Integrating Bacterial ChIP-seq and RNA-seq Data With SnakeChunks

Claire Rioualen, Lucie Charbonnier-Khamvongsa, Julio Collado-Vides, Jacques van Helden

To cite this version:

Claire Rioualen, Lucie Charbonnier-Khamvongsa, Julio Collado-Vides, Jacques van Helden. Integrating Bacterial ChIP-seq and RNA-seq Data With SnakeChunks. Current Protocols in Bioinformatics, 2019, pp.e72. 10.1002/cpbi.72 . hal-02078136

HAL Id: hal-02078136

https://amu.hal.science/hal-02078136

Submitted on 19 Jun 2019

Integrating Bacterial ChIP-seq and RNA-seq Data With SnakeChunks

Claire Rioualen,[1,2] Lucie Charbonnier-Khamvongsa,[1] Julio Collado-Vides,[2,3] and Jacques van Helden[1,4,5]

[1] Aix-Marseille University, INSERM, Laboratory of Theory and Approaches of Genome Complexity (TAGC), Marseille, France
[2] Programa de Genómica Computacional, Centro de Ciencias Genómicas, Universidad Nacional Autónoma de México, Cuernavaca, México
[3] Department of Biomedical Engineering, Boston University, Boston, Massachusetts
[4] Institut Français de Bioinformatique (IFB), UMS 3601-CNRS, Université Paris-Saclay, Orsay, France
[5] Corresponding author: *Jacques.van-Helden@univ-amu.fr*

Next-generation sequencing (NGS) is becoming a routine approach in most domains of the life sciences. To ensure reproducibility of results, there is a crucial need to improve the automation of NGS data processing and enable forthcoming studies relying on big datasets. Although user-friendly interfaces now exist, there remains a strong need for accessible solutions that allow experimental biologists to analyze and explore their results in an autonomous and flexible way. The protocols here describe a modular system that enable a user to compose and fine-tune workflows based on SnakeChunks, a library of rules for the Snakemake workflow engine (Köster and Rahmann, 2012). They are illustrated using a study combining ChIP-seq and RNA-seq to identify target genes of the global transcription factor FNR in *Escherichia coli* (Myers et al., 2013), which has the advantage that results can be compared with the most up-to-date collection of existing knowledge about transcriptional regulation in this model organism, extracted from the RegulonDB database (Gama-Castro et al., 2016). © 2019 by John Wiley & Sons, Inc.

Keywords: ChIP-seq • *Escherichia coli* K-12 • FAIR Guiding Principles • reproducible science • RNA-seq • workflow

How to cite this article:
Rioualen, C., Charbonnier-Khamvongsa, L., Collado-Vides, J., & van Helden, J. (2019). Integrating bacterial ChIP-seq and RNA-seq data with SnakeChunks. *Current Protocols in Bioinformatics*, e72.
doi: 10.1002/cpbi.72

INTRODUCTION

Next-generation sequencing (NGS) technologies enable the characterization of biological gene regulation at an unprecedented scale. Transcription-factor binding can be characterized at the genome scale by chromatin immunoprecipitation with DNA sequencing (ChIP-seq), whereas RNA sequencing (RNA-seq) makes it possible to quantify all transcripts.

The analysis of sequenced reads requires a number of successive bioinformatics processing steps, organized into workflows. A workflow, or pipeline, is defined as a chaining of commands and tools applied to a set of data files, such that the output of a given step is used as input for the subsequent one (Fig. 1). Ideally, the experimental design should from the outset take into account a perspective on the bioinformatics analyses that will enable relevant information to be extracted from the raw data. Biological samples are

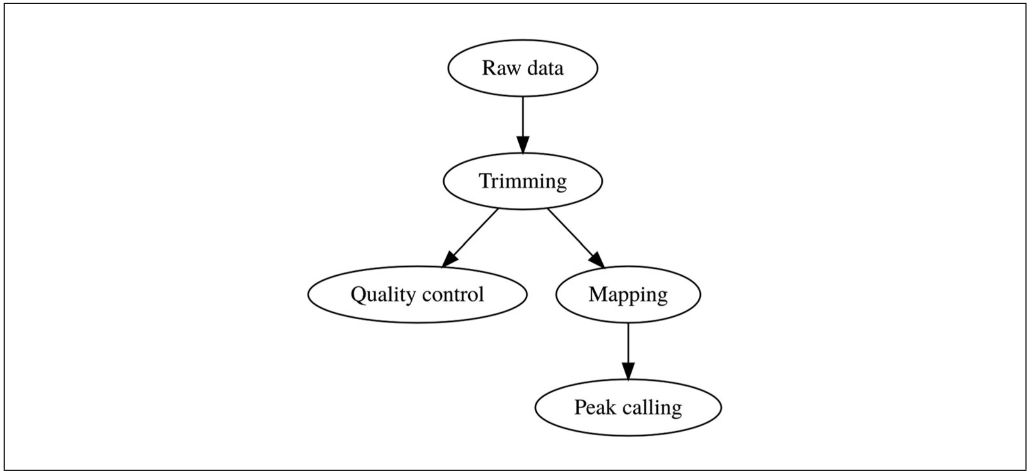


Figure 1 Schematic wiring of a basic workflow for ChIP-seq analysis.

subject to variation, and replication is thus essential to make it possible to estimate the statistical significance of the final results and to ensure an appropriate tradeoff between sensitivity and specificity. It is also necessary, as in any other biological experiment, to carefully define the control conditions that will distinguish signal from noise (see Commentary for more details).

Exploitation of the data by properly implemented bioinformatics workflows (with comprehensive specification of the tools and their versions and selection of parameters) is crucial to ensuring the traceability and reproducibility of the results from the raw data. Following a defined workflow also makes it possible to perform identical operations on dozens of samples, using powerful computing infrastructures when necessary. Snakemake (Köster & Rahmann, 2012) is a software conceived for building such workflows. Based on the Python language, it inherits concepts from GNU make (*https://www.gnu.org/software/make*): a workflow is defined by a set of rules, each defining an operation characterized by its inputs, outputs, and parameters, and a list of target files to be generated through these operations.

SnakeChunks is a library of workflows using the Snakemake framework and designed for the analysis of ChIP-seq and RNA-seq data. It includes rules for the quality control of sequencing reads, removal of adapters and trimming of low-quality bases, read mapping on a reference genome, peak calling to detect local enrichment of reads resulting from the binding of a transcription factor, gene-wise quantification of RNAs, and differential gene expression analysis (Fig. 2A).

The SnakeChunks library has been used to analyze RNA-seq data from *Mus musculus*, *Drosophila melanogaster*, *Saccharomyces cerevisiae*, and *Glossina palpalis* (Tsagmo Ngoune et al., 2017) and from *Desulfovibri desulfuricans* (Cadby et al., 2017), as well as ChIP-seq data from *Arabidopsis thaliana* (Castro-Mondragon, Rioualen, Contreras-Moreira, & van Helden, 2016). We illustrate here its use on combined RNA-seq and ChIP-seq data from *Escherichia coli* (Myers et al., 2013).

Since the initial description of the operon structure (Jacob & Monod, 1961), *E. coli* K-12 has been a model organism of reference for the study of gene regulation, resulting in thousands of publications reporting information about around 200 of the total ~300 transcription factors (TFs) identified in its genome (Blattner et al., 1997; Pérez-Rueda & Collado-Vides, 2000). Detailed information about TFs and their binding sites, binding motifs, target genes, and operons has been collected for three decades in RegulonDB, the database on the transcriptional regulation in *E. coli* (Gama-Castro et al., 2016), by manual curation of publications based on low-throughput experiments. Nonetheless, a

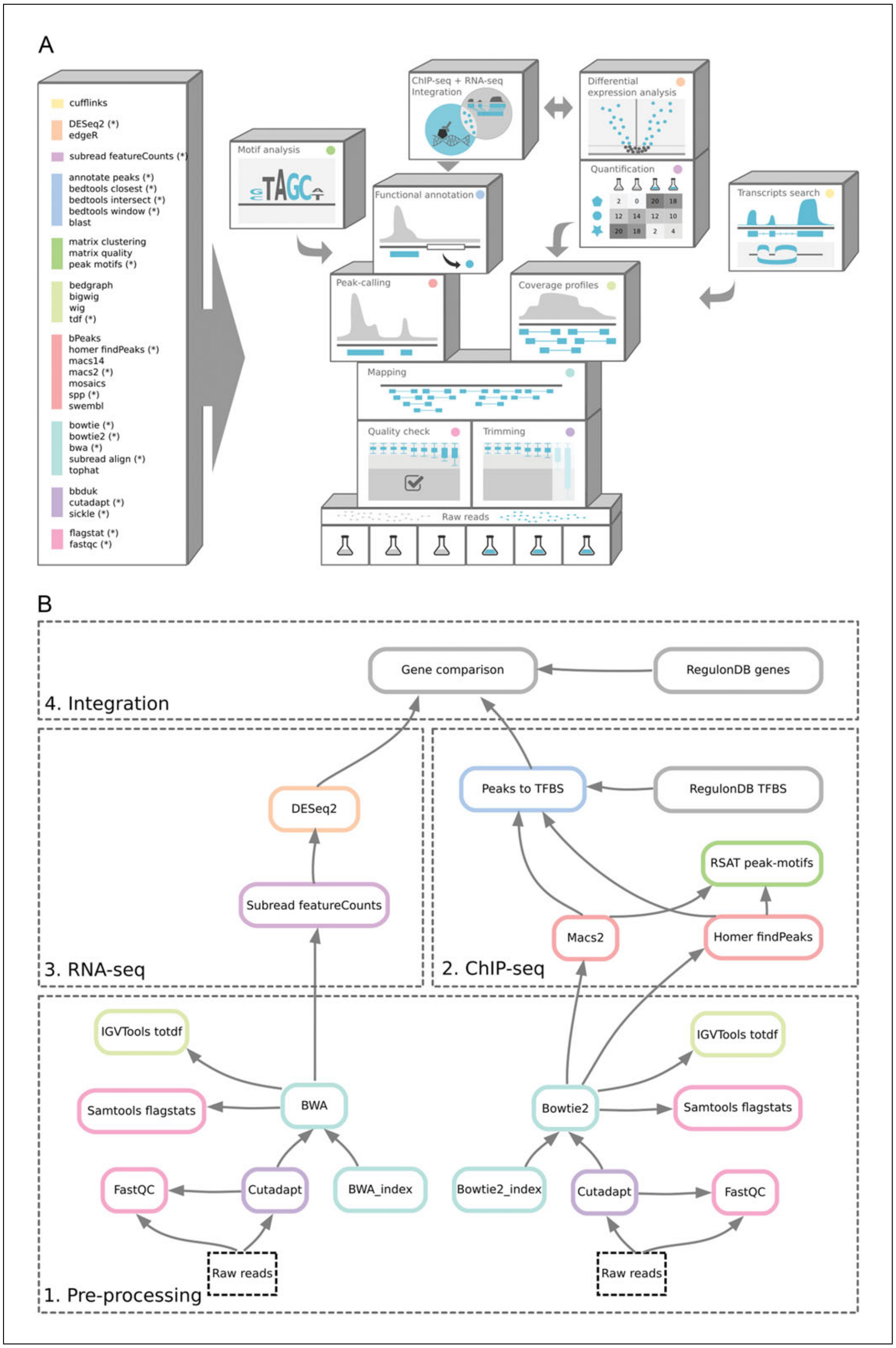


Figure 2 Organization of the SnakeChunks library. (**A**) Principle of the SnakeChunks library. The library is built around a set of Snakemake rules that can be used as building blocks to build workflows in a modular way. Each rule makes it possible to perform a given type of operation with a given tool. A given operation can also be done with alternative tools, as denoted by the color code in list of rules (left side) and on the building bricks. The rules marked with an asterisk (*) are currently supported by Conda. (**B**) Schematic flowchart of the workflows described in this unit.

good deal of information remains to be discovered to provide a global, comprehensive picture of the regulatory network of even this best-characterized model organism. NGS technologies enable the characterization of biological regulation at an unprecedented scale, and have been widely adopted by research communities. ChIP-seq gives insight into regulatory mechanisms by providing genome-wide binding locations for transcription factors, whereas RNA-seq provides information about the functional implications of regulation by measuring the level of transcription of all genes under different conditions.

ChIP-seq publications initially focused on human and metazoan models (PubMed currently returns ~1,600 ChIP-seq studies for *Homo sapiens* and more than 2,000 for *M. musculus*), and a surprisingly small number of factors were characterized by ChIP-seq in *E. coli* (44 entries in PubMed). However, systematic studies have led to the characterization of 50 transcription factors of *Mycobacterium tuberculosis* (Galagan et al., 2013), and similar projects are on the way for other bacteria, including *E. coli*. The protocols described here address the foreseeable needs of microbiologists undertaking projects based on ChIP-seq, RNA-seq, or both together to analyze bacterial regulation. Those are illustrated by a case study based on a genome-scale analysis of the FNR transcription factor (Myers et al., 2013), a DNA-binding protein that regulates a large family of genes involved in cellular respiration and carbon metabolism during anaerobic cell growth.

This unit is organized as follows.

- Strategic Planning: installation and configuration of the software environment (Conda environment, software tools, SnakeChunks library, and reference genome).
- Basic Protocol 1: preprocessing, which includes quality control, trimming, and mapping of the raw reads on the reference genome. This protocol is illustrated for the case of a ChIP-seq study but can be applied to RNA-seq data as well.
- Basic Protocol 2: analysis of ChIP-seq data: peak calling, assignation of peaks to genes, motif discovery, and comparison between ChIP-seq peaks and sites annotated in RegulonDB.
- Basic Protocol 3: analysis of RNA-seq data: preprocessing (as in Basic Protocol 1), transcript quantification (counts per gene), and detection of differentially expressed genes.
- Basic Protocol 4: integration of ChIP-seq and RNA-seq results: comparison between genes associated with the ChIP-seq peaks, differentially expressed genes reported by transcriptome analysis, and experimentally proven TF target genes annotated in RegulonDB, as well as visualization of the results using a genome browser.
- Alternate Protocol: running of the RNA-seq workflow with the user-friendly graphical interface Sequanix.
- Support Protocol: customization of the ChIP-seq workflow parameters.

The basic protocols are conceived in a modular way (Fig. 2B). In particular, ChIP-seq and RNA-seq analyses can be done separately.

