1529.

SHAPE-Seq: (a) Julius B. Lucks et al., "Multiplexed RNA Structure Characterization with Selective 2′-hydroxyl Acylation Analyzed by Primer Extension Sequencing (SHAPE-Seq)," *Proceedings of the National Academy of Sciences* 108, no. 27 (July 5, 2011): 11063–11068, doi:10.1073/pnas.1106501108.
(b) Sharon Aviran et al., "Modeling and Automation of Sequencing-based Characterization of RNA Structure," *Proceedings of the National Academy of Sciences* (June 3, 2011), doi:10.1073/pnas.1106541108.

PARTE-Seq: Yue Wan et al., "Genome-wide Measurement of RNA Folding Energies," *Molecular Cell* 48, no. 2 (October 26, 2012): 169–181, doi:10.1016/j.molcel.2012.08.008.

PARS-Seq: Michael Kertesz et al., "Genome-wide Measurement of RNA Secondary Structure in Yeast," *Nature* 467, no. 7311 (September 2, 2010): 103–107, doi:10.1038/nature09322.

Structure-Seq: Yiliang Ding et al., "In Vivo Genome-wide Profiling of RNA Secondary Structure Reveals Novel Regulatory Features," *Nature* advance online publication (November 24, 2013), doi:10.1038/nature12756.

DMS-Seq: Silvi Rouskin et al., "Genome-wide Probing of RNA Structure Reveals Active Unfolding of mRNA Structures in Vivo," *Nature* advance online publication (December 15, 2013), doi:10.1038/nature12894.

But what is bioinformatics, *really*?

So now that you know what bioinformatics is all about, you're probably wondering what it's like to practice it day-in-day-out as a bioinformatician. The truth is, it's not easy. Just take a look at this "Biostar Quote of the Day" from Brent Pedersen in Very Bad Things:

> I've been doing bioinformatics for about 10 years now. I used to joke with a friend of mine that most of our work was converting between file formats. We don't joke about that anymore.

Jokes aside, modern bioinformatics relies heavily on file and data processing. The data sets are large and contain complex interconnected information. A bioinformatician's job is to simplify massive datasets and search them for the information that is relevant for the given study. Essentially, bioinformatics is the art of finding the needle in the haystack.

Is creativity required?

Bioinformatics requires a dynamic, creative approach. Protocols should be viewed as guidelines, not as rules that guarantee success. Following protocols by the letter is usually quite counterproductive. At best, doing so leads to sub-optimal outcomes; at worst, it can produce misinformation that spells the end of a research project.

Living organisms operate in immense complexity. Bioinformaticians need to recognize this complexity, respond dynamically to variations, and understand when methods and protocols are not suited to a data set. The myriad complexities and challenges of venturing at the frontiers of scientific knowledge always require creativity, sensitivity, and imagination. Bioinformatics is no exception.

Unfortunately, the misconception that bioinformatics is a *procedural* skill that anyone can quickly add to their toolkit rather than a scientific domain in its own right can lead some people to underestimate the value of a bioinformatician's individual contributions to the success of a project.

As observed in Core services: Reward bioinformaticians, Nature, (2015),

> Biological data will continue to pile up unless those who analyze it are recognized as creative collaborators in need of career paths.

Bioinformatics requires multiple skill sets, extensive practice, and familiarity with multiple analytical frameworks. Proper training, a solid foundation and an in-depth understanding of concepts are required of anyone who wishes to develop the particular creativity needed to succeed in this field.

This need for creativity and the necessity for a bioinformatician to think "outside the box" is what this Handbook aims to teach. We don't just want to list instructions: "do this, do that". We want to help you establish that robust and reliable foundation that will allow you to be creative when (not if) that time comes.

What are common characteristics of bioinformatics projects?

Most bioinformatics projects start out with a "standardized" plan, like the ones you'll find in this Handbook. But these plans are never set in stone. Depending on the types and features of observations and results of analyses, additional tasks will inevitably deviate from the original plan to account for variances observed in the data. Frequently, the studies need substantive customization.

As the authors of Core services: Reward bioinformaticians, Nature, (2015) have observed of their own projects,

> No project was identical, and we were surprised at how common one-off requests were. There were a few routine procedures that many people wanted, such as finding genes expressed in a disease. But 79% of techniques applied to fewer than 20% of the projects. In other words, most researchers came to the bioinformatics core seeking customized analysis, not a standardized package.

In summary, this question is difficult to answer because there isn't a "typical" bioinformatics project. It is quite common for projects to deviate from the standardized workflow.

Authors and Collaborators

The Handbook's main author and editor is Dr. Istvan Albert.

- Read more about the author.
- Personal website: https:/www.ialbert.me
- Email: istvan.albert@gmail.com

Collaborators and Contributors

- Aswathy Sebastian, MS, Bioinformatics Analyst, Bioinformatics Consulting Center, Pennsylvania State University, USA
- Reka Albert, PhD, Professor of Physics and Biology, Department of Physics, Pennsylvania State University, USA
- Jeremy Leipzig, MS, Senior Data Integration Analyst, The Children's Hospital of Philadelphia, USA
- Hemant Kelkar, PhD, Research Associate Professor, Department of Genetics and Center for Bioinformatics, University of North Carolina, USA
- Ming Tang, PhD, Computational biologist in Genomic Medicine Department, MD Anderson Cancer Center, USA
- Ram Srinivasan, MS, Bioinformatician, Icahn School of Medicine and Mount Sinai, New York, NY, USA
- Wei Shen, PhD student, Third Military Medical University, China.
- Wouter De Coster , PhD Student, University of Antwerp, Belgium
- Madelaine Gogol, BS, Programmer Analyst, Stowers Institute, Kansas City, MO, USA

Contribute to the Handbook!

Join our effort to build a comprehensive and up to date compendium for genomic data analysis.

It is a simple process that works through GitHub via simple, text based, Markdown files. The only permission we ask from authors are the rights to publish their content under our own terms (see below). Please note that this right cannot be revoked later by the author.

Authors retain re-publishing rights for the material that they are the principal author of and may re-distribute the content that they have created for this book under other terms of their choosing.

What are the licensing terms?

The Handbook's content is copyrighted material owned by Biostar Genomics LLC. Republishing any part of the Handbook is permitted only on a limited basis and must follow the terms of the Fair Use policy unless other, prior agreements have been made.

The only exception to this rule is that authors and contributors to the book retain re-publishing rights for the material that they are the principal (primary) author of and may re-distribute that content under other terms of their choosing. We define principal author as typically done in academia as the person that performed the majority of the work and is primarily responsible for its content.

What can I reuse from the book?

If you have an account on this site that means that you have been provided with a license to access, apply, modify and reuse information from the book as it applies to your own work. You may not share your login information with others or distribute the book to others.

Acknowledgements

- The Unix Bootcamp section is based on the Command-line Bootcamp by Keith Bradnam.
- The word cloud was created by Guillaume Fillion from the abstracts of Bioinformatics from 2014 (for a total around 1000 articles).

Biostar Handbook Updates

This page list the updates to the book.

Please note that you have to be logged in to access the book.

August 8th, 2017

Download: Biostar-Handbook-August-2017.zip

A chapter on Metagenomics has been added

1. Introduction to metagenomics
2. How to analyze metagenomics data
3. Taxonomies and classification
4. Microbial sequence data
5. Classifying 16S sequences
6. Human Metagenome Demonstration Projects
7. Classifying whole-genome sequences
8. A realistic sample: Alaska Oil Reservoir
9. Tool installation

July 4th, 2017

Download: Biostar-Handbook-July-2017.zip

Extensive editing of various chapters. Special thanks to Madelaine Gogol and Paige M. Miller

April 6, 2017

Download: Biostar-Handbook-April-2017.zip

A chapter on Chip-Seq analysis has been added:

1. ChIP-Seq introduction
2. ChIP-Seq alignments
3. ChIP-Seq peak calling
4. ChIP-Seq motifs
5. ChIP-Seq analysis
6. ChIP-Seq downstream 1
7. ChIP-Seq downstream 2

January 28, 2017

Download: Biostar-Handbook-January-2017.zip

Added the following sections:

1. What is sequence assembly
2. Assembly basics
3. GAGE: Evaluation of genome assemblies
4. Genome assembly terminology
5. How to set parameters
6. The lucky bioinformatician's guide to genome assembly
7. How to perform a genome assembly
8. Installing assembly software

In addition to the new chapter editors have performed an expansive proofreading and editing the prior content. Special thanks to Madelaine Gogol and Ram Srinivasan for their effort.

Added a better web search interface that highlights the matches.

December 14, 2016

Archive contains PDF, MOBI and EPUB formats.

Download: biostar-handbook-14-12-2016.zip

- Added the RNA-Seq Puzzle.
- Started section on Interval Data Types.
- Added more information on dealing with anxiety in data analysis.
- Proofreading and many other stylistic changes.

December 5, 2016

The first version of the book is released.

ChIP-Seq analysis

The author of this guide is Ming Tang. The material was developed for the Biostar Handbook.

- GitHub profile
- Diving into Genetics and Genomics

What is ChIP-seq?

Chromatin immunoprecipitation (ChIP) followed by high-throughput DNA sequencing (ChIP-seq) is a technique to map genome-wide transcription factor binding sites and histone-modification enriched regions.

Briefly, DNA bounding proteins and DNA (Chromatin) are cross-linked by formaldehyde and the chromatin is sheared by sonication into small fragments (typically 200 ~ 600 bp). The protein-DNA complex is immnuoprecipitated by an antibody specific to the protein of interest. Then the DNA is purified and made to a library for sequencing. After aligning the sequencing reads to a reference genome, the genomic regions with many reads enriched are where the protein of interest bounds. ChIP-seq is a critical method for dissecting the regulatory mechanisms of gene expression.

What are the processing steps for ChIP-seq data?

The general processing steps are as following:

1. Quality control of your `fastq` read files using tools such as FASTQC. Read our previous section: Visualizing sequencing data quality.
2. Aligning fastq reads to reference genome. The most popular read aligners are `bwa` and `bowtie`. Read section Short Read Aligners. `bowtie` has two versions: `bowtie1` and `bowtie2`. `bowtie2` is better for reads length greater than 50bp. It is still very common to use 36bp single-end sequencing library for ChIP-seq, so `bowtie1` is preferred for ChIP-seq with short reads.
3. Calling peaks from the alignment bam files.
4. Visualizing the resulting peak files and raw signal files.

What are IgG control and input control for ChIP-seq?

Like any other experiments, a control is needed for ChIP-seq. There are two kinds of controls for ChIP-seq: IgG control and input control. IgG control is DNA resulting from a "mock" ChIP with Immunoglobulin G (IgG) antibody, which binds to non-nuclear antigen; input control is DNA purified from cells that are cross-linked, fragmented, but without adding any antibody for enrichment.

One problem with IgG control is that if too little DNA is recovered after immunoprecipitation(IP), sequencing library will be of low complexity and binding sites identified using this control could be biased. Input DNA control is ideal in most of the cases. It represents all the chromatin that was available for IP. Read this biostars post for discussion.

