Norwegian University
of Life Sciences

Master's Thesis 2023 30 ECTS
Faculty of Chemistry, Biotechnology and Food Science

CutAndTag-Analyzer: A New Python Package for CUT&Tag Data Analysis

Jenny Sofie Dragland
Chemistry and Biotechnology – Bioinformatics

Abstract

Ever since the invention of next-generation sequencing, new methods for understanding gene expression and control through epigenetics marks have been created. Among these, CUT&Tag analysis has emerged to become an efficient epigenomic profiling technique with low input requirements, high sensitivity, and lower background signals. Even though the wet-lab techniques are in place, analyzing the data is still a challenge for scientists with less computational skills, such as biologists. Therefore, this master’s thesis presents a new Python package that not only simplifies the data analysis of CUT&Tag sequencing but also allows biomedical scientists to easily interpret the results. The new pipeline package is based on the original CUT&Tag innovation team´s data analysis protocol. It includes every step necessary from quality control to annotation and differential peak analysis. The package also fixed a few bugs, (e.g., reproducibility assessment) from the original protocol, and added new visualization plots, and features like genome annotation and differential peak analysis. In a demonstration run on a real CUT&Tag data set, the new package successfully recreated the plots from the original study. Additionally, function annotation analysis on annotated genes revealed predictions supporting current literature on the target proteins.

Sammendrag

Helt siden oppfinnelsen av neste generasjons sekvensering har nye metoder for å forstå genuttrykk og kontroll gjennom epigenetiske merker blitt langet. Blant disse har Cut&Tag-analyse vist seg å bli en effektiv epigenetisk profileringsteknikk med høy sensitivitet, lave bakgrunnsverdier, som er kompatibel med små prøvemengder. Selv om våtlabstekknikene er på plass, kan analyse av CUT&Tag data skape problemer for forskere med mindre programmeringserfaringer. Derfor presenterer denne masteroppgaven en ny Python-pakke som forenkler datanalysen av CUT&Tag-data, noe som resulterer i raskere og enklere analyse, slik at flere forskere kan utnytte teknikken. Pipeline-pakken er basert på det originale CUT&Tag-innovasjonsteamests datanalyseprotokoll og inkluderer alle nødvendige trinn fra kvalitetskontroll til annotering og differensialanalyse. Under en testkjøring gjenskapte den nyopprettede pipelinen plottene fra den opprinnelige protokollen, fikset en feil i reprodduserbarhetsvurdering og opprettet nye plot samt filer fra tilleggs funksjonen; annotering og differensialanalyse, som ikke var en del av den opprinnelige protokollen. I tillegg viste funksjonsannoterings analyse av annoterte gener, at prediksjonene var i samsvar med gjeldende litteratur om histon modifikasjonene som var målene i eksperimentet.

Acknowledgments

I want to thank my supervisor Junbai Wang at the Faculty of Medicine at Oslo University, for guiding me through the whole process of creating my first Python package, always answering my questions, and motivating me to pursue computational biology. I would also like to thank Torgeir Rhoden Hvidsten for reading through my thesis and giving helpful advice. Additionally, I would like to thank my family, friends, and especially my partner, Joakim, for supporting and cheering me on.

Table of Contents

1. Introduction

It was long thought that DNA was a set recipe for how an organism was built, functioned, and behaved. Additionally, it was believed that only mutations to the nucleotide sequence itself could lead to offspring different from the parent. It would be excellent if this were a simple truth; however, it turns out not to be the case [1]. Epigenetics could be described as the study of heritable gene expression or cellular phenotype caused by underlying mechanisms other than changes in the DNA sequence. These changes could be attributed to environmental factors, such as diet or stress, and may be passed down for generations [1]. Epigenetics is crucial in understanding the mechanisms behind gene expression, cellular differentiation, and organism and disease development. The world of epigenetics tells us how a skin cell and a liver cell can be so drastically different even though they share the same genetic martial, or what are the mechanisms behind the differential gene expression between healthy and cancer cells [1, 2]. Some crucial aspects of gene regulation are the binding of transcription factors, and the presence of histone modifications, which control gene expression through sequence-specific targets, or changes in DNA packing, respectively [3].

Since the beginning of epigenetics in 1939 (Waddington), several methods for profiling epigenetic marks and transcription factor binding events have been developed [1]. Among them, CUT&Tag (Cleavage Under Targets and Tagmentation) has emerged as a powerful and efficient method from 2019 [4], which is an innovative and efficient epigenetic profiling method [4]. It offers several advantages over traditional methods like ChIP- seq (Chromatin Immunoprecipitation sequencing) and CUT&RUN (Cleavage Under Targets and Release Under Nuclease): for example, shorter lab time, lower input requirements, high sensitivity, and lower background signals, etc. [5]. Even though there are many benefits of Cut&Tag, some challenges remain in analyzing and interpreting the data, especially for biologists or researchers without a bioinformatics background.

The aim of this thesis is to develop a user-friendly, Python-based data analysis pipeline to simplify the analysis of CUT&Tag sequencing data. It includes all essential data analysis steps, from preprocessing raw sequencing reads to providing biological interpretation of results such as quality control, alignment, peak calling, and differential binding analysis. Since the pipeline includes both visualization and genome annotation of results, it may become a useful tool for scientists with limited informatics skills to analyze CUT&Tag data.