NECESSARY RESOURCES

Computer Resources

This protocol runs on any Unix system (Linux, Mac OS X). Memory and CPU requirements depend on the volumes of data being handled. The study cases have been tested on Ubuntu 14.04, 16.04, and 18.04 (4 CPUs, 16 Gb RAM), on Centos 6.6, and on Mac OSX High Sierra (4 CPUs, 16 Gb RAM).

The full procedure uses ~60 Gb of disk space, including ~5 Gb for the installation of the software environment (Conda, libraries, and tools), ~15 Gb of downloaded raw reads (compressed fastq files, genome annotations), and ~40 Gb for the intermediate and final result files.

The total processing time for all tasks is ~12 h, of which 45% is spent on read mapping and 33% on trimming RNA-seq samples. This time might be further reduced by parallelizing some tasks on a multi-CPU server or cluster (on our four-core configurations, the analyses were completed in ~3 h).

Conda

Conda is an open-source package and environment management system used to automate the installation of all the software components required by the workflows. It greatly facilitates the installation of software tools from multiple sources on different Unix operating systems (Linux and Mac OS X). In addition, the installation and use of all software tools inside a custom environment ensures their isolation from the hosting system and prevents potential clashes with existing tools and libraries.

Conda should be installed prior to the execution of the protocols. It comes in two different versions, Anaconda and Miniconda. We recommend using Miniconda, which takes less disk space and makes it possible to install only the required software. Instructions can be found here: *https://conda.io/docs/user-guide/install/index.html*.

Make sure that the folder containing the Conda executable is added to your $PATH variable. This can be done automatically during the execution of the Miniconda installation script, or later by adding the following command to the bash profile (file `~/.bash_profile`).

```
export PATH=$PATH:~/miniconda3/bin/
```

You now need to log out and open a new terminal session in order for the path to be updated.

Other Software

In the protocols, we use the “tree” software to display the structure of folders and included files in the Unix terminal. This software is not technically required for the analysis, but offers a convenient way to check the proper organization of the files in the shell. Its installation can vary depending on the operating system or Linux distribution. Here are examples of tree installation with some popular package management systems.

Linux Ubuntu: `sudo apt-get install tree`
Linux CentO: `sudo yum install tree`
Mac OS X: `brew install tree`

IMPORTANT NOTE: *Throughout the following protocols, the instructions (text in Courier font) should be typed or copy-pasted in a terminal.*

STRATEGIC PLANNING

Configuration of the Conda Environment

This section provides a succession of Unix commands that enable a user to configure Conda, create a specific environment, install the required software (Snakemake and NGS tools), and download the reference genome and annotations (in our case, *E. coli* K-12 MG1655, release 37). Much of this procedure needs to be done only once, when first setting up the environment; steps 3, 5, and 7 then need to be repeated for each session (see annotation to step 10 for details).

1. Configure Conda.

   ```
   conda config --add channels r;
   conda config --add channels defaults;
   conda config --add channels conda-forge;
   conda config --add channels bioconda
   ```

IMPORTANT NOTE: These commands must be typed in the precise order indicated above, which defines the priorities for packages that exist in several channels. Conda may issue warnings, which can be ignored, when some of the channels are already present — we intentionally re-add these channels in order to place them in the right order of precedence.

2. Create an empty SnakeChunks environment using Python version 3.6.

   ```
   conda create --name snakechunks_env python=3.6
   ```

3. Activate the environment.

   This must be done for each new analysis session.

   ```
   source activate snakechunks_env
   ```

Check that the environment is active: i.e., that the Unix prompt is prepended by "`(snakechunks_env)`".

4. Install Snakemake and some required software tools in the Conda environment: GNU make software, Python panda library, and the Integrative Genomics Viewer (IGV).

   ```
   conda install make snakemake=5.1.4 igv=2.4.9 pandas=
     0.23.4
   ```

5. Define an environment variable with the directory for this analysis.

   This must be done for each new analysis session (alternatively, you can declare it in your bash profile).

   ```
   export ANALYSIS_DIR=$HOME/FNR_analysis
   ```

6. Create the analysis directory.

   ```
   mkdir -p $ANALYSIS_DIR
   ```

7. Set the current working directory to the analysis directory.

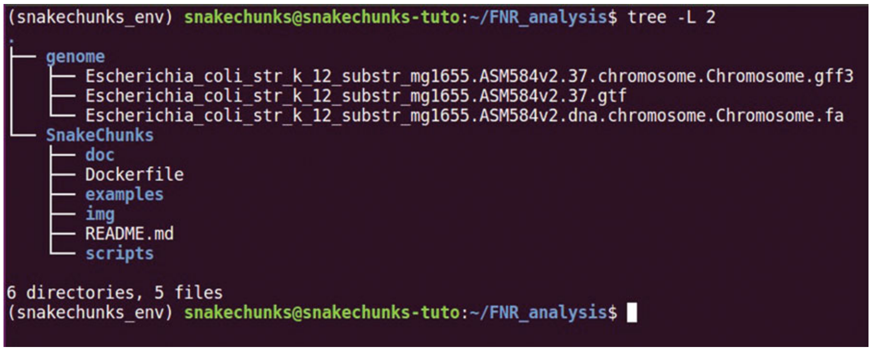
   This must be done for each new analysis session.

   ```
   cd $ANALYSIS_DIR
   ```

8. Download the SnakeChunks library from GitHub. We recommend keeping a copy of the library in the analysis directory to ensure consistency and reproducibility. The latest version of the SnakeChunks library can be downloaded easily with the following Git command.

   ```
   git clone https://github.com/SnakeChunks/SnakeChunks.
     git
   ```

IMPORTANT NOTE: The SnakeChunks code will continue evolving with time. For the sake of backward compatibility, we froze the precise version of the library used at the time of publication of this article. This version can be downloaded with the following command.



```
(snakechunks_env) snakechunks@snakechunks-tuto:~/FNR_analysis$ tree -L 2
.
├── genome
│   ├── Escherichia_coli_str_k_12_substr_mg1655.ASM584v2.37.chromosome.Chromosome.gff3
│   ├── Escherichia_coli_str_k_12_substr_mg1655.ASM584v2.37.gtf
│   └── Escherichia_coli_str_k_12_substr_mg1655.ASM584v2.dna.chromosome.Chromosome.fa
└── SnakeChunks
    ├── doc
    ├── Dockerfile
    ├── examples
    ├── img
    ├── README.md
    └── scripts

6 directories, 5 files
(snakechunks_env) snakechunks@snakechunks-tuto:~/FNR_analysis$
```

Figure 3 File organization after the Strategic Planning section is completed.

```
wget --no-clobber \
  https://github.com/SnakeChunks/SnakeChunks/archive/
  4.1.4.tar.gz
tar xvzf 4.1.4.tar.gz
mv SnakeChunks-4.1.4 SnakeChunks
```

9. Download the reference genome of *E. coli* K-12 and its annotations.

```
make -f SnakeChunks/examples/GSE41195/tutorial_
  material.mk \
download_genome_data
```

10. Check the organization of the files in the genome directory (Fig. 3).

```
tree -L 2
```

IMPORTANT NOTE: The above steps are used to set up the environment and need to be executed only once, except for steps 3, 5, and 7, which are required for each working session for this project. If you log out of the terminal and want to start a new session later, you will need to reactivate the Conda environment (step 3), redefine the environment variable for the analysis directory (step 5), and set it as the current directory (step 7).

DATA PREPROCESSING AND READ MAPPING

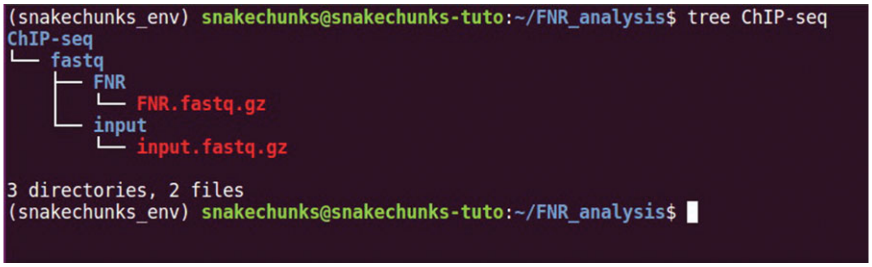
BASIC PROTOCOL 1

Data preprocessing covers the first steps of the analysis, which are common to most NGS workflows. The goal is to make sure that the raw sequencing data are suitable for a proper bioinformatics analysis. This process includes quality control of the sequenced reads, removal of the sequencing adapters, and trimming of the read extremities when needed. These operations are described more thoroughly in the Guidelines for Understanding Results below. We illustrate these steps with a ChIP-seq dataset, but they can be applied similarly to RNA-seq data.

Once the reads are processed and filtered appropriately, a common operation to perform before ChIP-seq and RNA-seq analyses is to map the reads on a reference genome in order to identify their genomic location.

This protocol covers the following steps:

- Quality control of the reads using the program FastQC (Andrews, 2010);
- Removal of the adapters and trimming of the read extremities using the utility cutadapt (Martin, 2011);
- Read mapping using the algorithm bowtie2 (Langmead & Salzberg, 2012).



```
(snakechunks_env) snakechunks@snakechunks-tuto:~/FNR_analysis$ tree ChIP-seq
ChIP-seq
└── fastq
    ├── FNR
    │   └── FNR.fastq.gz
    └── input
        └── input.fastq.gz

3 directories, 2 files
(snakechunks_env) snakechunks@snakechunks-tuto:~/FNR_analysis$ 
```

Figure 4 File organization of the ChIP-seq samples before the analyses are run.

1. Download the ChIP-seq dataset from the GEO series GSE41195 (Myers et al., 2013).

   ```
   make -f SnakeChunks/examples/GSE41195/tutorial_
     material.mk\
   download_chipseq_data
   ```

   This creates a subdirectory called "ChIP-seq" in the analysis directory defined in the Strategic Planning section above (Fig. 4), with two fastq files corresponding to the FNR-chipped and control samples, respectively.

   ```
   tree ChIP-seq
   ```

2. Create a local copy of the metadata folder.

   ```
   make -f SnakeChunks/examples/GSE41195/tutorial_
     material.mk copy_metadata;
   tree metadata
   ```

   This creates a local copy of the metadata folder, which contains files describing the samples, the analysis design, and the workflow configuration.

3. Run the workflow for quality control.

   ```
   snakemake -s SnakeChunks/scripts/snakefiles/
     workflows/quality_control.wf \
   --configfile metadata/config_ChIP-seq.yml
   --config trimming="" -p --use-conda
   ```

   The command above runs a workflow using the "snakemake" command with the following specifications.

   The wiring of the workflow is defined in the file `quality_control.wf`, specified with the option `-s`. Modifying this wiring requires some knowledge of the Snakemake language, which is outside the scope of this protocol (Snakemake tutorials can be found in the Snakemake documentation at http://snakemake.readthedocs.io/en/stable/tutorial/tutorial.html). `quality_control.wf` produces quality reports using the FastQC tool (Andrews, 2010), and running this is an essential step to assess the quality of the samples and plan the next steps of the analysis.

   The workflow invokes a series of tools, each of which can be tuned with different parameters. All of the parameters of the workflow are specified in a YAML-formatted configuration file, specified with the option `--configfile`. The YAML format is human readable and can be easily edited with a standard text editor (see Support Protocol).

   - *The option `--config` is used in order to specify that trimming will not be performed during this run. It overrules the configuration defined in the configuration file mentioned above, which is to perform trimming automatically, as will be done in step 5.*
   - *The option `-p` tells Snakemake to print out all the Unix commands that will be executed. This listing is very convenient as a means to check that each*

command is called with the appropriate parameters and to keep a trace of the full process between raw data and final results.

- *When the option `--use-conda` is used, Snakemake creates a separate virtual environment for each rule executed in the workflow, and installs the required tools and their dependencies in a rule-specific subfolder. This ensures compatibility between the different tools invoked. The process can take some time at the first invocation of a given environment, but is faster for subsequent uses of the same environment.*

4. The presence of the two FastQC reports can be checked with the `ls` commands below.

   ```
   ls -l $ANALYSIS_DIR/ChIP-seq/fastq/FNR1/FNR1_fastq.
     gz_qc/FNR1_fastqc.html;
   ls -l $ANALYSIS_DIR/ChIP-seq/fastq/input1/input1_
     fastq.gz_qc/input1_fastqc.html
   ```

   These files can be opened with a Web browser. Insights about these reports can be found in the Guidelines for Understanding Results below.

5. Run the quality control workflow again using the software cutadapt, which performs both read trimming and adapter removal.

   ```
   snakemake -s SnakeChunks/scripts/snakefiles/
     workflows/quality_control.wf \
   --configfile metadata/config_ChIP-seq.yml -p
     --use-conda
   ```

   This time, the workflow will run cutadapt, as defined in the configuration file, before doing a new FastQC check. Note that SnakeChunks can be used to specify several tools for the same step, in order to compare the results. An overview of the options is proposed in Support Protocol.

6. The presence of FastQC reports can be checked with the `ls` commands below.

   ```
   ls -l \
     $ANALYSIS_DIR/ChIP-seq/fastq/FNR1/FNR1_cutadapt_
       fastq.gz_qc/FNR1_cutadapt_fastqc.html;
   ls -l \
     $ANALYSIS_DIR/ChIP-seq/fastq/input1/input1_
       cutadapt_fastq.gz_qc/input1_cutadapt_fastqc.html
   ```

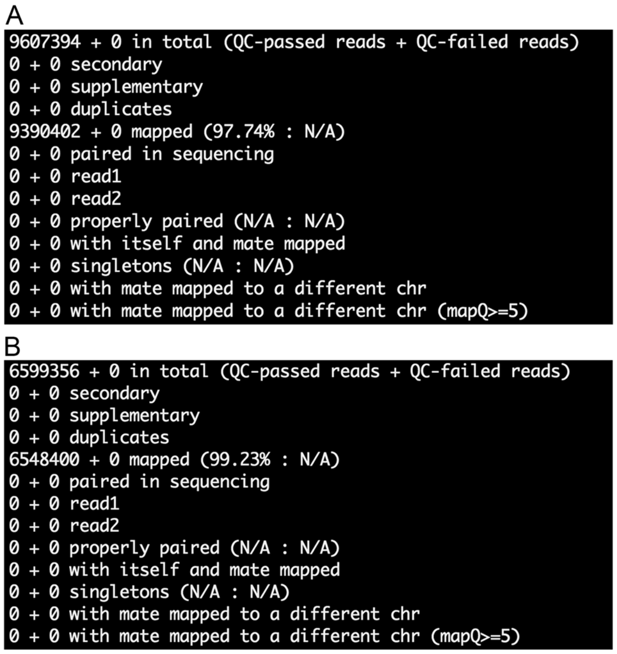
   Open the new FastQC reports with a Web browser. The reports show the improvement in the quality of the reads, as well as the absence of over-represented sequences corresponding to adapters. This is further discussed in the Guidelines for Understanding Results.