Data sets

To illustrate all the analysis steps, we are going to use a public data set from two papers:

- Genome-wide association between YAP/TAZ/TEAD and AP-1 at enhancers drives oncogenic growth. Francesca Zanconato et.al. 2014. *Nature Cell Biology*. We will use transcription factor YAP1 ChIP-seq data in MDA-MB-231 breast cancer cells from this paper.
- Nucleosome positioning and histone modifications define relationships between regulatory elements and nearby gene expression in breast epithelial cells. Suhn Kyong Rhie et.al. 2014. *BMC Genomics*. We will use H3K27ac ChIP-seq data in MDA-MB-231 cells from this paper.
- The data sets can be found with accession number GSE66081 and GSE49651.

How to obtain the data?

Read our previous section: Accessing the Short Read Archive (SRA).

Prepare folders and download the data.

```
# go to your home directory
cd ~
# make folders for this example
mkdir -p ChIP-seq/{data,results,scripts,software}
mkdir -p ChIP-seq/results/{bams,peaks,fastqc,motifs}
cd ChIP-seq/data

# for YAP1. It takes time, alternatively download fastq directly from EGA https://www.ebi.ac.u
k/ega/home
fastq-dump SRR1810900

# for H3K27ac
fastq-dump SRR949140

# for input DNA
fastq-dump SRR949142
```

In general, do not change the names of files. This can be error prone, especially when you have many samples. Instead, prepare a name mapping file, and interpret the final results with the mapped names. For demonstrating purpose, I will re-name the files to more meaningful names.

```
mv SRR1810900.fastq YAP1.fastq
mv SRR949140.fastq H3K27ac.fastq
mv SRR949142.fastq Input.fastq
```

Now, run `FASTQC` on all the samples. Read previous section on looping over files. or use GNU parallel:

```
# always specify -j to not abuse the computing cluster
find *fastq | parallel -j 4 fastqc {} -o ../results/fastqc
```

Overall, the sequencing qualities are good for all three samples, I will skip trmming low quality bases, and go ahead with the original fastq files for aligning.

How to align ChIP-seq reads?

once you have finished QC confirming the sequencing reads are of high quality and with no adaptor contamination, you can do:

```
# use 10 threads to speed up. should finish within 10 mins
# change the path to the bowtie1 index in your own machine.
bowtie -p 10 --best --chunkmbs 320 /risapps/reference/bowtie1/hg19 -q YAP1.fastq -S ../results
/bams/YAP1.sam

# reads processed: 24549590
# reads with at least one reported alignment: 19664247 (80.10%)
# reads that failed to align: 4885343 (19.90%)
Reported 19664247 alignments to 1 output stream(s)

bowtie -p 10 --best --chunkmbs 320 /risapps/reference/bowtie1/hg19 -q H3K27ac.fastq -S ../resu
lts/bams/H3K27ac.sam

# reads processed: 29759754
# reads with at least one reported alignment: 29100444 (97.78%)
# reads that failed to align: 659310 (2.22%)
Reported 29100444 alignments to 1 output stream(s)

bowtie -p 10 --best --chunkmbs 320 /risapps/reference/bowtie1/hg19 -q Input.fastq -S ../result
s/bams/Input.sam

# reads processed: 18811563
# reads with at least one reported alignment: 18127743 (96.36%)
# reads that failed to align: 683820 (3.64%)
Reported 18127743 alignments to 1 output stream(s)
```

For demonstrating purpose, I will write down explicitly all the commands, but the GNU parallel trick can be used to save you some typing. In the end, I will introduce a shell script to chain all the steps together and show you how to make reusable codes.

Convert `sam` to sorted `bam` :

```
# avoid writing unsorted bam to disk
# note that samtools sort command changed invoke pattern after version 1.3 https://www.biostar
s.org/p/169745/
# samtools view and index will be multi-thread soon https://github.com/samtools/htslib/pull/397

cd ../results/bams
samtools view -bS YAP1.sam | samtools sort -@ 4 - -T YAP1 -o YAP1.sorted.bam
# index the sorted bam
samtools index YAP1.sorted.bam

samtools view -bS H3K27ac.sam | samtools sort -@ 4 - -T H3K27ac -o H3K27ac.sorted.bam
samtools index H3K27ac.sorted.bam

samtools view -bS Input.sam | samtools sort -@ 4 - -T Input -o Input.sorted.bam
samtools index Input.sorted.bam

# remove sam files to save space , rm can be dangerous, use rm -i if necessary
rm *sam
```

How do I call peaks?

MACS is short for Model Based Analysis for ChIP-Seq data, which was developed in Sheirly Liu's lab at Harvard by Tao Liu. It is one of the most popular peak-calling algorithums used in literatures. First, you need to install MACS.

```
## install through pip
pip install macs

# peak calling for human samples, with default p value 1e-5
macs -t YAP1.sorted.bam -c Input.sorted.bam -n YAP1 -g hs -p 1e-5 -o ../peaks/

## peak calling for H3K27ac
macs -t H3K27ac.sorted.bam -c Input.sorted.bam -n H3K27ac -g hs -p 1e-5 -o ../peaks/
```

In addition to `bam` files, `macs` can take in other input file formats such as `bed` and `sam`. type `macs` on your terminal and press `Enter` to see full list of helps.

Parameter settings can be different depending on your own data sets. There is no one-fit-all settings. You may want to load the called peaks in to `IGV` and visually inspect them.

Note: Tao Liu is developing MACS2. there is a `-broad` option to call broad peaks for histone modification ChIP-seq(e.g. H3K36me3). However, in my experience, `macs14` is still widely used and performs very well.

Take a look at How to set MACS2 peak calling parameters.

How do I call super-enhancers?

The fancy "super-enhancer" term was first introduced by Richard Young's lab in Whitehead Institute. Basically, super-enhancers are clusters of enhancers (most commonly defined by H3K27ac peaks) stitched together if they are within 12.5kb apart. The concept of super-enhancer is NOT new. One of the most famous example is the Locus Control Region (LCR) that controls the globin gene expression, and this has been known for decades. If you are interested in the controversy, please read:

A review in Nature Genetics What are super-enhancers? Another comment in Nature Genetics Is a super-enhancer greater than the sum of its parts?

From the HOMER page **How finding super enhancers works:**

> Super enhancer discovery in HOMER emulates the original strategy used by the Young lab. First, peaks are found just like any other ChIP-Seq data set. Then, peaks found within a given distance are 'stitched' together into larger regions (by default this is set at 12.5 kb). The super enhancer signal of each of these regions is then determined by the total normalized number reads minus the number of normalized reads in the input. These regions are then sorted by their score, normalized to the highest score and the number of putative enhancer regions, and then super enhancers are identified as regions past the point where the slope is greater than 1.

`HOMER` is a very nice tool for finding enriched peaks, doing motif analysis and many more. It requires making a tag directory first. `ROSE` is the tool from Richard Young's lab. Since it works directly with `bam` files, we will use it for demonstration.

Download `ROSE` to the `software` folder.

```
cd ../../software/
wget https://bitbucket.org/young_computation/rose/get/1a9bb86b5464.zip
unzip 1a9bb86b5464.zip
cd young_computation-rose-1a9bb86b5464/

# there are multiple python scripts in the folder.
# one has to run ROSE inside the ROSE folder.
python ROSE_main.py -g hg19 -i ../../results/peaks/H3K27ac_peaks.bed -r ../../results/bams/H3K
27ac.sorted.bam -c ../../results/bams/Input.sorted.bam -o ../../results/peaks
```

How do I get a normalized `bigWig` track for visualizing the raw signal?

After you get the peak calls, we want to inspect the quality of the peaks by visulizing them in a browser such as `IGV` along with the raw signal track. We will use `bigWig` track for this purpose.

From the UCSC help:

> The bigWig format is for display of dense, continuous data that will be displayed in the Genome Browser as a graph. BigWig files are created initially from wiggle (wig) type files, using the program wigToBigWig. The resulting bigWig files are in an indexed binary format. The main advantage of the bigWig files is that only the portions of the files needed to display a particular region are transferred to UCSC, so for large data sets bigWig is considerably faster than regular wiggle files

Make sure you understand the other two closely related file formats: bedgraph and wig file.

`macs14` can generate a `bedgraph` file by specifying `--bdg` , but the resulting `bedgraph` is **NOT** normalized to sequencing depth. Instead, we are going to use another nice tool `deeptools` for this task. It is a very versatile tool and can do many other things.

Make a bigWig file:

```
# install deeptools
conda install -c bioconda deeptools

# normalized bigWig with Reads Per Kilobase per Million mapped reads (RPKM)
bamCoverage -b YAP1.sorted.bam --normalizeUsingRPKM --binSize 30 --smoothLength 300 -p 10 --ex
tendReads 200 -o YAP1.bw

bamCoverage -b H3K27ac.sorted.bam --normalizeUsingRPKM --binSize 30 --smoothLength 300 -p 10 -
-extendReads 200 -o H3K27ac.bw

bamCoverage -b Input.sorted.bam --normalizeUsingRPKM --binSize 30 --smoothLength 300 -p 10 --e
xtendReads 200 -o Input.bw
```

I set `--extendReads` to 200bp which is the fragment length. Why should we extend the reads? Because in a real ChIP-seq experiment, we fragment the genome into small fragments of ~200bp, and pull down the protein bound DNA with antibodies. However, we only sequence the first 36bp(50bp, or 100bp depending on your library) of the pull-down DNA. To recapitulate the real experiment, we need to extend it to the fragment size.

You can read more details Why do we need to extend the reads to fragment length/200bp?

How do I put all the steps together?

Shell script comes to rescue

Up until now, we have finished the basic analyzing steps for ChIP-seq. We aligned the reads to get `bam` files; we called peaks using `macs` ; we generated `bigWig` tracks as well. This is great, but one of the stumbling blocks for beginners is learning how to reuse the commands we have typed. We do not want to type the same command time and time again changing the file names only when new data come in.

The strategy is to write a shell script to chain all the steps together. see the section writing scripts in the handbook.

First, you need to think about how the workflow should be. We will need to download the `fastq` files and convert them to `bam` files; next, we will call peaks for a pair of IP and Input. Sometimes, you may get sorted `bam` files, which can be used directly to call peaks. It is reasonable that we have two shell scripts for each task, respectively.

```
#! /bin/bash

set -euo pipefail

# the SRA number to work with
SRA=$1

# the reference file path, change to where you put the reference
REF="/risapps/reference/bowtie1/hg19"

##extract the fastq files
fastq-dump $SRA

## step1, quality control of the fastq files
fastqc "${SRA}".fastq

## step2, algin the reads with bowtie
bowtie -p 10 --best --chunkmbs 320 $REF -q "${SRA}".fastq -S "${SRA}".sam

## step3, convert sam to bam, and index the bam
samtools view -bS "${SRA}".sam | samtools sort -@ 4 - -T "${SRA}" -o "${SRA}".sorted.bam
samtools index "${SRA}".sorted.bam

# remove sam to save space
rm "${SRA}".sam
```

For more on `set -euo pipefail`, read this post Use the Unofficial Bash Strict Mode (Unless You Looove Debugging).

Save it to `sra2bam.sh`. Make it executable `chmod u+x sra2bam.sh`.

execute:

```
# YAP1
./sra2bam.sh SRR1810900

# H3K27ac
./sra2bam.sh SRR949140

# Input
./sra2bam.sh SRR949142
```

Now, with the `sra2bam.sh` script, it saves you from typing different file names for aligning `fastqs` and converting `sam` files. Moreover, it can be used to process any number of `sra` files.