7. Run the read-mapping workflow.

   ```
   snakemake -s SnakeChunks/scripts/snakefiles/
     workflows/mapping.wf \
   --configfile metadata/config_ChIP-seq.yml -p --use-
     conda -j 2
   ```

   This workflow essentially performs two operations: read mapping and genome coverage.

   We added the option `-j 2`, which permits Snakemake to parallelize the processing with a maximum of two simultaneous jobs. Because the mapping step can be time consuming, we recommend running it in parallel for the different samples. This option should be adapted to the number of cores of your system. For example, if you analyze a large number of files on a cluster, you could increase the number of simultaneous jobs to 40 or even more (this has to be negotiated with your system administrator).



```
9607394 + 0 in total (QC-passed reads + QC-failed reads)
0 + 0 secondary
0 + 0 supplementary
0 + 0 duplicates
9390402 + 0 mapped (97.74% : N/A)
0 + 0 paired in sequencing
0 + 0 read1
0 + 0 read2
0 + 0 properly paired (N/A : N/A)
0 + 0 with itself and mate mapped
0 + 0 singletons (N/A : N/A)
0 + 0 with mate mapped to a different chr
0 + 0 with mate mapped to a different chr (mapQ>=5)
```

```
6599356 + 0 in total (QC-passed reads + QC-failed reads)
0 + 0 secondary
0 + 0 supplementary
0 + 0 duplicates
6548400 + 0 mapped (99.23% : N/A)
0 + 0 paired in sequencing
0 + 0 read1
0 + 0 read2
0 + 0 properly paired (N/A : N/A)
0 + 0 with itself and mate mapped
0 + 0 singletons (N/A : N/A)
0 + 0 with mate mapped to a different chr
0 + 0 with mate mapped to a different chr (mapQ>=5)
```

Figure 5 Read mapping statistics. Statistics were computed using the flagstats software from SAMtools for the FNR ChIP-seq sample (**A**) and genomic input (**B**), respectively.

More information about the mapping results can be found in the Guidelines for Understanding Results.

8. Check the contents of the files containing the statistics of the mapping from the shell (Fig. 5).

```
cat \
$ANALYSIS_DIR/ChIP-seq/results/samples/FNR1/FNR1_
  cutadapt_bowtie2_bam_stats.txt;
cat \
$ANALYSIS_DIR/ChIP-seq/results/samples/input1/
  input1_cutadapt_bowtie2_bam_stats.txt
```

These files, generated by the SAMtools program flagstat, display basics statistics for the mapping. As can be seen in Figure 5A and B, here both samples have a very high mapping rate, which confirms that the sequencing data are of good quality and that we are going to dispose of a large quantity of data to perform the ChIP-seq analysis.

BASIC PROTOCOL 2

ChIP-seq

ChIP-seq (Johnson, Mortazavi, Myers, & Wold, 2007; Robertson et al., 2007) is a technology that allows the characterization of DNA binding at a genome scale. The experiment includes the following steps: cross-linking DNA and the bound proteins with a fixative agent, breaking DNA into random fragments by ultrasonication, immunoprecipitating a transcription factor of interest together with its cross-linked DNA, unlinking these DNA fragments, amplifying them by PCR, and sequencing them using massively parallel sequencing technologies. The raw sequences (“reads”) are then mapped onto a reference genome, and putative binding regions—regions that contain a large number of reads, usually extending over a few hundred base pairs—are denoted as “peaks.” These peaks can then be used to search for precise transcription-factor (TF) binding sites, which can then be associated with nearby genes to infer the potential TF target genes.

Table 1 Descriptions of the ChIP-seq Samples

ID	Condition	GSM identifier	SRR identifier
FNR1	FNR	GSM1010220	SRR576934
input1	Input	GSM1010224	SRR576938

Column headers indicate their contents. the columns ID and Condition are mandatory for the proper use of the workflow. Additional columns can be added at will to document samples.

Table 2 Experimental Design of the ChIP-seq Dataset

Control	Treatment
input1	FNR1

A critical step of a ChIP-seq data analysis is peak calling, which is the detection of these genomic regions with a higher density of mapped reads than would be expected by chance. The choice of a peak-calling algorithm and the tuning of its parameters can drastically affect the number of returned peaks and their sizes. To identify reliable peaks and avoid false positives, it is important to use control samples (see Commentary for more details). Peak callers also have parameters that can be used to tune the rate of false positives by imposing more or less stringent thresholds on peak scores, in order to optimize the tradeoff between sensitivity (the proportion of actual binding regions detected) and specificity (the ability to reject non-binding regions).

Table 1 describes each sample used in the analysis: a test sample resulting from the immunoprecipitation of the FNR transcription factor, and a genomic input. Table 2 specifies the design of the analysis, by indicating the respective status of the samples (control versus treatment).

Although many publications rely on the Macs2 peak caller (Feng, Liu, & Zhang, 2011), generally used with its default parameters, there are actually a variety of tools that can be used and customized in different ways (Pepke, Wold, & Mortazavi, 2009). SnakeChunks currently supports seven of these in a completely interchangeable way (Fig. 2A). We will demonstrate two, Homer (Heinz et al., 2010) and Macs2, which are among the most widely used, maintained, and up-to-date programs for this purpose and which are also supported by Conda.

The main operations performed by the workflow described are the following:

- Peak calling using Homer and Macs2 (Feng et al., 2011; Heinz et al., 2010);
- Motif discovery by remote invocation of the tool peak-motifs (Thomas-Chollier et al., 2012) from the RSAT software suite (Nguyen et al., 2018) via its Web services interface; RSAT peak-motifs also compares discovered motifs with the TF-binding motifs annotated in RegulonDB;
- Comparison between ChIP-seq peaks and known TF binding sites listed in the RegulonDB database (Gama-Castro et al., 2016);
- Assignment of genes to peaks with the tool “annotate peaks” from the Homer suite;
- Gene comparison: comparison between genes associated with peaks and TF target genes (as annotated in RegulonDB).

1. Run the ChIP-seq workflow.

```
snakemake \
-s SnakeChunks/scripts/snakefiles/workflows/
  ChIP-seq_RegulonDB.wf \
--configfile metadata/config_ChIP-seq.yml -p --use-
  conda -j 2
```

2. The output files can be found here.

   a. Peaks: Because these files are quite large, we use the Unix command `less` to display them page by page (press enter to move one page forward). After inspecting a few pages, type "`q`" to quit the less program.

   ```
   less \
   $ANALYSIS_DIR/ChIP-seq/results/peaks/FNR1_vs_
     input1/homer/FNR1_vs_input1_cutadapt_bowtie2_
     homer.bed;
   less \
   $ANALYSIS_DIR/ChIP-seq/results/peaks/FNR1_vs_
     input1/macs2/FNR1_vs_input1_cutadapt_bowtie2_
     macs2.bed
   ```

   b. Motifs discovered with RSAT in the peaks: Check that the html files produced by peak-motifs are at the expected place.

   ```
   ls -l \
   $ANALYSIS_DIR/ChIP-seq/results/peaks/FNR1_vs_input1/
     homer/peak-motifs/FNR1_vs_input1_cutadapt_bowtie2_
     homer_peak-motifs/peak-motifs_synthesis.html;
   ls -l \
   $ANALYSIS_DIR/ChIP-seq/results/peaks/FNR1_vs_input1/
     macs2/peak-motifs/FNR1_vs_input1_cutadapt_bowtie2_
     macs2_peak-motifs/peak-motifs_synthesis.html
   ```

   Open the peak-motifs reports with a Web browser. The results of this workflow are further described in the Guidelines for Understanding Results below.

BASIC PROTOCOL 3

RNA-seq

RNA-seq technology, or whole-transcriptome shotgun sequencing, reveals the presence or absence of RNAs from a given sample, at a given moment in time, and also quantifies them if needed. It consists of extracting the total RNA from a cell and filtering out genomic DNA using a deoxyribonuclease (DNase). The RNA is then reverse transcribed to cDNA, which can either be mapped onto a genome of reference or assembled *de novo*. Subsequent analysis options include quantification of gene expression, identification of alternative transcripts, and discovery of single-nucleotide variation.

In this protocol, we will use as a case study an RNA-seq experiment published by Myers et al. (2013), in which the transcriptome of *E. coli* K-12 was measured in two samples from the wild type (WT) and from a mutant strain whose FNR transcription factor activity is inhibited (Lazazzera, Bates, & Kiley, 1993). To perform reliable RNA-seq analyses, it is crucial to dispose of biological replicates (see Commentary). This dataset includes two replicates per genotype (Table 3). Our goal will be to identify genes that are differentially expressed between the FNR mutant (defined as the test condition in Table 4) and the WT (reference condition).

This workflow accomplishes the following steps:

- Quality control and trimming of the reads (for further detail, see Basic Protocol 1);
- Mapping onto a genome of reference using the algorithm BWA (Li & Durbin, 2009) (for further detail, see Basic Protocol 1);

Table 3 Descriptions of the RNA-seq Samples

ID	Condition	GSM identifier	SRR identifier
WT1	WT	GSM1010244	SRR5344681
WT2	WT	GSM1010245	SRR5344682
dFNR1	FNR	GSM1010246	SRR5344683
dFNR2	FNR	GSM1010247	SRR5344684

Column headers indicate their contents. The columns ID and Condition are mandatory for the proper use of the workflow. Additional columns can be added at will to document samples

Table 4 Experimental Design of the RNA-seq Analysis

Test	Reference
FNR	WT

The design file can contain one or several rows, each describing a pair of conditions to be compared. The test and reference conditions must correspond to the values in the Condition column of the sample description table.

- Quantification of transcripts per gene with featureCounts from the Subread package (Liao et al., 2014);
- Detection of differentially expressed genes with DESeq2 (Love, Huber, & Anders, 2014) and edgeR (Robinson, McCarthy, & Smyth, 2010);
- Automatic generation of a report summarizing the results.

1. Copy the example metadata from the SnakeChunks library (can be skipped if already done in Basic Protocol 1, step 2), and check the content of the metadata folder.

```
make -f SnakeChunks/examples/GSE41195/tutorial_
  material.mk copy_metadata; tree metadata
```

2. Download RNA-seq data.

```
make -f SnakeChunks/examples/GSE41195/tutorial_
  material.mk download_rnaseq_data
```

   This creates a subdirectory "RNA-seq" in the analysis directory defined in Strategic Planning (Fig. 6), and downloads the raw data. Beware: during our tests, the download takes approximately 8 min per sample. Since the analysis requires eight files, this download can take up to a few hours depending on your connection speed. After the command has been completed, check the organization of the downloaded files.

```
tree -C RNA-seq
```

   You should now see four directories (one per sample), each containing two files with the extension .fastq.gz (there is one file per sequencing end).

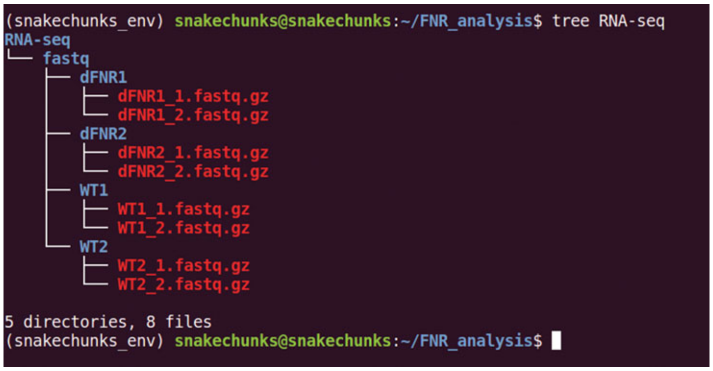
3. Run the RNA-seq analysis workflow.

```
snakemake -s SnakeChunks/scripts/snakefiles/
  workflows/RNA-seq_complete.wf \
--configfile metadata/config_RNA-seq.yml -p
  --use-conda -j 4
```

   Here we use the option `-j 4` *in order to parallelize the treatment of the four samples, which is time consuming.*

4. Check the organization of the result files in the RNA-seq folder, with a folder depth limit of 3.

```
tree -C -L 3 RNA-seq
```



```
(snakechunks_env) snakechunks@snakechunks:~/FNR_analysis$ tree RNA-seq
RNA-seq
└── fastq
    ├── dFNR1
    │   ├── dFNR1_1.fastq.gz
    │   └── dFNR1_2.fastq.gz
    ├── dFNR2
    │   ├── dFNR2_1.fastq.gz
    │   └── dFNR2_2.fastq.gz
    ├── WT1
    │   ├── WT1_1.fastq.gz
    │   └── WT1_2.fastq.gz
    └── WT2
        ├── WT2_1.fastq.gz
        └── WT2_2.fastq.gz

5 directories, 8 files
(snakechunks_env) snakechunks@snakechunks:~/FNR_analysis$
```

Figure 6 File organization of the RNA-seq samples before the analyses are run.

5. The results of the differential expression analysis performed by this workflow are summarized in an automatically generated HTML report, which can be opened using a web navigator.

```
RNA-seq/results/diffexpr/cutadapt_bwa_featureCounts_
  rna-seq_deg_report.html
```

The elements of this report are further described in the Guidelines for Understanding Results below.

6. Optionally, it is now possible to check the content of the main result files, which can be found here.

```
ls -l RNA-seq/results/diffexpr
```

This folder contains a table with the counts of reads per gene:

```
less RNA-seq/results/diffexpr/cutadapt_bwa_
  featureCounts_all.tsv
```

and a subfolder with the differential analysis results produced by edgeR, DESeq2, and the two together.

```
ls -l RNA-seq/results/diffexpr/FNR_vs_WT
```

It also contains two tables with the differential analysis statistics returned by DESeq2 and edgeR, respectively.

```
less \
RNA-seq/results/diffexpr/FNR_vs_WT/cutadapt_bwa_
  featureCounts_FNR_vs_WT_DESeq2.tsv;
less \
RNA-seq/results/diffexpr/FNR_vs_WT/cutadapt_bwa_
  featureCounts_FNR_vs_WT_edgeR_TMM.tsv
```

The subset of differentially expressed genes (those declared positive because they pass the significance threshold) are exported in an additional file.

```
less \
RNA-seq/results/diffexpr/FNR_vs_WT/cutadapt_bwa_
  featureCounts_FNR_vs_WT_DEG_table.tsv
```

In the tutorial, we retain the union of genes called positive by either DESeq2 or edgeR, but alternatively, the combination rule can be tuned in the YAML configuration file.

7. We can count the rows of this file to get an idea of the number of differentially expressed genes (after subtracting one for the header line).

```
wc -l
RNA-seq/results/diffexpr/FNR_vs_WT/cutadapt_bwa_
  featureCounts_FNR_vs_WT_DEG_table.tsv\
| awk '{print $1 -1}'
```

BASIC PROTOCOL 4

INTEGRATION

We have seen in Basic Protocol 2 that a ChIP-seq experiment followed by peak calling can be used to identify genomic binding locations for a given transcription factor. In Basic Protocol 3, we analyzed results of an RNA-seq experiment to identify genes differentially expressed between two conditions (wild-type versus FNR mutant).

Here, we show how to combine the results of those two types of experiments in order to unravel the links between genome binding data (ChIP-seq) and differential expression data (RNA-seq). This allows to detect not only direct target genes of a factor, i.e., genes whose transcription level is affected in the mutant, and whose upstream region contains a binding peak, but also indirect regulation (absence of a binding peak but presence of an observed effect on the expression of a gene) or binding of the FNR transcription factor without detected effect on the level of transcription of the associated genes. We also compare the NGS results with the list of FNR target genes annotated in the RegulonDB database (Gama-Castro et al., 2016).

1. Run integration workflow.

```
snakemake -p \
-s SnakeChunks/scripts/snakefiles/workflows/
  integration_ChIP_RNA.wf \
--configfile metadata/config_integration.yml
  --use-conda
```

2. Check the first lines of the table summarizing the results for each gene.

```
less $ANALYSIS_DIR/integration/ChIP-RNA-regulons_
  homer_gene_table.tsv
```

   For a better readability, we recommend opening this table with spreadsheet software (e.g., Office Calc or Excel). The table contains annotations for all genes known in E. coli K-12, as well as an indication of whether they are associated with FNR binding (ChIP-seq column), whether their transcription is affected by FNR (RNA-seq column), and whether they have been previously demonstrated to be regulated by FNR (FNR_regulon column).

3. Launch the IGV browser (Robinson et al., 2011; Thorvaldsdóttir, Robinson, & Mesirov, 2013):

   On Linux operating systems: `igv`

   In Mac OS X: open the IGV in the Applications folder.

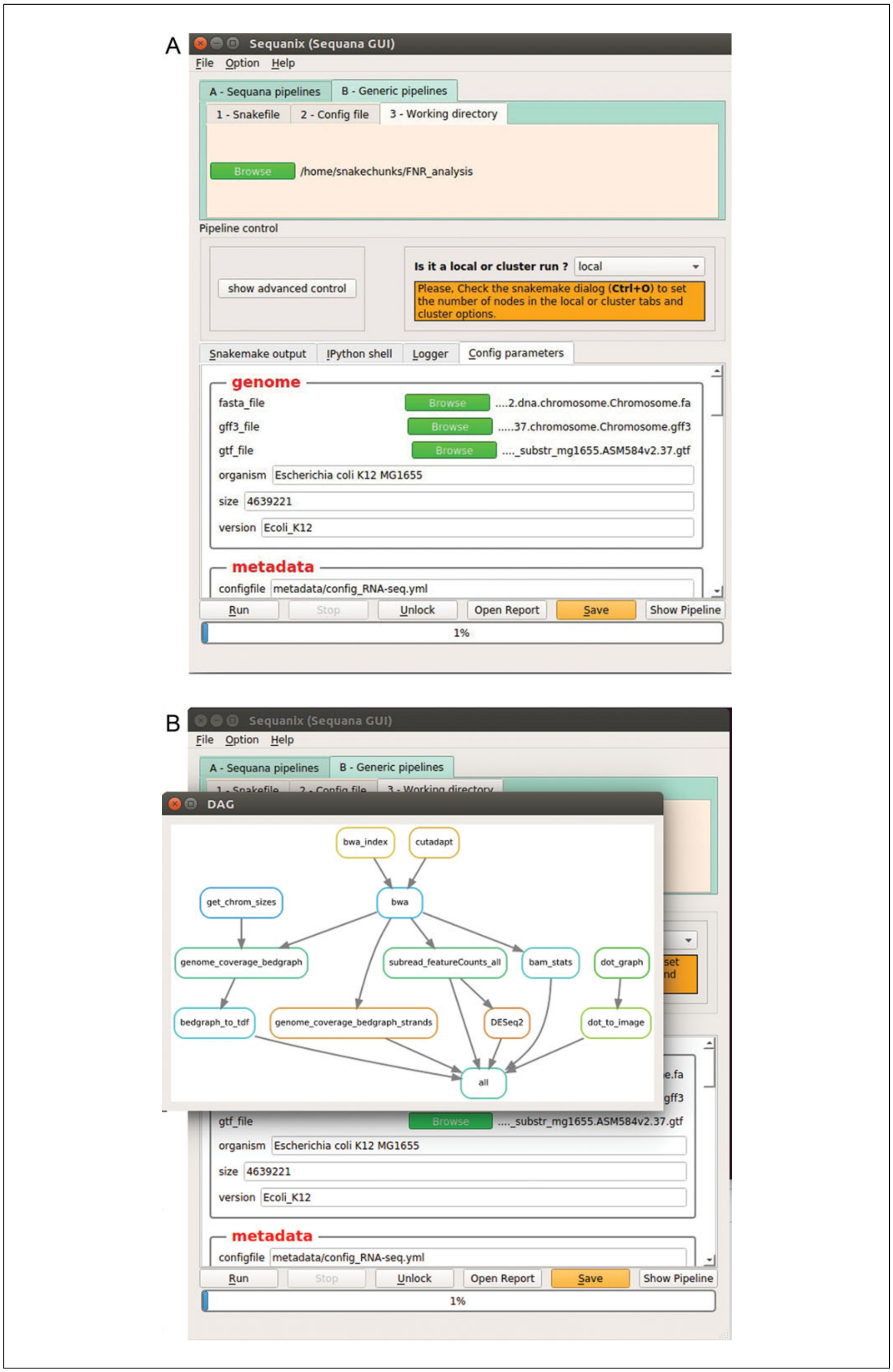
4. Click on menu File, select Open session..., and select the session file `metadata/igv_session.xml` in the FNR analysis directory.

   This will load an IGV session with our selection of relevant tracks for the interpretation of ChIP-seq and RNA-seq results, which are discussed further in the Guidelines for Understanding Results below.

ALTERNATE PROTOCOL

RUNNING THE WORKFLOW WITH THE USER-FRIENDLY INTERFACE SEQUANIX

Sequanix (Desvillechabrol et al., 2018) is a graphical user interface (GUI) based on PyQt, developed to facilitate the execution of NGS Snakemake pipelines. It was originally designed to run workflows included in the Sequana project (*http://sequana.readthedocs.io*),



/home/snakechunks/FNR_analysis

Is it a local or cluster run ? local

Please, Check the snakemake dialog (**Ctrl+O**) to set the number of nodes in the local or cluster tabs and cluster options.

fasta_file 2.dna.chromosome.Chromosome.fa

gff3_file 37.chromosome.Chromosome.gff3

gtf_file ...._substr_mg1655.ASM584v2.37.gtf

organism Escherichia coli K12 MG1655

size 4639221

version Ecoli_K12

configfile metadata/config_RNA-seq.yml

1%

gtf_file ...._substr_mg1655.ASM584v2.37.gtf

organism Escherichia coli K12 MG1655

size 4639221

version Ecoli_K12

configfile metadata/config_RNA-seq.yml

1%

Figure 7 Sequanix graphical user interface. (**A**) Configuration of the workflow parameters. (**B**) Display of workflow wiring. The diagram shows the directed acyclic graph (DAG) of rules automatically generated by Snakemake.

but can also handle any Snakemake pipeline. Thanks to the graphical interface, the parameters can be customized easily and the workflows can be run without using any command line.

Here we demonstrate the execution of the RNA-seq workflow (see Basic Protocol 3) using this interface.

Necessary Resources

Conda: If not already done, create and activate a Conda environment (Strategic Planning, steps 1 to 10)

Sequana: Install Sequana: type `conda install -c bioconda sequana=0.7.1`

RNA-seq dataset: If not already done, download the RNA-seq dataset (Basic Protocol 3, step 1 and 2) to install the metadata and download RNA-seq raw reads

1. Launch Sequanix.

   ```
   sequanix
   ```

2. At the top of the Sequanix window, select the tab "Generic pipelines."
3. Under the Snakefile tab, fetch the workflow file `RNA-seq_complete.wf` in the directory `SnakeChunks/scripts/snakefiles/workflows`.
4. Under the Config file tab, fetch the configuration file `config_RNA-seq.yml` in the directory `metadata.`
5. Under the Working directory tab, select the directory you defined above as `$ANALYSIS_DIR` (Strategic Planning, step 5) (Fig. 7A).
6. In the menu of the application, select Options > Snakemake options . . . > General, and type "`--use-conda`" in the bottom box "other options," then press OK.
7. In the Sequanix main window, press Save.
8. Press Show pipeline to check that everything looks reasonable (Fig. 7B).
9. Press Run.

   If you have followed Basic Protocol 3, the Run button should not start any new analysis, because Snakemake will detect that the result files are already present. If not, Sequanix will run the workflow just as in the terminal.

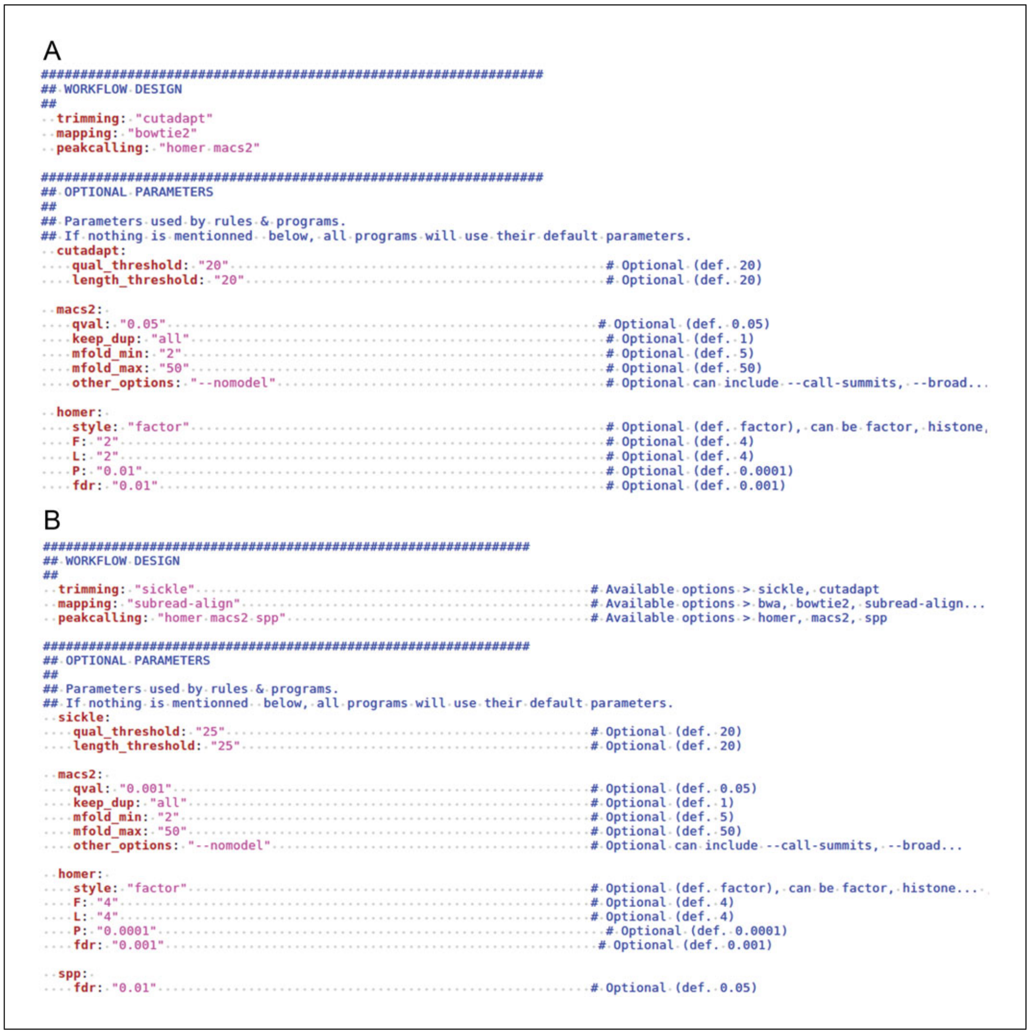
CUSTOMIZATION OF PARAMETERS

SUPPORT PROTOCOL

Each workflow available in SnakeChunks requires three basic files in order to specify the input data files and all the parameters of an analysis. These files have been placed in a directory named "metadata." We explain here how to adapt the ChIP-seq metadata files, but the same principle applies to the RNA-seq and integration workflows. The ChIP-seq workflow runs using three metadata files:

- Sample file: `samples_ChIP-seq.tab;`
- Design file: `design_ChIP-seq.tab`;
- Workflow and tool parameters: `config_ChIP-seq.yml` (Fig. 8A).

The sample file (Table 1) describes each sample to be analyzed (one row per sample), with two mandatory columns (ID and Condition) and optional columns for complementary information such as GSM identifiers. Here, we have two samples: one ChIP-ped with FNR, and a control sample labeled "input" following the ChIP-seq convention.