If you have a `sra_id.txt` file with one `sra` id on each line, you can:

```
## run 6 jobs in parallel
cat sra_ids.txt | parallel -j 6 ./sra2bam {}
```

to process all of them.

second shell script `bam2peaks.sh`:

```
#! /bin/bash

set -euo pipefail

IP_bam=$1
Input_bam=$2
oprefix=$(basename "${IP_bam}" .sorted.bam)

macs -t "${IP_bam}" -c "${Input_bam}" -n "${oprefix}" -g hs -p 1e-5 -o ${oprefix}_peaks
```

The `sra2bam.sh` script will generate the indexed `bam` files, those `bam` files can be fed into `bam2peaks.sh` to call peaks. In our example data set, we only have two IPs and one Input. We can call peaks by:

```
# call peaks for H3K27ac
./bam2peaks.sh SRR949140.sorted.bam SRR949142.sorted.bam

# call peaks for YAP1
./bam2peaks.sh SRR1810900.sorted.bam SRR949142.sorted.bam
```

Imagine we have a lot of bam files generated by `sra2bam.sh`, and we want to call peaks for them, how should we stream the workflow?

Because it invovles a pair of files (IP and Input control) for calling peaks, we need to make a tab delimited `txt` file with pairs of file names on each line.

```
cat bam_names.txt
SRR949140.sorted.bam    SRR949142.sorted.bam
SRR1810900.sorted.bam    SRR949142.sorted.bam
```

Now, we can loop over the `bam_names.txt` file one by one and call peaks:

```
cat bam_names.txt | while read -r IP Input
do
    ./bam2peaks.sh "${IP}" "${Input}"
done
```

Arguments handling for shell script

What we have so far is all good, but what if you want to make your script more flexiable? e.g. you want to specify mouse genome for mapping reads. You can add arguments for your script.

```
#! /bin/bash

set -euo pipefail

# show help
show_help(){
cat << EOF
  This is a wrapper to align fastq reads to bam files for ChIP-seq experiments.
  usage: ${0##*/} -d < a directory path containing the fastq.gz files > -r < h or m>
        -h display this help and exit
        -f the full path name of the fastq file
        -r reference genome to be used. m for mouse; h for human
EOF
}

## if there are no arguments provided, show help
if [[ $# == 0 ]]; then show_help; exit 1; fi

## parsing arguments
while getopts ":hf:r:" opt; do
  case "$opt" in
    h) show_help;exit 0;;
    f) fq=$OPTARG;;
    r) REF=$OPTARG;;
    '?') echo "Invalid option $OPTARG"; show_help >&2; exit 1;;
  esac
done

## set up some defaults
REF=${REF:-"h"}

## check if the fastq file exist
if [ ! -f "$fq" ]; then
  echo "file ${fq} does not exit"
  exit 1
fi

## reference genome path for mouse and human
human_ref="/risapps/reference/bowtie1/hg19"
mouse_ref="/risapps/reference/bowtie1/mm9"

## which species? human or mouse?
if [[ $REF == "m" ]]; then
  ref_genome="${mouse_ref}"
elif [[ $REF == "h" ]]; then
  ref_genome="${human_ref}"
else
  echo "please only specify m or h for the reference genome"
  exit 1
fi

## extract output name
oprefix=$(basename "${fq}" .fastq)

## mapping
bowtie -p 10 --best --chunkmbs 320 ${ref_genome} -q "${fq}" -S "${oprefix}".sam

## convert sam to sorted bam
samtools view -bS "${oprefix}".sam | samtools sort -@ 4 - -T "${oprefix}" -o "${oprefix}".sort
ed.bam

# index sorted bam
samtools index "${oprefix}".sorted.bam

rm "${oprefix}".sam
```

With this script, you can map not only ChIP-seq data for human but also for mouse. You can surely add more arguments to set `bowtie` mapping thread numbers and `samtools sort` thread numbers (the above script uses 10 and 4 respectively). You can read more on arguments handling for shell scripts: small getopts tutorial.

Congratulations! You have come this far with me. If you want to take the next level of making your analysis reproducible and flexiable, read our Makefile section.

What are the black-listed regions?

For ChIP-seq, it is important to filter artifact regions that has abnomral high signals. From the website of Anshul Kundaje in Stanford University:

> These regions are often found at specific types of repeats such as centromeres, telomeres and satellite repeats. It is especially important to remove these regions that computing measures of similarity such as Pearson correlation between genome-wide tracks that are especially affected by outliers

One can download the `hg19` blacklist:

```
wget https://www.encodeproject.org/files/ENCFF001TDO/@@download/ENCFF001TDO.bed.gz

# 411 regions are black-listed
zless ENCFF001TDO.bed.gz | wc -l
411
```

Note that more and more sequencing projects are moving their reference genome to the latest version `GRCh38` . `GRCh38` blacklist was uploaded by Anshul Kundaje on 2016-10-16. The file can be downloaded by:

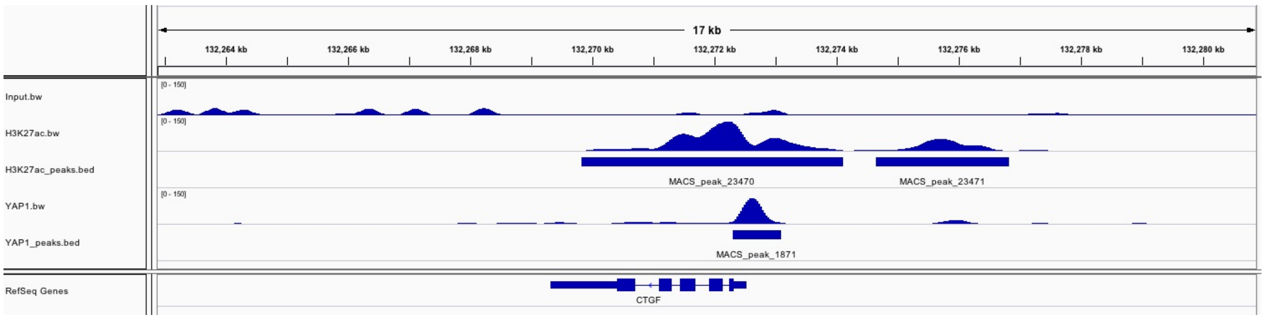
```
wget https://www.encodeproject.org/files/ENCFF419RSJ/@@download/ENCFF419RSJ.bed.gz
```

How do I visualize the peaks?

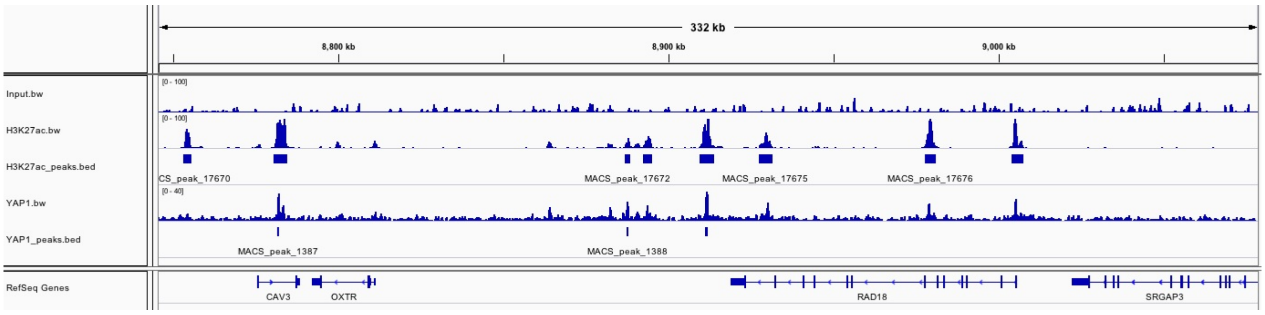
We will use `IGV` to visualize the peaks and raw signal. check our previous section on using IGV.

Open `IGV` , click `File` , then `Load From File` , choose the peak files end with `bed` and the raw signal files end with `bw` . you can change the scale of the `bigWig` tracks by right click the tracks and choose `Set Data Range` . Now, you can browser through the genome or go to your favoriate genes.

YAP1 binds to known target gene *CTGF*:



one other example from the original *Nature Cell Biology* paper at the *RAD18* locus:



We see that `macs` did a pretty good job in identifying enriched regions for histone modifications and transcription factor (TF) binding peaks. However, we do also see some potential YAP1 binding sites are missed. Fine tuning the peak calling parameters may improve the results.

ChIP-Seq Downstream Analysis 1

The author of this guide is Ming Tang. The material was developed for the Biostar Handbook.

- GitHub profile
- Diving into Genetics and Genomics

How do I compare the peak sets?

Now, we have the peaks called for H3K27ac and YAP1 and we want to exclude the peaks that overlap with the blacklisted regions. How should you do it? We will use a very popular tool `bedtools` developed by Aaron Quinlan lab for this purpose. Documentation of `bedtools` is of top niche. Also read Interval analysis tools section in the handbook.

There are many sub-commands for `bedtools`, but the most commonly used one is `bedtools intersect`:

```
# make sure you are inside the folder containing the peak files and the black-listed region fi
le.

# unzip the file
gunzip ENCFF001TDO.bed.gz

# How many H3K27ac peaks overlap with black-listed regions?
bedtools intersect -a H3K27ac_peaks.bed -b ENCFF001TDO.bed -wa | wc -l
#14

# exclude those peaks
bedtools intersect -a H3K27ac_peaks.bed -b ENCFF001TDO.bed -v > H3K27ac_filtered_peaks.bed

# do the same for YAP1
bedtools intersect -a YAP1_peaks.bed -b ENCFF001TDO.bed -v > YAP1_filtered_peaks.bed
```

How many YAP1 peaks overlap with H3K27ac peaks?

```
bedtools intersect -a YAP1_filtered_peaks.bed -b H3K27ac_filtered_peaks.bed -wa | wc -l
#1882

bedtools intersect -a YAP1_filtered_peaks.bed -b H3K27ac_filtered_peaks.bed -wa | sort | uniq |
 wc -l
#1882

bedtools intersect -a H3K27ac_filtered_peaks.bed -b YAP1_filtered_peaks.bed -wa | wc -l
#1882

bedtools intersect -a H3K27ac_filtered_peaks.bed -b YAP1_filtered_peaks.bed -wa | sort | uniq |
 wc -l
#1772
```

what's happening here? why there are only 1772 unique H3K27ac peaks overlap with YAP1 peaks?