A

```
################################################################
## WORKFLOW DESIGN
##
  trimming: "cutadapt"
  mapping: "bowtie2"
  peakcalling: "homer macs2"

################################################################
## OPTIONAL PARAMETERS
##
## Parameters used by rules & programs.
## If nothing is mentionned  below, all programs will use their default parameters.
  cutadapt:
    qual_threshold: "20"                                                # Optional (def. 20)
    length_threshold: "20"                                              # Optional (def. 20)

  macs2: 
    qval: "0.05"                                                       # Optional (def. 0.05)
    keep_dup: "all"                                                     # Optional (def. 1)
    mfold_min: "2"                                                      # Optional (def. 5)
    mfold_max: "50"                                                     # Optional (def. 50)
    other_options: "--nomodel"                                          # Optional can include --call-summits, --broad...

  homer: 
    style: "factor"                                                     # Optional (def. factor), can be factor, histone,
    F: "2"                                                              # Optional (def. 4)
    L: "2"                                                              # Optional (def. 4)
    P: "0.01"                                                           # Optional (def. 0.0001)
    fdr: "0.01"                                                         # Optional (def. 0.001)
```

B

```
##############################################################
## WORKFLOW DESIGN
##
  trimming: "sickle"                                                  # Available options > sickle, cutadapt
  mapping: "subread-align"                                            # Available options > bwa, bowtie2, subread-align...
  peakcalling: "homer macs2 spp"                                      # Available options > homer, macs2, spp

##############################################################
## OPTIONAL PARAMETERS
##
## Parameters used by rules & programs.
## If nothing is mentionned  below, all programs will use their default parameters.
  sickle:
    qual_threshold: "25"                                              # Optional (def. 20)
    length_threshold: "25"                                            # Optional (def. 20)

  macs2: 
    qval: "0.001"                                                     # Optional (def. 0.05)
    keep_dup: "all"                                                   # Optional (def. 1)
    mfold_min: "2"                                                    # Optional (def. 5)
    mfold_max: "50"                                                   # Optional (def. 50)
    other_options: "--nomodel"                                        # Optional can include --call-summits, --broad...

  homer: 
    style: "factor"                                                   # Optional (def. factor), can be factor, histone... 
    F: "4"                                                            # Optional (def. 4)
    L: "4"                                                            # Optional (def. 4)
    P: "0.0001"                                                         # Optional (def. 0.0001)
    fdr: "0.001"                                                       # Optional (def. 0.001)

  spp: 
    fdr: "0.01"                                                       # Optional (def. 0.05)
```

Figure 8 YAML-formatted configuration file for the ChIP-seq workflow. The YAML format enables the user to specify all the parameters of a workflow in a structured way while being human readable and easily editable. (**A**) Default configuration. (**B**) Customized configuration.

The design file (Table 2) defines the samples to be compared in order to perform peak calling. Here, we are going to perform peak calling of the ChIP sample, using the input sample as a background control. For RNA-seq, the design defines the conditions to be compared.

The configuration file (Fig. 8A) is specific to the workflow to be run. It contains three main parts: (1) general information about the reference genome, metadata file, and file organization; (2) general design of the workflow, such as the steps to be performed (trimming, mapping, peak calling, annotation) and the tools to be used at each step; and (3) an optional section enabling to customize the parameters used for each tool (if not specified, their default parameters are used).

Below, we explain how to edit the configuration file in order to generate alternative results, using different tools and parameters.

IMPORTANT NOTE: Be aware that performing alternative trimming and/or mapping can require additional disk space, since FASTQ files (raw reads, trimmed reads) and BAM files (aligned reads) are very space consuming. In the following protocol, that requires about 2 Gb of disk space, but this can go as high as tens of gigabases in the case of larger raw files, such as the RNA-seq files analyzed in Basic Protocol 3.

1. Create a copy of the ChIP-seq config file.

```
cd $ANALYSIS_DIR; \
cp metadata/config_ChIP-seq.yml metadata/config_
  ChIP-seq_custom.yml
```

2. With a text editor, make the following changes to your custom configuration file (`metadata/config_ChIP-seq_custom.yml`).
   a. Change the trimming software from cutadapt to sickle.
   b. Change the mapping software from bowtie2 to subread-align.
   c. Add the SPP peak caller to Homer and Macs2.
   d. Customize the SPP, Homer, and Macs2 parameters in the third section according to the values shown in Figure 8B.

   Alternatively, you can avoid manual editing of parameters by copying the ready-to-use customized configuration file provided in the distribution. To do this, skip step 2 and instead run the following command:

```
cp metadata/config_ChIP-seq_advanced.yml metadata/
  config_ChIP-seq_custom.yml
```

3. Run the commands below, which correspond to steps 5 and 7 of Basic Protocol 1, and step 1 of Basic Protocol 2, 1.5, 1.7, and 2.1 adapted to use the custom configuration file.

```
snakemake \
-s SnakeChunks/scripts/snakefiles/workflows/quality_
  control.wf \
--configfile metadata/config_ChIP-seq_custom.yml -p
  --use-conda -j 2;
snakemake \
-s SnakeChunks/scripts/snakefiles/workflows/mapping.
  wf \
--configfile metadata/config_ChIP-seq_custom.yml -p
  --use-conda -j 2;
snakemake \
-s SnakeChunks/scripts/snakefiles/workflows/
  ChIP-seq_RegulonDB.wf \
--configfile metadata/config_ChIP-seq_custom.yml -p
  --use-conda -j 2
```

4. Visualize the differences in the IGV: load a session as in Basic Protocol 4, steps 3 and 4.

5. Click on the menu File, select "Load from File . . . ," and select the following peak files:

```
$ANALYSIS_DIR/ChIP-seq/results_advanced/peaks/FNR1_
  vs_input1/spp/FNR1_vs_input1_sickle_subread-align_
  spp.bed
$ANALYSIS_DIR/ChIP-seq/results_advanced/peaks/FNR1_
  vs_input1/homer/FNR1_vs_input1_sickle_subread-
  align_homer.bed
$ANALYSIS_DIR/ChIP-seq/results_advanced/peaks/FNR1_
  vs_input1/macs2/FNR1_vs_input1_sickle_subread-
  align_macs2.bed
```

   By running the command `wc -l` on these files, you can note the influence of the choice of peak caller, as well as its parameters.

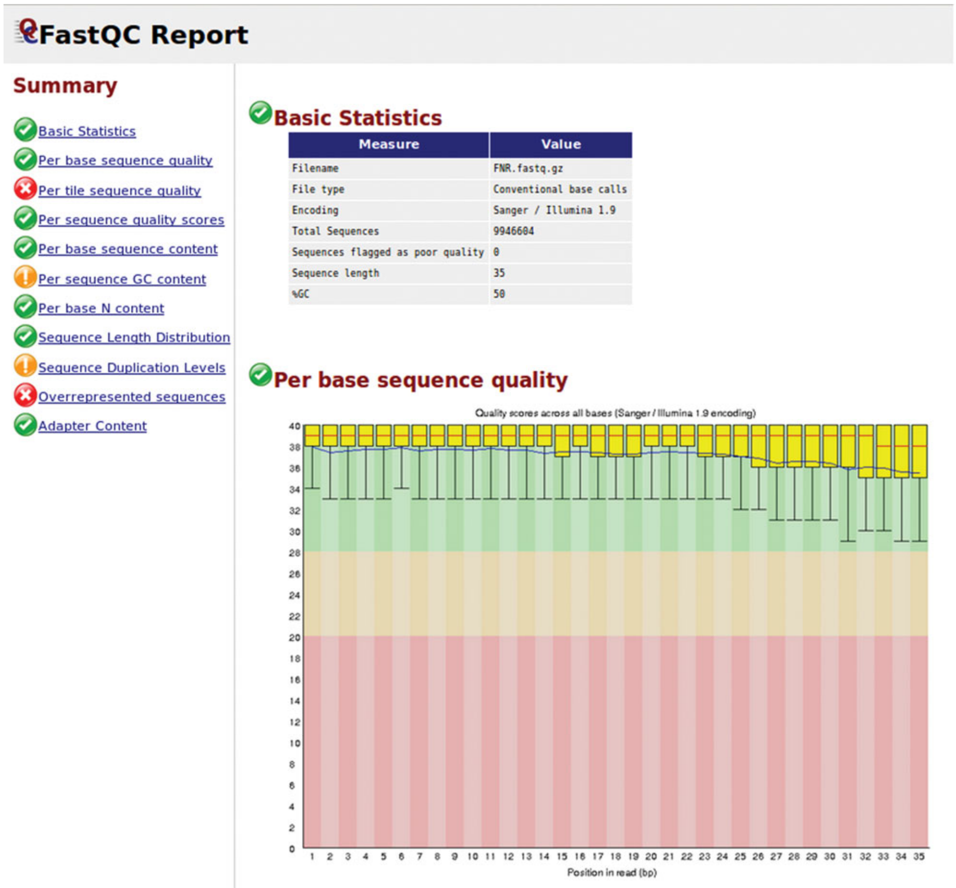
GUIDELINES FOR UNDERSTANDING RESULTS

Data Preprocessing and Read Mapping (Basic Protocol 1)

Quality control

For each sample, FastQC produces a box plot representing per-base sequence quality. A common phenomenon in high-throughput sequencing is a decrease in sequence quality at the 3′ end of the reads. This can indeed be observed for the input sample in our case study (Fig. 9). Low read quality can reduce the percentage of reads mapped on the reference genome. To avoid this, we recommend performing sequence trimming to remove low-quality read extremities.

Another interesting category of information in FastQC reports is the sequence-duplication levels. The graph outlines read sequences found in an excessive number of copies, which may diagnose an effect of PCR amplification due to poor complexity of the DNA library. Note that duplication is often interpreted in contexts in which the sequence library is much smaller than the genome size (typically ∼50 M reads for a ∼3-Gb mammalian genome), so that reads resulting from a random sampling are not expected to fall on exactly the same genomic position. When studying bacterial regulation, however, library size can exceed genome size (typically 4 Mb) so that multiple matches are expected along



FastQC Report

Summary

- ✓ Basic Statistics
- ✓ Per base sequence quality
- ✗ Per tile sequence quality
- ✓ Per sequence quality scores
- ✓ Per base sequence content
- ! Per sequence GC content
- ✓ Per base N content
- ✓ Sequence Length Distribution
- ! Sequence Duplication Levels
- ✗ Overrepresented sequences
- ✓ Adapter Content

✓ Basic Statistics

Measure	Value
Filename	FNR.fastq.gz
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	9946604
Sequences flagged as poor quality	0
Sequence length	35
%GC	50

✓ Per base sequence quality

Figure 9 Quality report of the FNR1 ChIP-seq raw reads before trimming. The abscissa (columns) corresponds to nucleotide positions along the mapped reads; the ordinate indicates read quality scores. For each position, statistics are summarized for all the reads of a library: median (red line), interquartile range (yellow box), and quality range (vertical line). Background colors indicate an arbitrary subdivision of quality scores, from red (insufficient) to green (good).

the whole genome. Another section of the FastQC report provides statistics about overrepresented sequences. Before removal of the adapters by cutadapt (Basic Protocol 1, step 5), Illumina adapters represent respectively 0.5% and 2.6% of the total number of reads of the FNR1 and input1 samples. After cutadapt is run, these sequences are gone (Basic Protocol 1, step 6). Detailed information on the interpretation of read quality is provided on the FastQC Web site (*http://www.bioinformatics.babraham.ac.uk/projects/fastqc/*).

Read mapping

Using the bowtie2 algorithm, the trimmed reads in FASTQ format are aligned onto a genome of reference, downloaded as described in Strategic Planning. In our case, the reference is *E. coli* K-12. The result of the alignment comes in a BAM format that retains all the information from the fastq files about read sequences and quality, but adds the putative positions of the reads in the reference genome.

Genome coverage

Genome coverage files makes it possible to visualize the mapped reads in a condensed way, by showing the number of reads overlapping each position on each strand of the reference genome (Fig. 10A, pink, gray, and jade tracks in the middle panel) or their sum on both strands (purple track). Coverage profiles can be stored in different file formats (e.g., tdf, bedgraph, bigwig) depending on the size of the dataset and the way to display it. In this protocol, we use the TDF format, which is the recommended format for optimal IGV visualization.

ChIP-seq (Basic Protocol 2)

Peak calling

The peaks detected by Homer and Macs2 can be visualized in IGV as BED files. This file format contains essentially the coordinates of the regions with a high density of mapped reads, which are called “peaks.” Although in bacteria it is expected that ChIP-seq peaks will fall into intergenic regions upstream of the regulated genes, it has been shown that a surprisingly high amount of binding may occur into coding or downstream regions (Galagan, Lyubetskaya, & Gomes, 2012). This observation should be interpreted by taking into account the fact that bacteria have a very small proportion of intergenic regions (10% to 15% of the genome).

Figure 10A shows a very clear peak around position 2,344,000, detected by both peak callers, in the noncoding region upstream of the gene *nrdA*. On comparing the ChIP-seq read coverage on the forward and reverse strands (pink tracks in the middle panel), we see a shift between forward and reverse peaks. This typical pattern is consistent with the expectation for ChIP-seq experiments, because immunoprecipitated fragments are sequenced at their extremities, so that the reads are expected to be found either on the forward strand to the left of the binding site, or on the reverse strand to its right.

Different peak-calling tools can produce very different results for the same dataset. In the same region (Fig. 10A), Macs2 detects another peak around position 2,347,000, associated with the gene *nrdB*, which belongs to the same operon as *nrdA*. It is not identified as a peak by Homer, and it is not associated with any known FNR TF binding sites from RegulonDB. However, RegulonDB indicates that *nrdB* is regulated by H-NS and Fis, nucleoid-associated proteins (NAPs) that are known to mask FNR binding sites under anaerobic conditions (Myers et al., 2013). Although barely detected by peak callers, this site is thus supported by some experimental evidence.