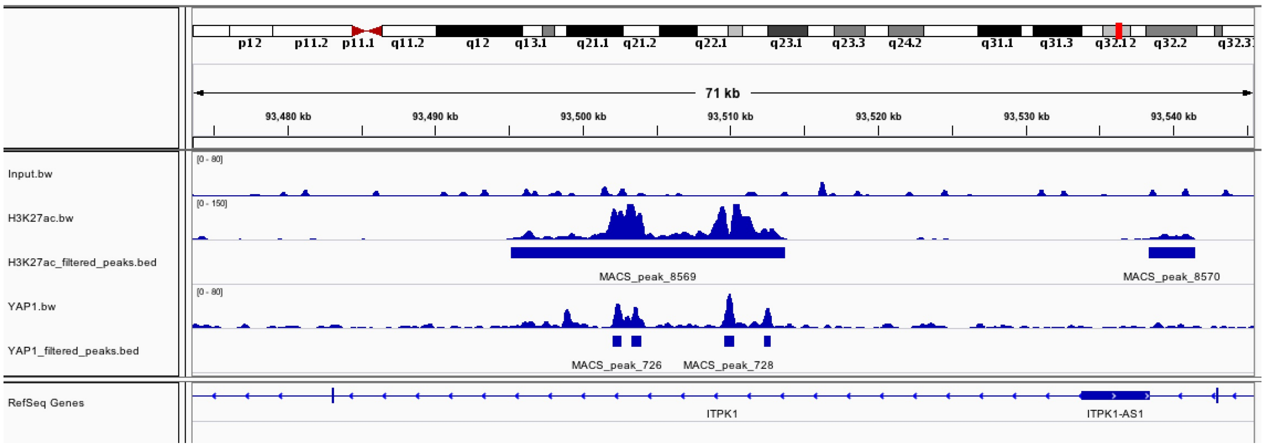
It turns out that bedtools will output the overlapping peaks of H3K27ac whenever there is an overlapping with YAP1 peaks, and it is possible that the same H3K27ac peak (tends to be really broad peaks) overlaps with multiple YAP1 peaks. With that in mind, I always do `sort | uniq` following `bedtools` command.

You can identify those H3K27ac peaks by:

```
bedtools intersect -a H3K27ac_filtered_peaks.bed -b YAP1_filtered_peaks.bed -wa | sort | uniq
-c | sort -k1,1nr | head
      4 chr14     93495168     93513756    MACS_peak_8569     935.17
      4 chr20     30276949     30312747    MACS_peak_16217     3100.00
      3 chr10     95216260     95244053    MACS_peak_3871     3100.00
      3 chr11     65654426     65689186    MACS_peak_5045     3100.00
      3 chr14     23441596     23453313    MACS_peak_7876     3100.00
      3 chr1     86040764     86052311    MACS_peak_1241     3100.00
      3 chr1     86070388     86080738    MACS_peak_1243     2714.64
      3 chr19     13943280     13955945    MACS_peak_13069     3100.00
      3 chr19     13956264     13965827    MACS_peak_13070     3100.00
      3 chr1     95080161     95091780    MACS_peak_1347     3038.66
```

We see some H3K27ac peaks can overlap with up to 4 peaks of YAP1.

If you choose to believe your eyes, let's visualize it in `IGV` :



One of the under-appreciated tools is `BEDOPS`. It has many nice features as well and the documentaion is also very good. Read this biostar post: Question: Bedtools Compare Multiple Bed Files?

How do I annotate my peaks?

The next obvious question is where are the peaks in terms of their genomic context. e.g. What genes are the peaks associated with? Are some of the peaks located in intergenic region? You need to **annotate** your peaks. There are many tools you can use. You can find a list here.

I will show you how to do peak annotation using an R bioconductor package `ChIPseeker` develeped by Guangchuan Yu.

```
> # load the packages, install them following the instructions in the links above
> library(ChIPseeker)
> library(TxDb.Hsapiens.UCSC.hg19.knownGene)
> library(rtracklayer)
> library("org.Hs.eg.db")
```

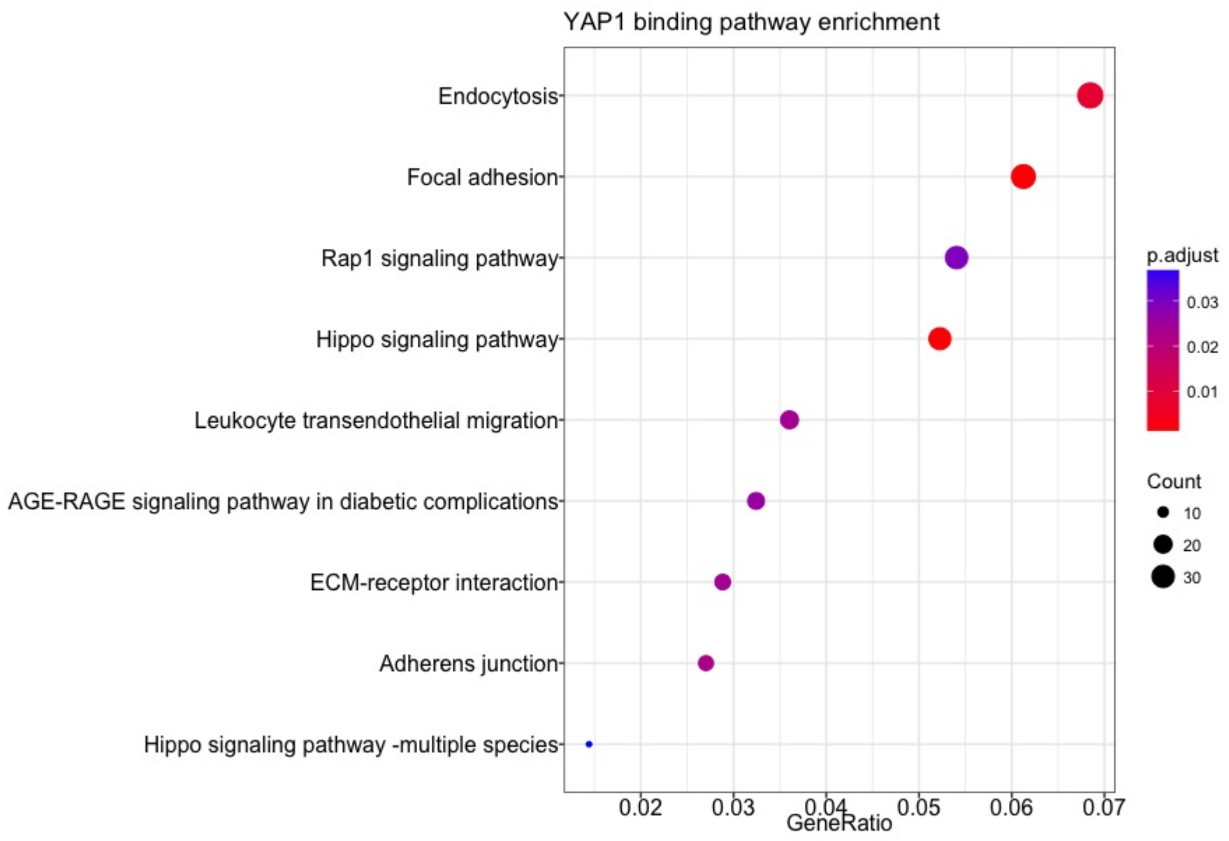
```
>
> txdb <- TxDb.Hsapiens.UCSC.hg19.knownGene

> # read in the peaks for YAP1
> YAP1<- import("YAP1_peaks.bed", format = "BED")
> YAP1
GRanges object with 2150 ranges and 2 metadata columns:
         seqnames                 ranges strand   |          name     score
            <Rle>              <IRanges>  <Rle>   |    <character> <numeric>
     [1]     chr1   [  959961,   960483]      *   |    MACS_peak_1    132.10
     [2]     chr1   [ 1295901,  1296269]      *   |    MACS_peak_2     59.44
     [3]     chr1   [ 1992556,  1993176]      *   |    MACS_peak_3     98.23
     [4]     chr1   [ 8963802,  8964418]      *   |    MACS_peak_4    126.02
     [5]     chr1   [16499970, 16500516]      *   |    MACS_peak_5    142.50
     ...      ...                    ...    ... ...          ...       ...
  [2146]     chrX [102911475, 102912040]      *   | MACS_peak_2146     73.47
  [2147]     chrX [103168254, 103168801]      *   | MACS_peak_2147     58.64
  [2148]     chrX [115403519, 115404015]      *   | MACS_peak_2148     50.08
  [2149]     chrX [149252969, 149253470]      *   | MACS_peak_2149     61.44
  [2150]     chrX [153147396, 153148074]      *   | MACS_peak_2150     61.75
  -------
  seqinfo: 23 sequences from an unspecified genome; no seqlengths

> YAP1_anno<- annotatePeak(YAP1, tssRegion=c(-3000, 3000),
                       TxDb=txdb, level = "gene", annoDb="org.Hs.eg.db",
                       sameStrand = FALSE, ignoreOverlap = FALSE,
                       overlap = "TSS")
> # some nice visualization you can do
> plotAnnoPie(YAP1_anno)
> upsetplot(YAP1_anno, vennpie=TRUE)

> # check the annotation
>  head(as.data.frame(YAP1_anno))
  seqnames    start      end width strand        name  score
1     chr1   959961   960483   523      * MACS_peak_1 132.10
2     chr1  1295901  1296269   369      * MACS_peak_2  59.44
3     chr1  1992556  1993176   621      * MACS_peak_3  98.23
4     chr1  8963802  8964418   617      * MACS_peak_4 126.02
5     chr1 16499970 16500516   547      * MACS_peak_5 142.50
6     chr1 17454948 17455379   432      * MACS_peak_6  59.60
                                 annotation geneChr geneStart  geneEnd
1 Intron (uc001ack.2/375790, intron 2 of 35)       1    955503   991499
2                          Promoter (<=1kb)       1   1288071  1297157
3   Intron (uc001aiq.3/5590, intron 4 of 17)       1   1981909  2116834
4                         Distal Intergenic       1   8921059  8939151
5                         Distal Intergenic       1  16450832 16482582
6                         Distal Intergenic       1  17393256 17445948
  geneLength geneStrand geneId distanceToTSS         ENSEMBL SYMBOL
1      35997          1 375790          4458 ENSG00000188157   AGRN
2       9087          2  54587           888 ENSG00000162576  MXRA8
3     134926          1   5590         10647 ENSG00000067606  PRKCZ
4      18093          2   2023        -24651 ENSG00000074800   ENO1
5      31751          2   1969        -17388 ENSG00000142627  EPHA2
6      52693          2  11240         -9000 ENSG00000117115  PADI2
                            GENENAME
1                              agrin
2     matrix-remodelling associated 8
3               protein kinase C zeta
4                  enolase 1, (alpha)
5                     EPH receptor A2
```

```
6 peptidyl arginine deiminase, type II

> # you can save it to a txt file to your computer
> write.table(as.data.frame(YAP1_anno), "YAP1_peaks_anno.txt", row.names =F, col.names=T, sep =
"\t", quote = F)
```

How do I do pathway enrichment analysis for the peaks?

Now, you have the genes that are associated with the peaks, you can do pathway enrichment analysis using the genes. I will use the `clusterProfiler` package developed by Guangchuan Yu as well:

```
library(clusterProfiler)

## GO term enrichment
ego <- enrichGO(gene          = as.data.frame(YAP1_anno)$SYMBOL,
                 OrgDb         = org.Hs.eg.db,
                 keytype       = "SYMBOL",
                 ont           = "BP",
                 pAdjustMethod = "BH",
                 pvalueCutoff  = 0.01,
                 qvalueCutoff  = 0.05)
# visualization
dotplot(ego, showCategory = 20)


## Kegg pathway enrichment, need the Entrez ID
kk<- enrichKEGG(gene    = as.data.frame(YAP1_anno)$geneId,
           organism     = 'hsa',
           pvalueCutoff = 0.05)

dotplot(kk, showCategory = 20, title = "YAP1 binding pathway enrichment")

## you can write the result to a tsv file
write.table(kk@result, "YAP_KEGG_pathway_genes.txt", sep = "\t", col.names = T, row.names = F,
 quote =F)
```



Note that we used all the genes that associated with a peak for the pathway analysis. This may not be optimal, especially when you have a lot of peaks. One alternative is that you can filter the genes/peaks by some rules first and then do the enrichment analysis.

e.g.

```
library(dplyr)
## only consider genes within 5000 bp of the peaks
as.data.frame(YAP1_anno) %>% dplyr::filter(abs(distanceToTSS) < 5000)

## or you can rank the peaks by the Pvalue and choose the top 1000 peaks (this number is arbit
rary..)
# score column is the -10*log10 Pvalue
as.data.frame(YAP1_anno) %>% dplyr::arrange(desc(score)) %>% head(n =1000)
```

The underlying mechanism is to compare the peaks associated gene set with the vairous annotated pathway gene sets to see whether the peak assoicated genes are overpresented in any of the known gene sets using hypergeometric test. Of course, many other complex algorithums have been developed for this type of analysis. Further reading: A Comparison of Gene Set Analysis Methods in Terms of Sensitivity, Prioritization and Specificity

GREAT predicts functions of cis-regulatory regions

How do I do motif analysis with the peaks?

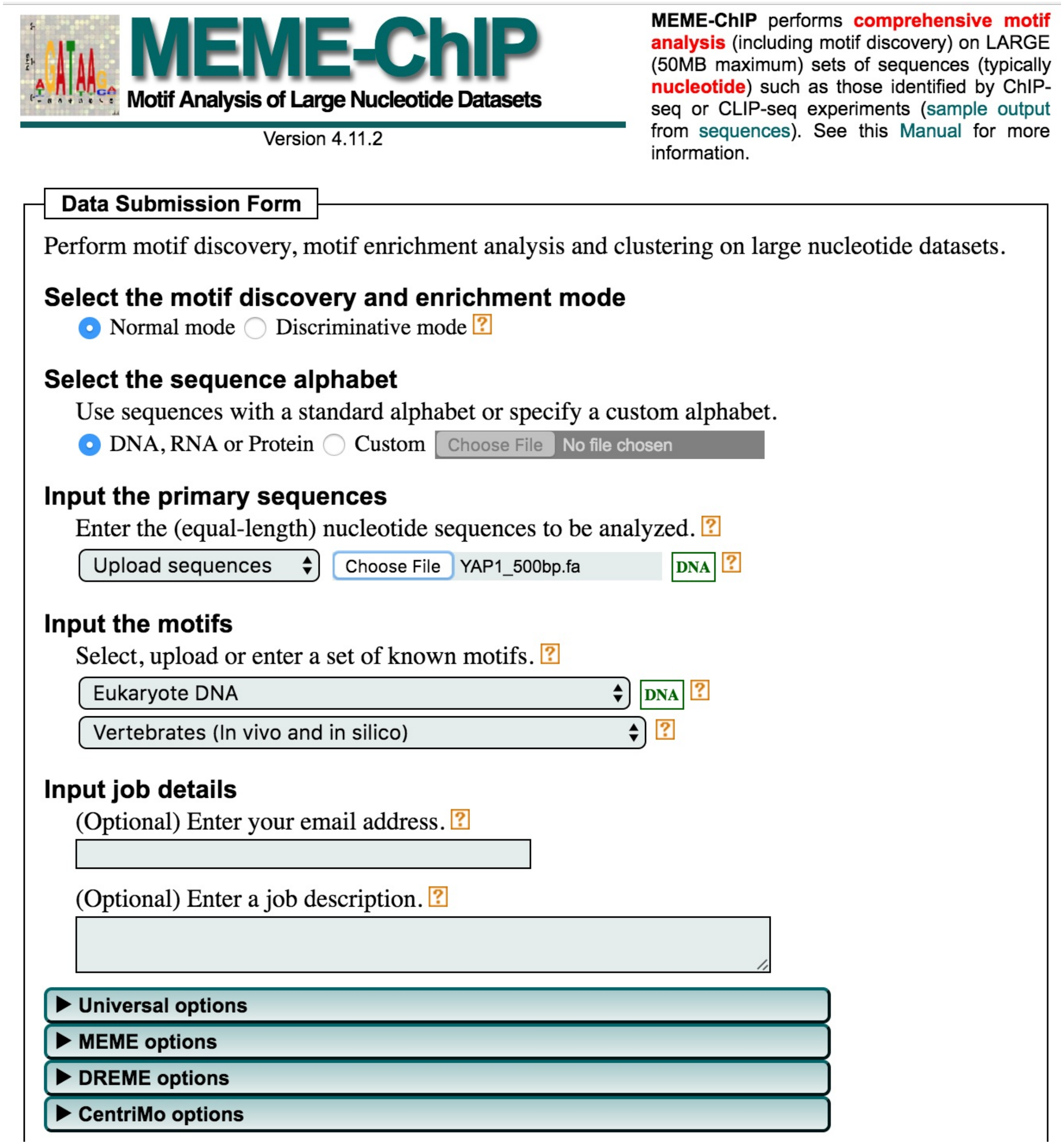
One of the most popular tools is MEME-suite for motif enrichment analysis. I will demonstrate how to use MEME-ChIP for YAP1 peaks.

`MEME-ChIP` needs 500bp DNA sequences around the YAP1 binding summits (with the highest signal of binding), which is output by `macs14` when you call peaks.

Fetching the 500 bp DNA sequence around the YAP1 summits:

```
# get the coordinates of 500bp centered on the summit of the YAP1 peaks
cat YAP1_summits.bed | awk '$2=$2-249, $3=$3+250' OFS="\t" > YAP1_500bp_summits.bed

# you need a fasta file of the whole genome, and a bed file containing the cooridnates
# the whole genome fasta file can be gotten by:
rsync -avzP rsync://hgdownload.cse.ucsc.edu/goldenPath/hg19/chromosomes/ .
cat *fa.gz > UCSC_hg19_genome.fa.gz
gunzip UCSC_hg19_genome.fa.gz

# Use betools get fasta http://bedtools.readthedocs.org/en/latest/content/tools/getfasta.html
bedtools getfasta -fi UCSC_hg19_genome.fa -bed YAP1_500bp_summits.bed -fo YAP1_500bp.fa
```

Now, upload it to `MEME-ChIP` :



MEME-ChIP
Motif Analysis of Large Nucleotide Datasets
Version 4.11.2

MEME-ChIP performs **comprehensive motif analysis** (including motif discovery) on LARGE (50MB maximum) sets of sequences (typically **nucleotide**) such as those identified by ChIP-seq or CLIP-seq experiments (sample output from sequences). See this Manual for more information.

Data Submission Form

Perform motif discovery, motif enrichment analysis and clustering on large nucleotide datasets.

Select the motif discovery and enrichment mode

[✓ Normal mode] Discriminative mode

Select the sequence alphabet

Use sequences with a standard alphabet or specify a custom alphabet.

[✓ DNA, RNA or Protein] Custom — Choose File — No file chosen

Input the primary sequences

Enter the (equal-length) nucleotide sequences to be analyzed.

Upload sequences — Choose File — YAP1_500bp.fa — DNA

Input the motifs

Select, upload or enter a set of known motifs.

Eukaryote DNA — DNA

Vertebrates (In vivo and in silico)

Input job details

(Optional) Enter your email address.

(Optional) Enter a job description.

▶ Universal options

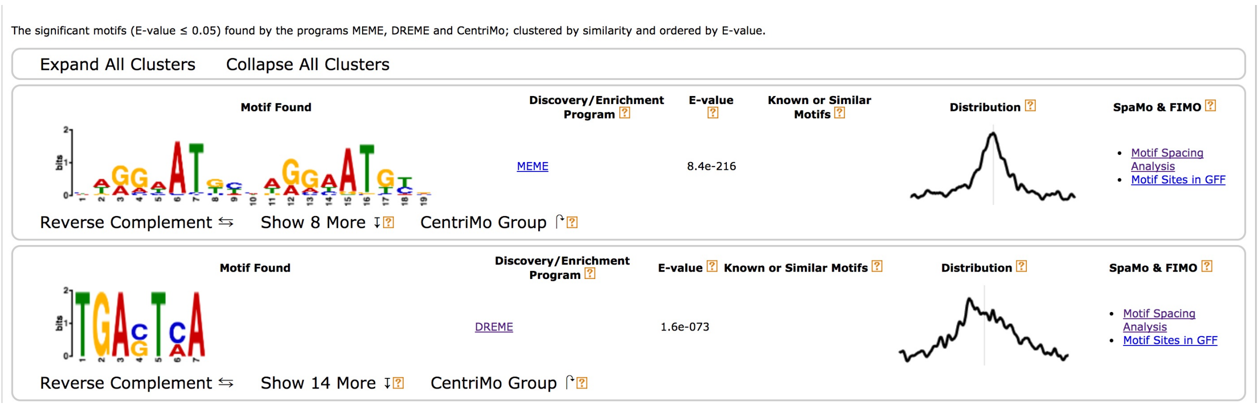
▶ MEME options

▶ DREME options

▶ CentriMo options

I will just keep the defaults for all the options. It takes a while to finish.

Output from `MEME-ChIP` :



The significant motifs (E-value ≤ 0.05) found by the programs MEME, DREME and CentriMo; clustered by similarity and ordered by E-value.

Expand All Clusters   Collapse All Clusters

Motif Found	Discovery/Enrichment Program	E-value	Known or Similar Motifs	Distribution	SpaMo & FIMO
	MEME	8.4e-216			Motif Spacing Analysis; Motif Sites in GFF

Reverse Complement ⇆   Show 8 More ↧   CentriMo Group ↱

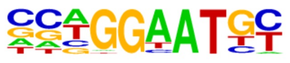
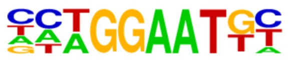
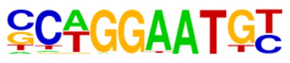
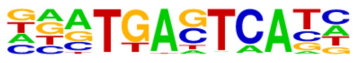
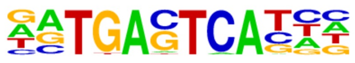
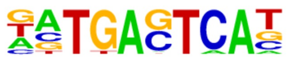
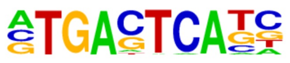
Motif Found	Discovery/Enrichment Program	E-value	Known or Similar Motifs	Distribution	SpaMo & FIMO
	DREME	1.6e-073			Motif Spacing Analysis; Motif Sites in GFF

Reverse Complement ⇆   Show 14 More ↧   CentriMo Group ↱

YAP1 is not a DNA binding transcription factor itself, rather it is in a complex with TEAD, which is a DNA binding transcription factor. The first motif is a TEAD binding motif in tandem, and the second motif is AP-1 binding motif.