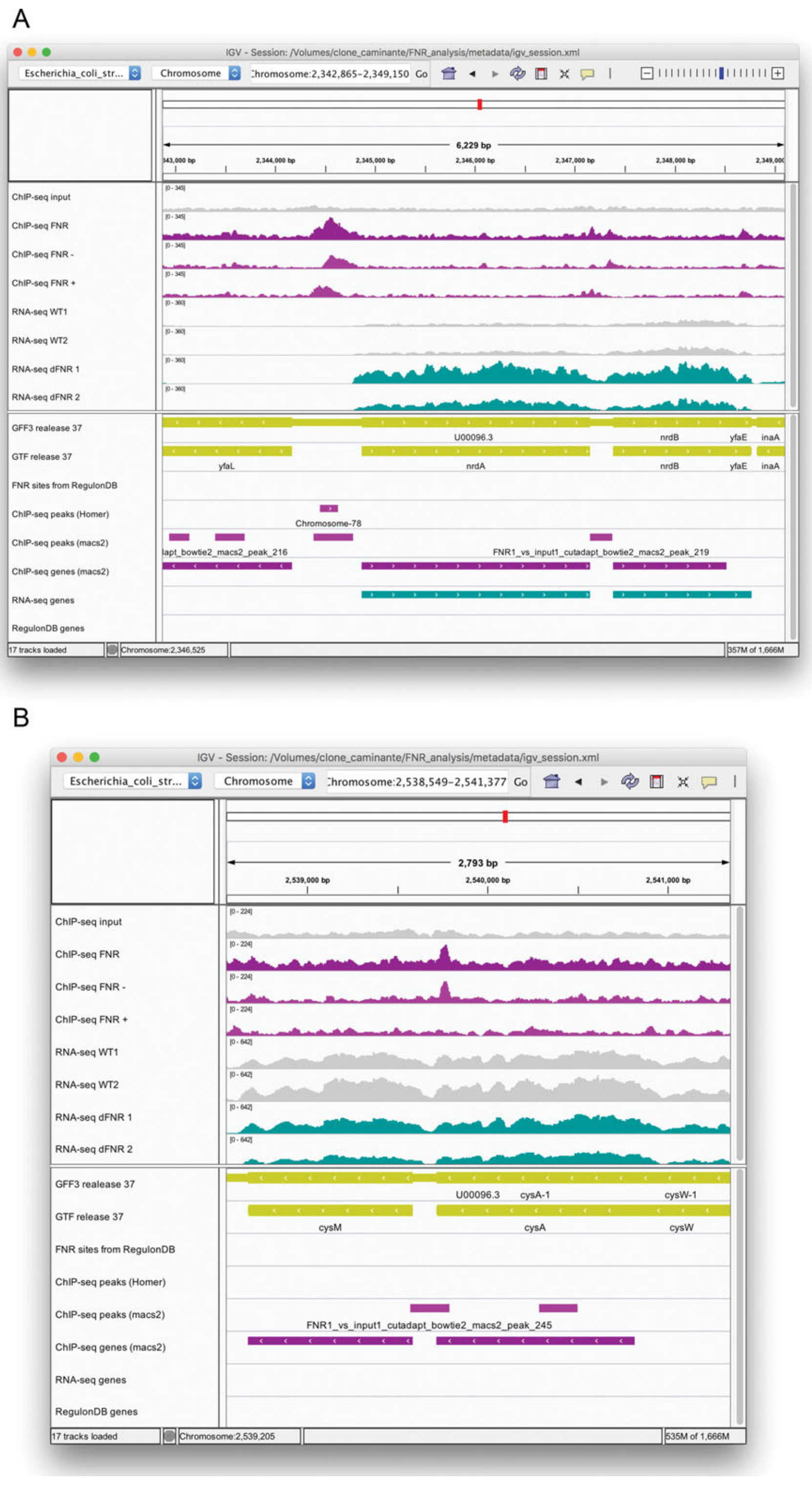


Figure 10 Snapshots of ChIP-seq results for selected genomic regions. The figures were generated with the Integrative Genomics Viewer (IGV). (**A**) High-confidence peak in the promoter region of the nrdAB operon. Note the characteristic shift between reads mapped on the plus and minus strands. (**B**) Example of a peak that is likely to be a false positive. For both IGV maps (A and B), the top panels show the coordinates of the displayed genomic region. The middle panels show read density profiles in the input (gray) and ChIP-seq samples (purple for strand-insensitive, pink for strand-sensitive profiles), and RNA-seq data (WT in gray, FNR mutants in turquoise). The lower panels show annotation tracks for genes (yellow), annotated FNR binding sites (none found in the displayed regions), and binding peaks.

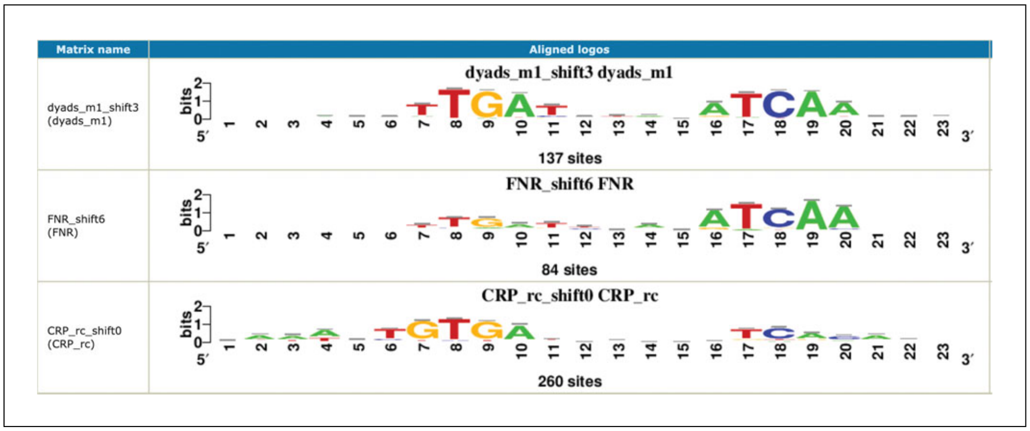


Figure 11 Most significant motif discovered by RSAT peak motifs in the FNR peaks, aligned with matching motifs in RegulonDB.

In contrast, Figure 10B shows a typical example of a peak that is likely to be a false positive. Note that its read enrichment is restricted to the reverse strand and falls within the coding region of a gene. Strand-specific display of read coverage thus makes it possible to assess the reliability of peaks by inspecting their distribution around the putative binding sites.

The number of peaks and their width can vary considerably, hence the need to adapt the tools to a given study and assess the relevance of the downstream results. Under our working conditions, Homer returns 161 peaks of equal width (exactly 177 bp each), whereas Macs2 returns 411 peaks ranging from 200 to 5893 bp (with an average of 475 bp), an obviously excessive size for TF binding sites. The broadest peaks reported by Macs2 correspond to wide regions covering several genes, which are entirely covered by reads in the ChIP-seq sample, and indeed enriched with respect to the genomic input, but which likely do not correspond to TF binding sites. For Macs2, the number of peaks can be strongly modified by tuning the q-value threshold and the minimal fold change. For example, the number of peaks drops from 547 with a q-value threshold of 0.05 and a minimal fold-change of 2, to 159 with q-value threshold of 0.001 and a minimal fold change of 5. The most permissive conditions give fewer relevant peaks, denoted by a drop in the significance of the FNR motif. In summary, the choice of a peak-calling algorithm and the fine-tuning of its parameters crucially affect ChIP-seq results, and should be evaluated case by case.

Motif discovery in peak sequences

The top panel of Figure 11 shows the most significant motif returned by RSAT peak-motifs (Thomas-Chollier et al., 2012) in the sequences of Homer peaks. This motif was discovered by the tool dyad-analysis (van Helden, Ríos, & Collado-Vides, 2000), which detects over-represented pairs of spaced oligonucleotides. This motif discovery approach is particularly relevant for bacteria, where most transcription factors form homodimers that bind spaced motifs. The comparison of this discovered motif with all the TF binding motifs annotated in RegulonDB returns two matches, corresponding to FNR and CRP, respectively. The alignment highlights the strong similarity between the motifs recognized by FNR and CRP (they differ only by one nucleotide at position 7 of the motif alignment), which is consistent with the fact that these two factors are known to co-regulate a number of genes (Gama-Castro et al., 2016; Myers et al., 2013).

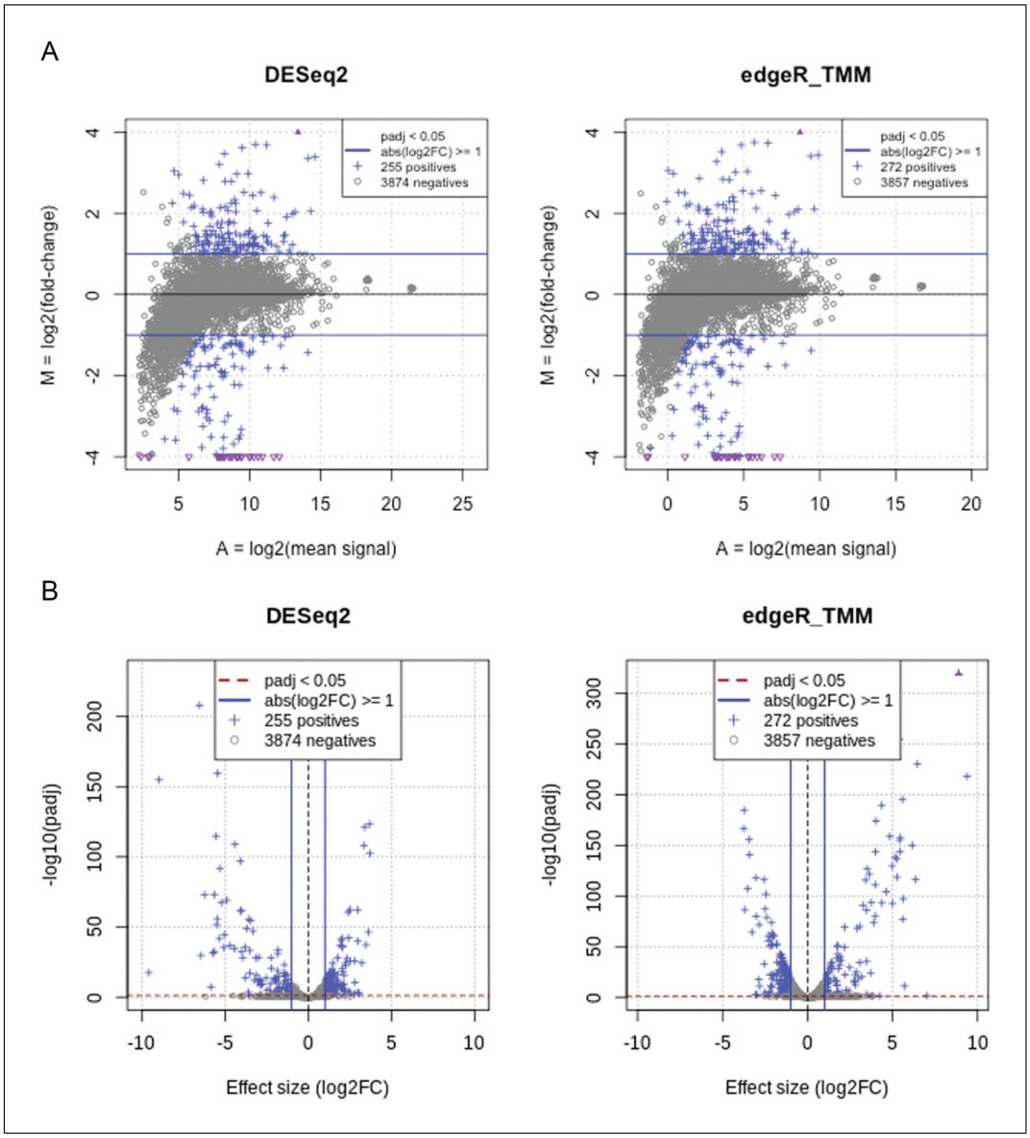


Figure 12 Global views of the results for the detection of differentially expressed genes between FNR mutant versus wild-type. These plots are generated as part of the differential analysis step, using an R script. Left and right panels respectively show the results of DESeq2 and edgeR. (A) MA plots. The abscissa indicates the mean level of expression (average of the log-transformed counts), and the ordinate shows the log fold change between FNR mutant and wild-type strain, which indicates the level of over- (positive values) or underexpression (negative values). Differentially expressed genes (DEGs), i.e., those passing both the effect size and significance thresholds, are highlighted in blue. Triangles indicate genes whose $\log_2$ fold change exceed the plot limits. (B) Volcano plots. The abscissa represents the log fold change, which indicates the size of the effect and its sign (–, downregulation; +, upregulation). The ordinate shows the significance of the differential expression (negative log of the adjusted *P* value).

RNA-seq (Basic Protocol 3)

Differentially expressed genes

The results of the RNA-seq analysis are summarized in an HTML report (`RNA-seq/results/diffexpr/cutadapt_bwa_featureCounts_rna-seq_deg_report.html`), which can be visualized using a web browser. It features information and statistics about the RNA-seq samples, read counts, and differentially expressed genes, detected by using two different tools: DESeq2 (Love et al., 2014) and edgeR (Robinson et al., 2010). Figure 12 shows MA plots and volcano plots that are automatically produced by the workflow to provide a synthetic representation of the global results of the RNA-seq differential analysis. The MA plots (Fig. 12A) indicate the relationship between the mean level of expression of each gene (abscissa) and its differential expression, measured as the log fold difference between FNR mutant and wild type (ordinate). The genes declared differentially expressed between the two conditions (WT versus FNR) are highlighted as blue crosses. Genes overexpressed and underexpressed in the FNR mutants appear above or below the *x* axis, respectively. The volcano plots (Fig. 12B) provide a combined view of the expression changes (log fold change, on the abscissa) and the statistical significance of these changes (on the ordinate). The significance is computed as the negative logarithm of the adjusted *P* values reported by DESeq2 (left) and by edgeR (right), respectively. High values are indicative of significant differences of expression between FNR mutant and WT strains. To select differentially expressed genes, SnakeChunks combines user-modifiable thresholds on the adjusted *P* value (default: $\alpha = 0.05$) and on the fold change (default: at least twofold over- or underexpression).

In total, these thresholds lead to the retention of 278 differentially expressed genes that were declared positive by either DESeq2 (255 genes) or edgeR (272 genes). This number is consistent with the fact that FNR acts as global regulator in *E. coli*. Note that we chose to keep the union of both lists in order to favor sensitivity, but this can be parameterized in the configuration file by specifying that the detection of differentially expressed genes relies on edgeR, DESeq2, their intersection, or their union.