One downside of `MEME-ChIP` is that you can only input the DNA sequences with the same length. Instead, you can use Homer `findMotifsGenome.pl` with the length of the peaks for finding motifs:

```
findMotifsGenome.pl YAP1_filtered_peaks.bed hg19 YAP1_motifs -size given
```

Rank	Motif	Name	P-value	log P-pvalue	q-value (Benjamini)	# Target Sequences with Motif	% of Targets Sequences with Motif
1		TEAD4(TEA)/Tropoblast-Tead4-ChIP-Seq(GSE37350)/Homer	1e-401	-9.255e+02	0.0000	1430.0	66.54%
2		TEAD(TEA)/Fibroblast-PU.1-ChIP-Seq(Unpublished)/Homer	1e-350	-8.066e+02	0.0000	1242.0	57.79%
3		TEAD2(TEA)/Py2T-Tead2-ChIP-Seq(GSE55709)/Homer	1e-348	-8.035e+02	0.0000	1116.0	51.93%
4		Fra1(bZIP)/BT549-Fra1-ChIP-Seq(GSE46166)/Homer	1e-297	-6.841e+02	0.0000	957.0	44.53%
5		Atf3(bZIP)/GBM-ATF3-ChIP-Seq(GSE33912)/Homer	1e-287	-6.631e+02	0.0000	1037.0	48.26%
6		BATF(bZIP)/Th17-BATF-ChIP-Seq(GSE39756)/Homer	1e-284	-6.552e+02	0.0000	1022.0	47.56%
7		AP-1(bZIP)/ThioMac-PU.1-ChIP-Seq(GSE21512)/Homer	1e-269	-6.212e+02	0.0000	1083.0	50.40%

The `Homer` results are in consistence with the `MEME-ChIP` results.

How to call differential peaks?

One of the other frequent questions biologists ask is how the peaks change in different conditions. e.g. different treatment of the cells, different subtypes of the same cancer. To answer this question, you'd better have biological replicates of each condition.

There is a review paper on this topic: A comprehensive comparison of tools for differential ChIP-seq analysis. You can choose tools based on various rules:

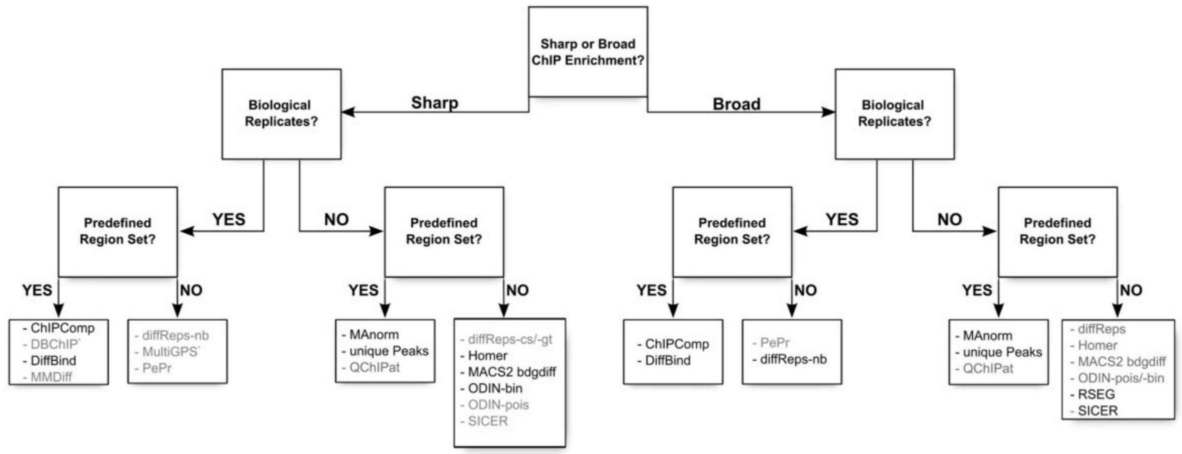


Figure 7. Decision tree indicating the proper choice of tool depending on the data set: shape of the signal (sharp peaks or broad enrichments), presence of replicates and presence of an external set of regions of interest. We have indicated in dark the name of the tools that give good results using default settings, and in gray the tools that would require parameter tuning to achieve optimal results: some tools suffer from an excessive number of DR (PePr, ODIN-pois), an insufficient number of DR (QChIPat, MMDiff, DBChIP) or from an imprecise definition of the DR for sharp signal (SICER, diffReps-nb). *MultiGPS has been explicitly developed for transcription factor ChIP-seq.

ChIP-Seq Downstream Analysis 2

The author of this guide is Ming Tang. The material was developed for the Biostar Handbook.

- GitHub profile
- Diving into Genetics and Genomics

How do I generate a heatmap with ChIP-seq data?

Heatmap is of no mystery. It is a way to visualize the data a.k.a. using colors to represent values. However, one really needs to understand the details of heatmaps. I recommend you to read Mapping quantitative data to color and Heat maps from a series of articles from *Nature Methods*.

Usually one has a matrix and then plot the matrix using R functions such as `heatmap.2` , `pheatmap` or `Heatmap` . With those R packages, it is very easy to make heatmaps, but you do need to read the documentations of the packages carefully to understand the details of the arguments. Read a tale of two heatmap functions.

For ChIP-seq data, one wants to plot ChIP-seq signal around the genomic features(promoters, CpG islands, enhancers, and gene bodies). To do this, you need to first take the regions of say 5kb upstream and downstream of all (around 30000) the transcription start sites (TSS), and then bin each 10kb region to 100 bins (100 bp per bin). Count the ChIP-seq signal (reads number) in each bin. Now, you have a data matrix of 30000 (promoters) x 100 (bin), and you can plot the matrix using a heatmap function mentioned above.

you can of course do it from scratch, but one of the taboos of bioinformatics is re-inventing the wheel. Most likely, someone has written a package for this type of analysis. Indeed, check this biostar post for all the available tools. Choose the right tool for yourself, if you are not that familiar with R, you can use some GUI tools. `EaSeq` is a pretty powerful tool for windows users.

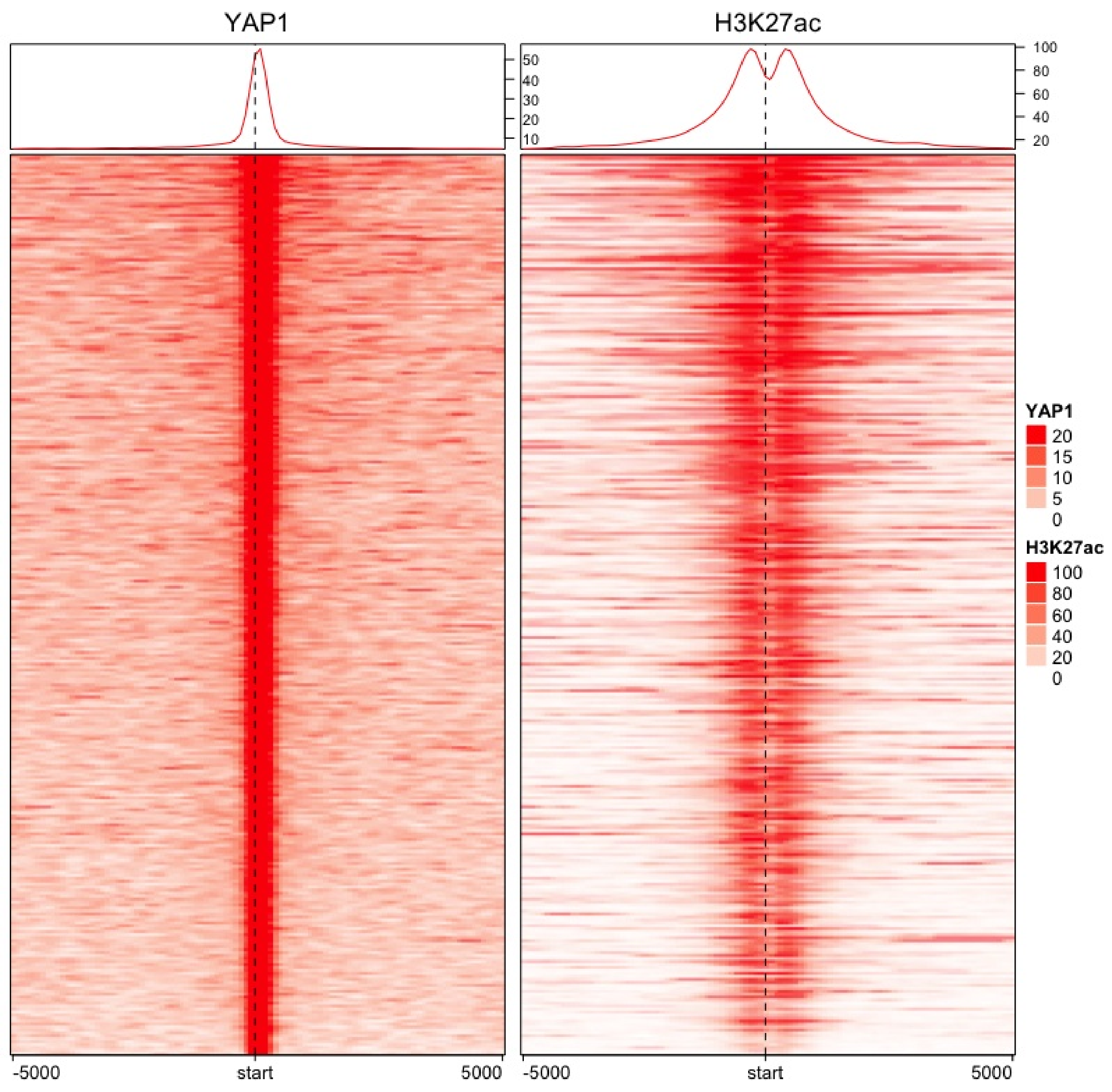
I personally like `EnrichmentHeatmap` by Zuguang Gu the most because it gives me the most flexiability to control the looks of my heatmap. It is based on `ComplexHeatmap` , which I highly recommend you to learn how to use. Below, I will walk you through how to make a heatmap with the `EnrichmentHeatmap` package.