Integration (Basic Protocol 4)

The Venn diagram generated by the workflow (Fig. 13, file `integration/ChIP-RNA-regulons_venn.png`) shows the number of *E. coli* genes associated with FNR peaks in the ChIP-seq experiment (pink), reported as differentially expressed in the RNA-seq analysis (green), or annotated as FNR targets in RegulonDB (violet), as well as the intersections between these gene sets. Supporting Information Tables S1 and S2 provide the complete data table used to generate these Venn diagrams. Depending on the peak-calling algorithm, the number of genes found at the intersection between the three gene lists (ChIP-seq, RNA-seq, and RegulonDB) will be quite small (38 for Macs2 peaks and 28 for Homer peaks) relative to the respective size of the compared gene sets. It is interesting to consider an interpretive guideline for the pairwise intersections or set memberships. The genes reported by both ChIP-seq (FNR binding) and RNA-seq (FNR transcriptional response) but not annotated in RegulonDB are likely to be direct FNR target genes, and might be considered to be added to RegulonDB, in an annotation track based on combined evidence from complementary high-throughput experiments. This would give 29 genes with Macs2 peaks and 25 with Homer peaks. It would be interesting to furthermore scan their promoter sequences in order to search instances of the FNR binding motif in order to predict binding-site locations, and consolidate the results. The genes detected as differentially expressed (RNA-seq) without any annotated FNR site (RegulonDB) or associated peak (Figure 13, pale green, on the Venn diagrams of Figure 13, covering, respectively, 160 and 167 genes for Macs2 and Homer) include genes located inside the target operons of FNR. Indeed, in bacteria, polycistronic transcripts are regulated by *cis*-acting elements located in the promoter of the operon leader gene. Consistently with this, 38 of these 167 genes (~23% when the analysis is led with Homer) have a very short upstream noncoding region (<55 bp) typical of intra-operon genes, whereas almost all the genes of the triple intersection (28 of 29) have larger upstream sequences typical of operon-leader genes. The remaining 77% of differentially expressed genes without associated ChIP-seq peak are likely to be indirect FNR targets, whose transcription might be affected via intermediate transcription factors that are themselves regulated by FNR. The genes associated with ChIP-seq peaks without transcriptional response (334 for Macs2, 119 for Homer) likely result from different effects:

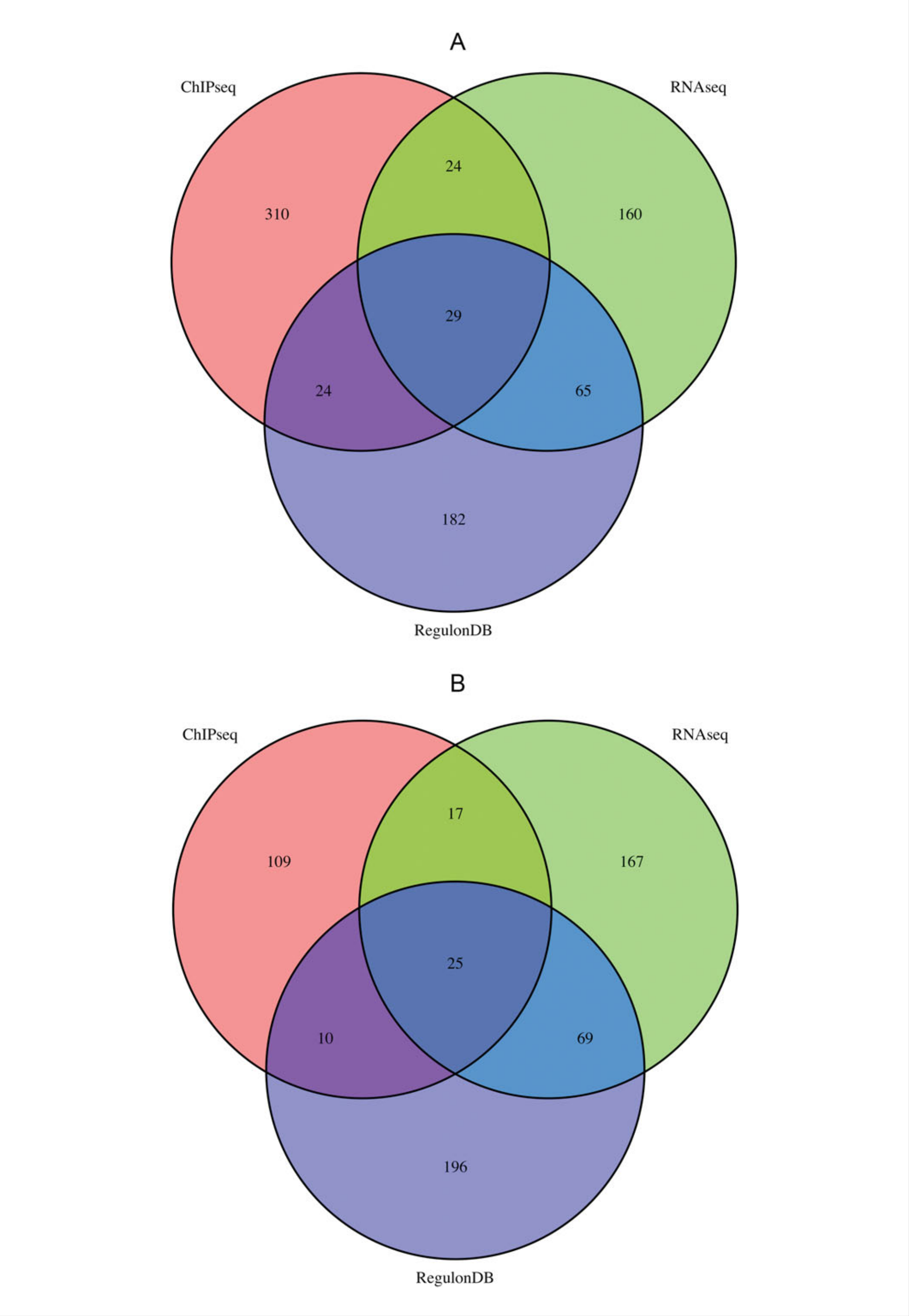


Figure 13 Integration of ChIP-seq, RNA-seq results, and RegulonDB annotations. Venn diagrams show the intersections of the genes linked to ChIP-seq peaks (pink), those declared differentially expressed by the RNA-seq experiment (green), and those annotated as FNR target genes in RegulonDB (violet). These diagrams are automatically generated by the integration workflow, using the R library VennDiagram. (A) Results with the 411 ChIP-seq peaks reported by Macs2 with $q < 0.01$ and fold change between 2 and 50. (B) Results with the 166 ChIP-seq peaks reported by Homer.

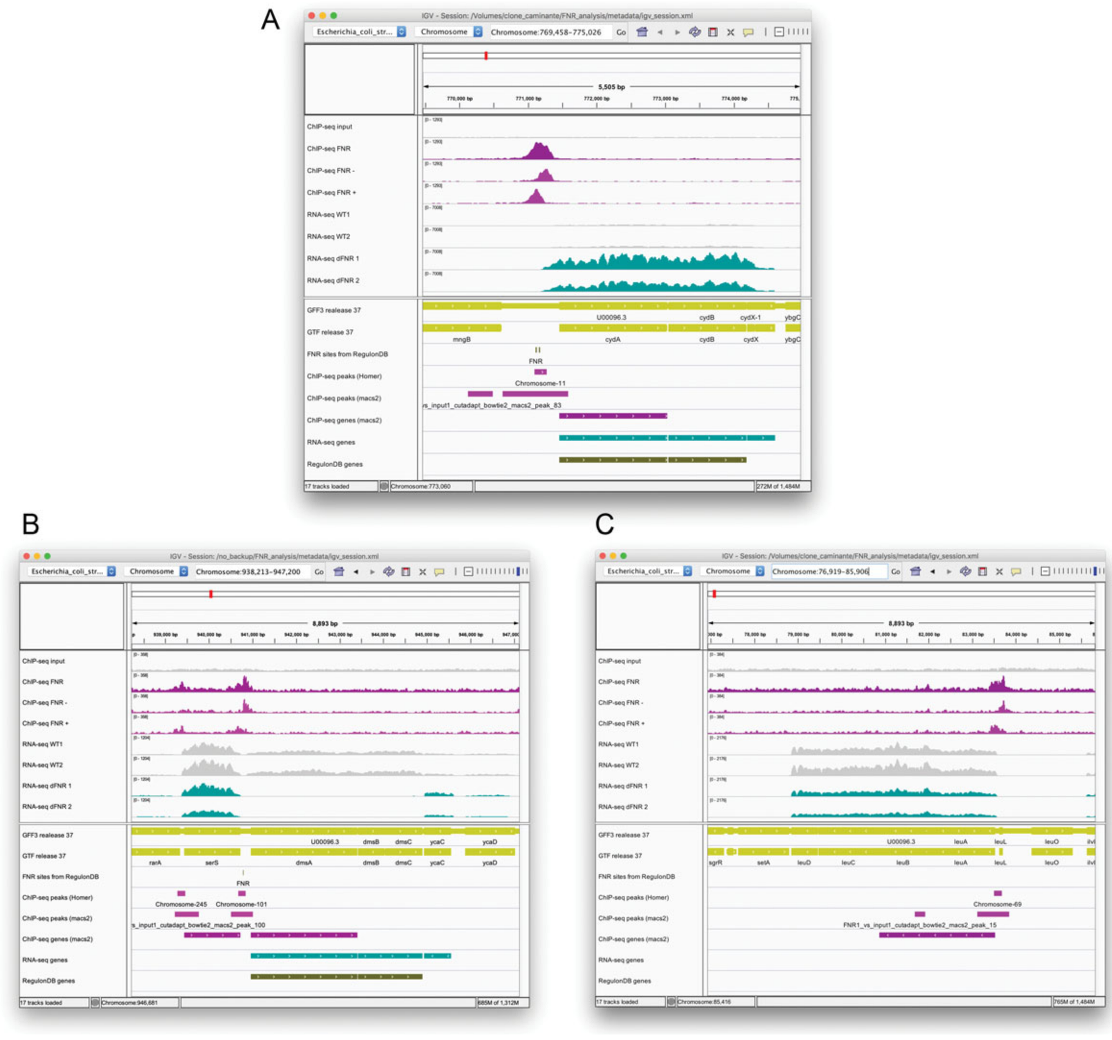


Figure 14 IGV snapshots of RNA-seq results for three illustrative operons. Middle panel, genome coverage profiles for the two replicas of the wild-type (gray) and FNR mutant (jade). Lower panel, genome annotations for the genes (yellow), FNR binding sites from RegulonDB (gray), differentially expressed genes (jade), and FNR target genes annotated in RegulonDB (dark olive). Shown are views of selected regions encompassing (A) the cydABX operon, (B) the dmsABC operon, and (C) the leuLABCD operon.

nonfunctional binding of the FNR factor under the experimental conditions of the study (missing co-activator, co-binding of a repressor); binding between two divergently transcribed transcription units, but regulating only one of them; or false positives from peak calling (e.g., regions with a high density of reads on one strand only, as discussed above).

Figure 14 highlights some illustrative examples of differentially expressed genes detected by DESeq2 or edgeR. For the cydABX operon (Fig. 14A), the FNR mutant (jade tracks on the genome coverage profiles) has an increased level of expression compared to the wild-type (gray tracks). Consistently with that result, this operon is repressed by FNR (Salmon et al., 2003), and it has two annotated FNR binding sites in RegulonDB, which overlap a strong peak detected by both Homer and Macs2 in the ChIP-seq results.

The dmsABC operon also exemplifies the genes found at the triple intersection: it is regulated by FNR (Melville & Gunsalus, 1996), and, consistently, it has one TF binding site listed in RegulonDB, and is reported by both the ChIP-seq and RNA-seq experiments (Fig. 14B).

A more subtle example is the leuLABCD operon (Fig. 14C): RNA-seq coverage profiles also reveal reduced expression, although the differential expression analysis did not report the presence of any significant gene, due to the stringent thresholds applied to both adjusted *P* value (<0.05) and fold change (>2). This operon encodes the enzymes responsible for the biosynthesis of leucine from valine. It has no binding sites annotated in RegulonDB for the FNR transcription factor, and based on the RNA-seq results only, several possibilities could be invoked to explain this inconsistency: the leu operon might (i) be indirectly regulated by FNR via another transcription factor, (ii) be a direct target of FNR whose binding sites have not yet been characterized, or (iii) be a false-positive. This situation can be clarified by analyzing the ChIP-seq profiles, since we observe a clear peak upstream of the operon, detected by both Macs2 and Homer (Fig. 14C), supporting the evidence for a direct regulation of the leu operon by FNR.

In summary, a detailed analysis and human-based interpretation of combined RNA-seq and ChIP-seq data is worthwhile as a means to go beyond the gene lists returned by the automatic comparison of target genes predicted by ChIP-seq and RNA-seq experiments.

COMMENTARY

Background Information

Next-generation sequencing (NGS) technologies (Schuster, 2007) emerged in 2007 with the development of several approaches for massively parallel sequencing of short DNA sequences (a few tens of base pairs per sequence). This unprecedented gain in sequencing speed was mobilized for a wide variety of applications: genome sequencing, transcriptome (RNA-seq), genome-wide binding location analysis (ChIP-seq), chromatin conformation (Hi-C), metagenomics, and many others. Research projects based on NGS typically lead to the situation where the biologist performs experiments, sends the samples to a sequencing center, and receives a link to download several gigabases of raw sequences known as “short reads.” Since 2007, a wide variety of software tools has been developed to handle NGS data and extract relevant information (Pepke et al., 2009).

Proper use of such software requires a good understanding of their parameters, strengths, and weaknesses. Beyond the choice and parameterization of each particular tool, it has become crucial to formalize their wiring by implementing workflows that ensure traceability and reproducibility of all the steps used to produce the results from the raw data. Many alternative software systems can be used to manage the development and execution of analysis workflows. Among them, Galaxy (Goecks, Nekrutenko, & Taylor, 2010) became highly popular because it offers an immediate access through a graphical interface to biologists with no experience in the Unix terminal. Snakemake (Köster & Rahmann, 2012) offers a complementary solution to achieve the same goals—developing, managing, and running NGS workflows—in the Unix command-line environment. Snakemake is currently being adopted by a growing number of bioinformaticians as well as experimental biologists willing to get one step further in the analysis of their own data. The goal of SnakeChunks is to facilitate the conception and use of NGS workflows by encapsulating Snakemake commands in a library of modular rules (one per tool) that can be combined in various ways to build and customize workflows (Fig. 2).

Critical Parameters

Control samples

When analyzing binding signals (ChIP-seq) or transcription signals (RNA-seq), it is crucial to generate appropriate control experiments, in order to measure differences in signal against a proper background signal, and thus avoid the detection of false positives. This is especially important when analyzing ChIP-seq data, since false peaks can arise from biases in the experiments: nonhomogeneous sonication of DNA due nonhomogeneous aperture of the chromatin, GC biases arising during PCR amplification of the fragments, low-complexity regions of the genome, and so on. Different types of controls can be used to estimate the background probabilities of read mapping in the different regions of the genome, including (1) sequencing genomic DNA without immunoprecipitation; (2) using “mock IP,” i.e., performing the immunoprecipitation with a nonspecific antibody; or (3) artificially knocking out the expression of the TF of interest. Irrespective of the method used, the control sequences are generally denoted as "input" for the peak-calling programs. In the study by Myers et al. (2013), genomic DNA was used as input. In the case of RNA-seq, knocked-out TFs or overexpressed TFs can be compared against WT samples. In this study, samples with an inactivated FNR protein were compared against WT strains.

Number of replicates

When performing biological experiments, it is crucial to account for the unavoidable variability intrinsic to living organisms. RNA-seq experiments are no exception, and it has been demonstrated that the greater the number of replicates, the more sensitive the detection of differentially expressed genes (Schurch et al., 2016). Designing experiments with a high number of replicates enables the analysis to distinguish subtle but relevant changes in expression from spurious fluctuations due to biological variability.