First, let's read in the data:

```
library(EnrichedHeatmap) # for making heatmap
library(rtracklayer)  # for reading bigwig and bed files
library(GenomicRanges)
# read in the bed peak files to GRanges object
H3K27ac.bed<- import("~/ChIP-seq/results/peaks/H3K27ac_peaks.bed", format = "BED")
YAP1.bed<- import("~/ChIP-seq/results/peaks/YAP1_peaks.bed", format = "BED")


# read in bigwig files to GRanges object, it can be slow. bigwig is several hundred MB
# you can use which argument to restrict the data in certain regions.
H3K27ac.bw<- import("~/ChIP-seq/results/bams/H3K27ac.bw", format = "BigWig")
YAP1.bw<- import("~/ChIP-seq/results/YAP1.bw", format = "BigWig")
```

We want to plot the H3K27ac signal and YAP1 signal 5kb flanking the center of YAP1 peaks.

```
YAP1.10kb<- resize(YAP1.bed, width = 10000, fix = "center")

# take only the center of the peaks
YAP1.10kb.center<- resize(YAP1.10kb, width =1, fix = "center")

YAP1.mat<- normalizeToMatrix(YAP1.bw, YAP1.10kb.center, value_column = "score",
                                  mean_mode="w0", w=100, extend = 5000)

H3K27ac.mat<- normalizeToMatrix(H3K27ac.bw, YAP1.10kb.center, value_column = "score",
                                  mean_mode="w0", w=100, extend = 5000)

## check data range
quantile(YAP1.mat, probs = c(0.005, 0.5,0.90))
quantile(H3K27ac.mat, probs = c(0.005, 0.5,0.90))

## mapping colors
library(circlize)

## from the quantile, I choose the color mapping range
col_fun_YAP<- circlize::colorRamp2(c(0, 20), c("white", "red"))
col_fun_H3K27ac<- circlize::colorRamp2(c(0, 100), c("white", "red"))

EnrichedHeatmap(YAP1.mat, axis_name_rot = 0, name = "YAP1",
                column_title = "YAP1", use_raster = TRUE, col = col_fun_YAP,
                top_annotation = HeatmapAnnotation(lines = anno_enriched()),
                top_annotation_height = unit(2, "cm")) +
EnrichedHeatmap(H3K27ac.mat, axis_name_rot = 0, name = "H3K27ac",
                column_title = "H3K27ac", use_raster = TRUE, col = col_fun_H3K27ac,
                top_annotation = HeatmapAnnotation(lines = anno_enriched()),
                top_annotation_height = unit(2, "cm"))
```

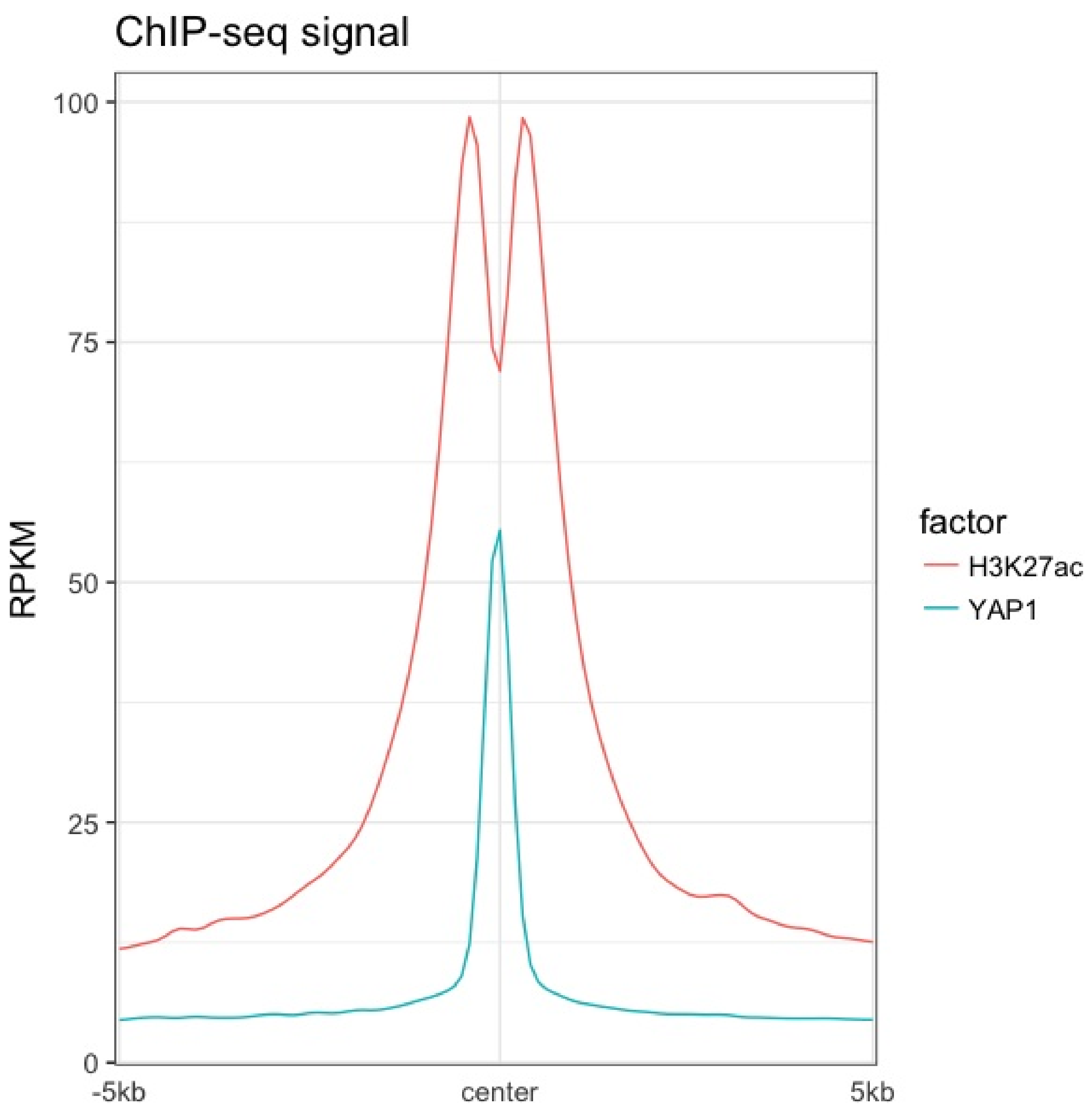


How do I generate a meta profile plot with ChIP-seq data?

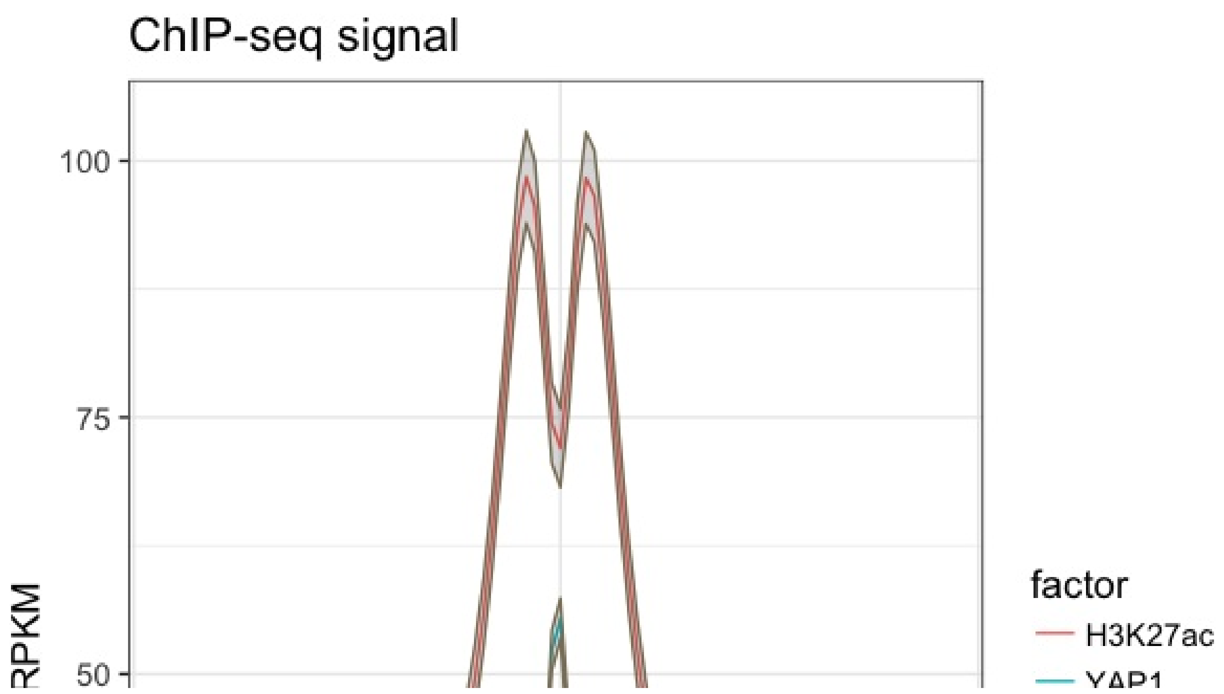
Although we used `HeatmapAnnotation(lines = anno_enriched())` to add a line plot on top of the heatmap, it may not easy to put the two line graph together. It is very common to compare signal strength for the same ChIPed factor in different conditions (treatment vs control, tumor vs normal etc). How can we do it?

The `YAP1_mat` and `H3K27ac_mat` are just like regular matrix, you can use `colMeans` to get the average and plot the average in the same figure with `ggplot2`. We can even add 95% confidence intervals onto the graph.

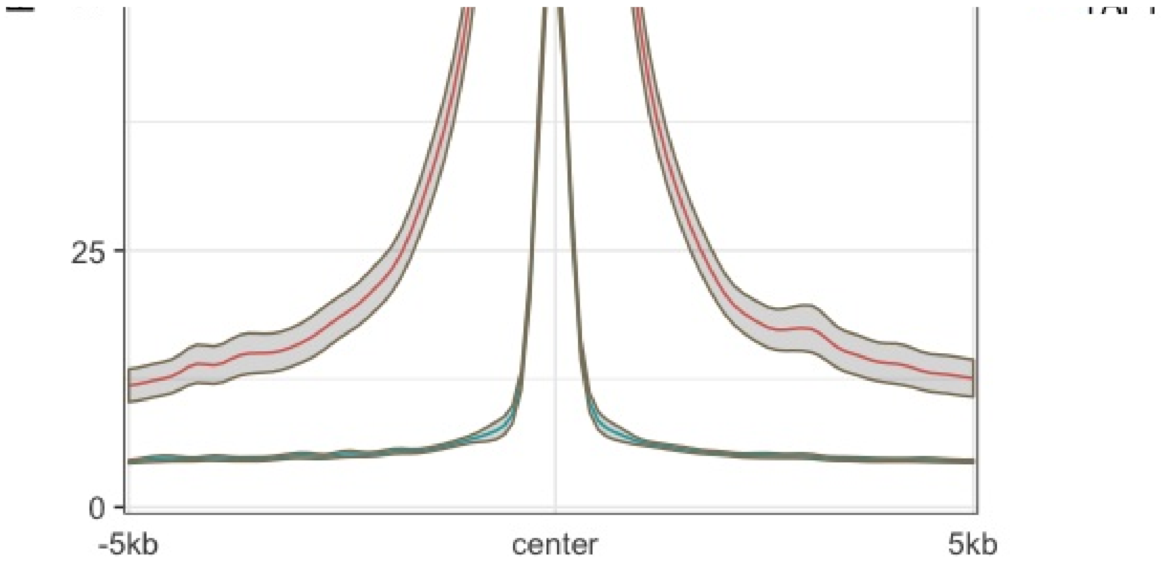
```r
YAP1_mean<- data.frame(avg = colMeans(YAP1.mat),
                          CI_lower = apply(YAP1.mat, 2, Hmisc::smean.cl.normal)[2,],
                          CI_upper = apply(YAP1.mat, 2, Hmisc::smean.cl.normal)[3,]) %>%
        mutate(factor = "YAP1", pos = colnames(YAP1.mat))

H3K27ac_mean<- data.frame(avg = colMeans(H3K27ac.mat),
                          CI_lower = apply(H3K27ac.mat, 2, Hmisc::smean.cl.normal)[2,],
                          CI_upper = apply(H3K27ac.mat, 2, Hmisc::smean.cl.normal)[3,]) %>%
        mutate(factor = "H3K27ac", pos = colnames(H3K27ac.mat))

library(tidyverse)
combine_both<- bind_rows(YAP1_mean, H3K27ac_mean)

## change position to factor and order it
combine_both$pos<- factor(combine_both$pos, levels= YAP1_mean$pos)

## without confidence interval

ggplot(combine_both, aes(x = pos,y = avg, group = factor)) + geom_line(aes(color = factor)) +
        theme_bw(base_size = 14) +
        theme(axis.ticks.x = element_blank()) +
        scale_x_discrete(breaks = c("u1", "d1", "d50"), labels =c ("-5kb", "center", "5kb")) +
        xlab(NULL) +
        ylab("RPKM")+
        ggtitle("ChIP-seq signal")

## take some touch up to finalize the figure
ggplot(combine_both, aes(x = pos,y = avg, group = factor)) + geom_line(aes(color = factor)) +
        geom_ribbon(aes(ymin= CI_lower, ymax=CI_upper), alpha=0.2, col = "#8B7E66") +
        theme_bw(base_size = 14) +
        theme(axis.ticks.x = element_blank()) +
        scale_x_discrete(breaks = c("u1", "d1", "d50"), labels =c ("-5kb", "center", "5kb")) +
        xlab(NULL) +
        ylab("RPKM")+
        ggtitle("ChIP-seq signal")
```



ChIP-seq signal
RPKM
100
75
50
25
0
-5kb
center
5kb
factor
H3K27ac
YAP1



ChIP-seq signal
RPKM
100
75
50
factor
H3K27ac
YAP1



As you can see, when you really know the power of `R` programming language, you do much more!