Choice of a read mapper

Read mapping is generally the most time- and resource-consuming task of RNA-seq and ChIP-seq data analysis. For the FNR study case developed in this article, the complete ChIP-seq workflow runs in a few minutes, whereas the RNA-seq workflows takes several hours. The modularity of the SnakeChunks library enabled us to run the same workflow with three alternative read-mapping tools: BWA (Li & Durbin, 2009), bowtie2 (Langmead & Salzberg, 2012), and subread-align (Liao, Smyth, & Shi, 2013). For this particular dataset, BWA runs approximately three times as fast as the two other algorithms, while giving very similar mapping rates. However, we experienced opposite rankings of tool performance with other datasets and reference genomes. The choice and parameterization of a read mapper should thus be considered as critical step, which has to be tuned in a case-specific way to optimize a workflow.

Troubleshooting

The Snakemake workflow management system is equipped with its own mechanisms for detecting, reporting, and fixing problems. Trouble is reported by red messages displayed on the terminal indicating the kind of problems and—when possible—suggested ways to fix them.

Advanced Parameters

Proper parameterization of the workflow is the key to optimize both computing efficiency and the biological relevance of the results.

Parameters can be changed either by modifying the YAML-formatted configuration file in the metadata (see Support Protocol) or with the option `--config` in the Snakemake command line (see example in Basic Protocol 1, step 3).

With the popularization of RNA-seq for transcriptome studies, the number of samples per research project has been expanding in recent publications. A crucial parameter will be the ability to keep up with increasing storage needs and to parallelize computation for large studies. The FNR case study discussed in this unit was intentionally selected for its small number of replicates per condition, but for wider-scale studies the number of simultaneous jobs handled by Snakemake should be adapted to the number of CPUs of the computing system (option `-j` option).

We also make a frequent use of the Snakemake option `-n`, which prints out all the commands required to complete a workflow, without actually executing them (as a dry run). This gives the user the ability to check that a command is properly parameterized before running it, which can be valuable when applying hours-long tasks to multiple samples.

Suggestions for Further Analysis

The main goal of the SnakeChunks library is to ensure the reproducibility of the analyses. This is why we recommend keeping a copy of the library with each dataset analyzed in order to ensure consistency between the results and the precise version of the library used to generate them. This is particularly crucial in the case of publication, so that readers can actually reproduce the analyses performed.

The use of Conda also enables the user to keep control over the software environment, and is in accordance with the FAIR Principles (Wilkinson et al., 2016).

A natural extension of this work will be to take advantage of SnakeChunks' flexibility in order to assess the impact of tool and parameter choice on the biological relevance of the results, and to optimize workflows by evaluating the correspondence between the lists of genes returned by combining ChIP-seq and RNA-seq results and those already annotated in RegulonDB for well-characterized transcription factors.

Acknowledgments

This work is funded by France Génomique, US National Institutes of Health grant

GM0110597, and FOINS-CONACYT–Fronteras de la Ciencia 2015, ID 15. L.C.K. is funded by a PhD grant from the Ecole Doctorale des Sciences de la Vie et de la Santé, Aix-Marseille Université.

We acknowledge the Institut Français de Bioinformatique (IFB) and Christophe Blanchet for the use of virtual machines on the IFB cloud, which enabled us to assess the portability and reproducibility of the workflows, as well as the Sequanix development team (Thomas Cokelaer, Dimitri Desvillechabrol, and Rachel Legendre), who helped us to port SnakeChunks to Conda and Sequanix.

Literature Cited

Andrews, S. (2010). FastQC: A quality control tool for high throughput sequence data. Retrieved from http://www.bioinformatics.babraham.ac.uk/projects/fastqc.

Blattner, F. R., Plunkett, G., Bloch, C. A., Perna, N. T., Burland, V., Riley, M., . . . Shao, Y. (1997). The complete genome sequence of *Escherichia coli* K-12. *Science*, *277*(5331), 1453–1462. doi: 10.1126/science.277.5331.1453.

Cadby, I. T., Faulkner, M., Cheneby, J., Long, J., van Helden, J., Dolla, A., & Cole, J. A. (2017). Coordinated response of the *Desulfovibrio desulfuricans* 27774 transcriptome to nitrate, nitrite and nitric oxide. *Scientific Reports*, *7*(1), 16228. doi: 10.1038/s41598-017-16403-4.

Castro-Mondragon, J. A., Rioualen, C., Contreras-Moreira, B., & van Helden, J. (2016). RSAT::Plants: Motif discovery in ChIP-seq peaks of plant genomes. *Methods in Molecular Biology 1482*, 297–322. doi: 10.1007/978-1-4939-6396-6_19.

Desvillechabrol, D., Legendre, R., Rioualen, C., Bouchier, C., van Helden, J., Kennedy, S., & Cokelaer, T. (2018). Sequanix: A dynamic graphical interface for Snakemake workflows. *Bioinformatics*, *34*(11), 1934–1936. doi: 10.1093/bioinformatics/bty034.

Feng, J., Liu, T., & Zhang, Y. (2011). Using MACS to identify peaks from ChiP-seq data. *Current Protocols in Bioinformatics*, *34*, 2.14.1–2.14.14. doi: 10.1002/0471250953.bi0214s34.

Galagan, J., Lyubetskaya, A., & Gomes, A. (2012). ChIP-Seq and the complexity of bacterial transcriptional regulation. In M. G. Katze (Ed.), *Systems biology* (pp. 43–68). Berlin, Heidelberg: Springer. Retrieved from https://link.springer.com/chapter/10.1007/82_2012_257.

Galagan, J. E., Minch, K., Peterson, M., Lyubetskaya, A., Azizi, E., Sweet, L., . . . Schoolnik, G. K. (2013). The *Mycobacterium tuberculosis* regulatory network and hypoxia. *Nature*, *499*, 178–183. doi: 10.1038/nature12337.

Gama-Castro, S., Salgado, H., Santos-Zavaleta, A., Ledezma-Tejeida, D., Muñiz-Rascado, L., García-Sotelo, J. S., . . . Collado-Vides, J. (2016). RegulonDB version 9.0: High-level integration of gene regulation, coexpression, motif clustering and beyond. *Nucleic Acids Research*, *44*(D1), D133–143. doi: 10.1093/nar/gkv1156.

Goecks, J., Nekrutenko, A., & Taylor, J., & Galaxy Team. (2010) Galaxy: A comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biology*, *11*(8), R86. doi: 10.1186/gb-2010-11-8-r86.

Heinz, S., Benner, C., Spann, N., Bertolino, E., Lin, Y. C., Laslo, P., . . . Glass, C. K. (2010). Simple combinations of lineage-determining transcription factors prime cis-regulatory elements required for macrophage and B cell identities. *Molecular Cell*, *38*(4), 576–589. doi: 10.1016/j.molcel.2010.05.004.

Jacob, F., & Monod, J. (1961). Genetic regulatory mechanisms in the synthesis of proteins. *Journal of Molecular Biology*, *3*, 318–356. doi: 10.1016/S0022-2836(61)80072-7.

Johnson, D. S., Mortazavi, A., Myers, R. M., & Wold, B. (2007). Genome-wide mapping of in vivo protein-DNA interactions. *Science*, *316*(5830), 1497–1502. doi: 10.1126/science.1141319.

Köster, J., & Rahmann, S. (2012). Snakemake—a scalable bioinformatics workflow engine. *Bioinformatics*, *28*, 2520–2522. doi: 10.1093/bioinformatics/bts480.

Langmead, B., & Salzberg, S. L. (2012). Fast gapped-read alignment with Bowtie 2. *Nature Methods*, 2012 Mar 4;*9*(4), 357–359. doi: 10.1038/nmeth.1923.

Lazazzera, B. A., Bates, D. M., & Kiley, P. J. (1993). The activity of the *Escherichia coli* transcription factor FNR is regulated by a change in oligomeric state. *Genes & Development*, *7*(10), 1993–2005. doi: 10.1101/gad.7.10.1993.

Li, H., & Durbin, R. (2009). Fast and accurate short read alignment with Burrows-Wheeler Transform. *Bioinformatics*, *25*, 1754–1760. doi: 10.1093/bioinformatics/btp324.

Liao, Y., Smyth, G. K., & Shi, W. (2013). The Subread aligner: Fast, accurate and scalable read mapping by seed-and-vote. *Nucleic Acids Research*, *41*(10), e108.

Liao, Y., Smyth, G. K., & Shi, W. (2014). Featurecounts: An efficient general-purpose program for assigning sequence reads to genomic features. *Bioinformatics*, *30*(7), 923–930. doi: 10.1093/bioinformatics/btt656.

Love, M. I., Huber, W., & Anders, S. (2014). Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biology*, *15*, 550. doi: 10.1186/s13059-014-0550-8.

Martin, M. (2011). Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnet.Journal*, *17*(1), 10–12.doi: 10.14806/ej.17.1.200.

Melville, S. B., & Gunsalus, R. P. (1996). Isolation of an oxygen-sensitive FNR protein of

Escherichia coli: Interaction at activator and repressor sites of FNR-controlled genes. *Proceedings of the National Academy of Sciences of the United States of America*, *93*(3), 1226–1231. doi: 10.1073/pnas.93.3.1226.

Myers, K. S., Yan, H., Ong, I. M., Chung, D., Liang, K., Tran, F., ... Kiley, P. J. (2013). Genome-scale analysis of *Escherichia coli* FNR reveals complex features of transcription factor binding. *PLoS Genetics*, *9*(6), e1003565. doi: 10.1371/journal.pgen.1003565.

Nguyen, N. T. T., Contreras-Moreira, B., Castro-Mondragon, J. A., Santana-Garcia, W., Ossio, R., Robles-Espinoza, C. D., ... Thomas-Chollier, M. (2018). RSAT 2018: Regulatory sequence analysis tools 20th anniversary. *Nucleic Acids Research*, *46*(W1), W209–W214. doi: 10.1093/nar/gky317.

Pepke, S., Wold, B., & Mortazavi, A. (2009). Computation for ChIP-seq and RNA-seq studies. *Nature Methods*, *6*, S22–S32. doi: 10.1038/nmeth.1371.

Pérez-Rueda, E., & Collado-Vides, J. (2000). The repertoire of DNA-binding transcriptional regulators in Escherichia coli K-12. *Nucleic Acids Research*, *28*(8), 1838–1847. doi: 10.1093/nar/28.8.1838.

Robertson, G., Hirst, M., Bainbridge, M., Bilenky, M., Zhao, Y., Zeng, T., ... Jones, S. (2007). Genome-wide profiles of STAT1 DNA association using chromatin immunoprecipitation and massively parallel sequencing. *Nature Methods*, *4*(8), 651–657. doi: 10.1038/nmeth 1068.

Robinson, J. T., Thorvaldsdóttir, H., Winckler, W., Guttman, M., Lander, E. S., Getz, G., & Mesirov, J. P. (2011). Integrative genomics viewer. *Nature Biotechnology*, *29*(1), 24–26. doi: 10.1038/nbt.1754.

Robinson, M. D., McCarthy, D. J., & Smyth, G. K. (2010). edgeR: A Bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics*, *26*(1), 139–140. doi: 10.1093/bioinformatics/btp616.

Salmon, K., Hung, S., Mekjian, K., Baldi, P., Hatfield, G. W., & Gunsalus, R. P. (2003). Global gene expression profiling in *Escherichia coli* K12: The effect of oxygen availability and FNR. *Journal of Biological Chemistry*, *278*(32), 29837–29855. doi: 10.1074/jbc.M213060200.

Schuster, S. C. (2007). Next-generation sequencing transforms today's biology. *Nature Methods*, *5*(1), 16–18. doi: 10.1038/nmeth1156.

Schurch, N. J., Schofield, P., Gierliński, M., Cole, C., Sherstnev, A., Singh, V., ... Barton, G. J. (2016). How many biological replicates are needed in an RNA-seq experiment and which differential expression tool should you use? *RNA*, *22*(6), 839–851. doi: 10.1261/rna. 053959.115.

Thomas-Chollier, M., Herrmann, C., Defrance, M., Sand, O., Thieffry, D., & Van Helden, J. (2012). RSAT peak-motifs: Motif analysis in full-size ChIP-seq datasets. *Nucleic Acids Research*, *40*(4), e31. doi: 10.1093/nar/gkr11 04.

Thorvaldsdóttir, H., Robinson, J. T., & Mesirov, J. P. (2013). Integrative Genomics Viewer (IGV): High-performance genomics data visualization and exploration. *Briefings in Bioinformatics*, *14*(2), 178–192. doi: 10.1093/bib/bbs017.

Tsagmo Ngoune, J. M., Njiokou, F., Loriod, B., Kame-Ngasse, G., Fernandez-Nunez, N., Rioualen, C., ... Geiger, A., (2017). Transcriptional profiling of midguts prepared from *Trypanosoma/T. congolense*-positive *Glossina palpalis palpalis* collected from two distinct Cameroonian foci: Coordinated signatures of the midguts. remodeling as *T. congolense*-supportive niches. *Frontiers in Immunology*, *8*, 876. doi: 10.3389/fimmu.2017.00876.

van Helden, J., Ríos, A. F., & Collado-Vides, J. (2000). Discovering regulatory elements in non-coding sequences by analysis of spaced dyads. *Nucleic Acids Research*, *28*(8), 1808–1818 doi: 10.1093/nar/28.8.1808.

Wilkinson, M. D., Dumontier, M., Aalbersberg, Ij. J., Appleton, G., Axton, M., Baak, A., ... Mons, B. (2016). The FAIR Guiding Principles for scientific data management and stewardship [Comments and Opinion]. Retrieved December 1, 2017, from https://www.nature.com/articles/sdata20161.

Key References

Gama-Castro et al. (2016). See above.
Describes RegulonDB version 9

Köster & Rahmann (2012). See above.
Describes the Snakemake workflow engine.

Myers et al. (2013). See above.

Publication associated to the dataset used in this protocols.

Internet Resources

Snakemake: Retrieved from http://snakemake.readthedocs.io

SnakeChunks GitHub repository: Retrieved from https://github.com/SnakeChunks/SnakeChunks

SnakeChunks documentation & tutorials: Retrieved from http://snakechunks.readthedocs.io

FastQC: Retrieved from http://www.bioinformatics.babraham.ac.uk/projects/fastqc/

UCSC file format description: Retrieved from https://genome.ucsc.edu/FAQ/FAQformat.html