Where can I get public available ChIP-seq data sets?

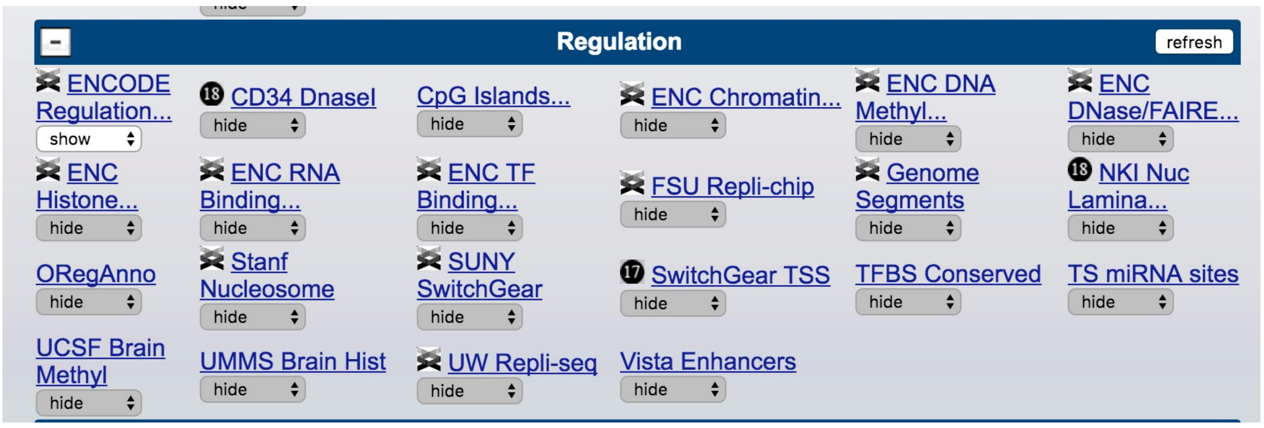
ENCODE

It is not complete for a ChIP-seq chapter without mentioning the ENCODE project. Many ChIP-seq data sets from various cell types can be found there. In addtion, ENCODE has many other data sets such as ChIA-PET, Hi-C and DNase-seq.

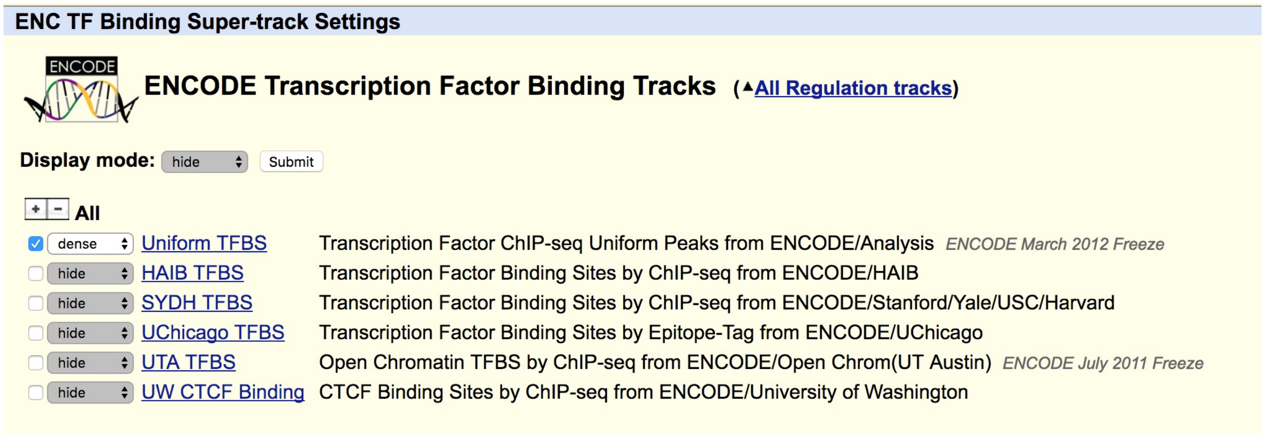
UCSC genome browser

Every biologist need to know how to use UCSC genome browser! It is one of the most popular web genome browsers.

If you scroll to the `Regulation` tracks (hg19 version). There are many ChIP-seq data avaiable for you to browse.



click `ENC TF Binding` :



ENC TF Binding Super-track Settings

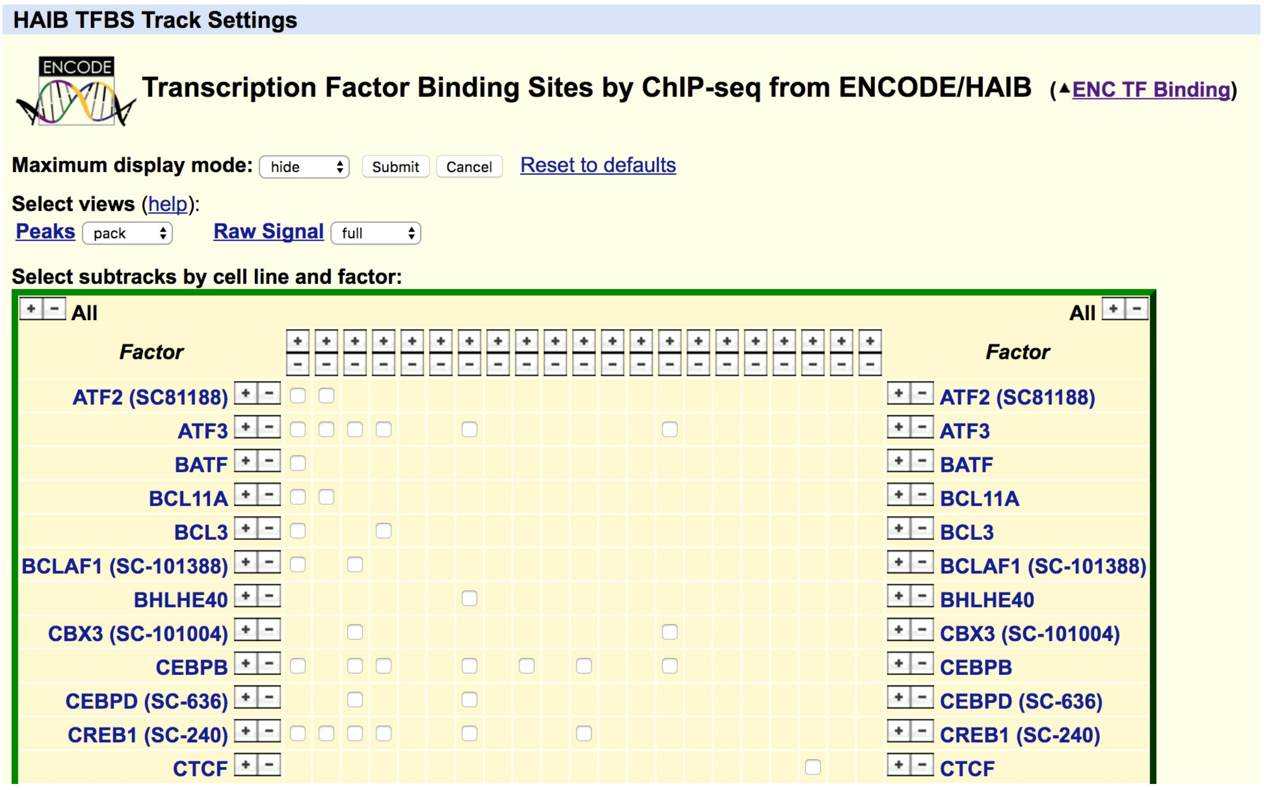
ENCODE Transcription Factor Binding Tracks (▲All Regulation tracks)

Display mode: hide Submit

All

✓	dense	Uniform TFBS	Transcription Factor ChIP-seq Uniform Peaks from ENCODE/Analysis	*ENCODE March 2012 Freeze*
☐	hide	HAIB TFBS	Transcription Factor Binding Sites by ChIP-seq from ENCODE/HAIB	
☐	hide	SYDH TFBS	Transcription Factor Binding Sites by ChIP-seq from ENCODE/Stanford/Yale/USC/Harvard	
☐	hide	UChicago TFBS	Transcription Factor Binding Sites by Epitope-Tag from ENCODE/UChicago	
☐	hide	UTA TFBS	Open Chromatin TFBS by ChIP-seq from ENCODE/Open Chrom(UT Austin)	*ENCODE July 2011 Freeze*
☐	hide	UW CTCF Binding	CTCF Binding Sites by ChIP-seq from ENCODE/University of Washington	

click `HAIB TFBS` :



HAIB TFBS Track Settings

Transcription Factor Binding Sites by ChIP-seq from ENCODE/HAIB (▲ENC TF Binding)

Maximum display mode: hide Submit Cancel Reset to defaults

Select views (help):
Peaks pack **Raw Signal** full

Select subtracks by cell line and factor:

Factor	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	Factor
ATF2 (SC81188)	☐	☐																				ATF2 (SC81188)
ATF3	☐	☐	☐	☐			☐							☐								ATF3
BATF	☐																					BATF
BCL11A	☐	☐																				BCL11A
BCL3	☐			☐																		BCL3
BCLAF1 (SC-101388)	☐		☐																			BCLAF1 (SC-101388)
BHLHE40							☐															BHLHE40
CBX3 (SC-101004)			☐											☐								CBX3 (SC-101004)
CEBPB	☐		☐	☐			☐		☐		☐			☐								CEBPB
CEBPD (SC-636)			☐				☐															CEBPD (SC-636)
CREB1 (SC-240)	☐	☐	☐	☐			☐				☐											CREB1 (SC-240)
CTCF																			☐			CTCF

Now you can check various boxes to make it show up in the browser. quite useful! Many of the data sets are from ENCODE.

I recommend you to watch the tutorials from openhelix to take full advantage of this great resource.

Cistrome

Cistrome is a project maintained by Sheirly Liu's lab in Harvard. It has a lot more data sets including ENCODE and other public data sets from GEO. Some features are very friendly to wet biologists.

GEO and ENA

Of course, if you want to adventure yourself with the the raw data in a publication, you can always go to GEO to get the SRA file and convert to fastq. Alternatively, you can directly download fastqs from European Nucleotide Archive (ENA).