To illustrate the development of this new bioinformatical tool, the master´s thesis will include the following parts: 1. A literature review on Cut&Tag and similar methods and how the data has been analyzed up to this point. 2. An overview of bioinformatical tools that might be used for the same purpose. 3. A workflow for designing and implementing the pipeline in Python. 4. An evaluation of the pipeline by publicly available CUT&Tag data.

Overall, the master´s thesis contributes a new tool to the field of epigenetics, which helps scientists harness the power of CUT&Tag sequencing, as well as improve understanding of the complex interactions between gene regulation, histone modifications, and transcription factor binding. Ultimately, understanding biological processes may help fight illnesses associated with epigenetic dysfunction.

2. Background

2.1 Epigenetics and Gene Regulation

Epigenetics, or “outside regular genetics”, plays a vital role in gene regulation [6], and refers to observable changes in DNA packing protein or DNA itself. While the DNA strands form the foundation of which genes are present in a cell, the regulatory factors such as transcription factors and “reader” proteins – chromatin effector proteins control gene regulation. These factors may recognize specific marks in the DNA with cytosine methylation or modifications on the DNA-protein complexes, such as nucleosomes or chromatin [7, 8]. In this master’s thesis, the focus will be on DNA-protein complexes and the methods that are used to profile DNA-binding proteins, and the modifications that may be present.

2.2 Histone Modifications

Modifications on the histones (e.g., methylation, acetylation. phosphorylation, ubiquitylation, cirullinatin, etc.) present in nucleosomes can impact gene regulation by reducing or increasing the availability of the DNA strands for transcription factors and acting as markers for “reader” proteins [9]. Nucleosomes, commonly described as peals on the DNA strands, are the fundamental subunit of chromatin. The nucleosome consists of two copies of each of the four histone proteins: H2A, H2B, H3, and H4, making the protein an octamer. For the condensation of DNA, approximately 150 base pairs (bp) are wrapped around the histone octamer. These histone proteins have polypeptide sequences, often called tails, extending out. Figure 2.1 shows the histone subunits, with these polypeptide tales, and the most common chemical groups or modifications added to the amino acid residues. Modifications (e.g., methylation or phosphorylation) of the amino acids present in these tails may affect the interaction between DNA and the histone octamer. Table 2.1 shows the most common histone mortifications, their genomic location, and their functions in biological processes. These modifications can cause the chromatin to be highly condensed in heterochromatin or loosely packed as euchromatin. Additionally, these modifications are marking points for chromatin effector molecules or “readers”, which recruit other molecules to alter the chromatin and influence the transcription state [10]. Thus, these modifications of the histone tales can alter gene expression.

Histone modification	**Function**	**Location**
H3K4me1	Activation	Enhancers
H3K4me3	Activation	Transcription start sites, promoters, bivalent domains
H3K36me3	Activation	Gene bodies
H3K79me2	Activation	Gene bodies
H3K9Ac	Activation	Enhancers, promoters
H3K27Ac	Activation	Enhancers, promoters
H3K27me3	Repression	Promoters in gene-rich regions, poised enhancers, bivalent domains
H3K9me3	Repression	Satellite repeats, telomers, pericentromeres
H3S10P	DNA replication	Mitotic chromosomes

Table 2.1 *- shows the most common histone modifications found in the genome and their function. [11]*

Acetylation is one of the most studied histone modifications that may result in gene activation. Lysine residues usually are positively charged in water, resulting in significant interactions between the amino acid and the negatively charged DNA. However, acetylation of lysine neutralizes it, weakening the DNA/peptide interactions and making the DNA more available for the transcription machinery [12]. Hence, analyzing acetylation marks can help in the prediction of open and active DNA.

Histone phosphorylation is important in chromosome condensation during DNA repair, cell division, and transcriptional regulation. It transpires on all core histones, yet, with differential effect [13, 14]. For example, the phosphorylation of histone H3 on serine 10 and 28 and T120 on histone H2A are important cell cycle markers, which regulate chromatin structure during mitoses. They are highly conserved in eukaryotic cells and crucial in the structural regulation of DNA packing.

Ubiquitylation occurs most commonly at H2A and H2B, two of the most highly ubiquitylated proteins in the nucleus. Nevertheless, all histone core proteins may be ubiquitylated. Moreover, ubiquitylation is imperative in DNA damage response as monoubiquitylation of H2A, H2B, and H2AX is found at DNA double-strand break sites [15]. Ubiquitylation of histone H2A is essential for embryonic stem cells to differentiate into embryonic bodies [16]. Thus, the reversible addition of ubiquitin to histones allows cells and organisms to control gene expression and respond to trauma-caused DNA breakage.

Methylation may occur on all basic amino acid residues, resulting in different impacts on transcription, which may affect the development and cellular response. Arginine methylation on histone H4 is found to promote transcriptional activation. Since methylation does not alter the charge of the lysine residue, its consequences depend on the location and the level of methylation. Lysine residues can be mono-, di- or tri-methylated, providing further diversity in gene regulation and marking: for example, the mono- and tri-methylation of K4 of histone H4, (H4K4Me1 and H4K4Me3) contribute to gene activation. Nevertheless, these modifications are found in different genomic regions (e.g., H4K4Me1 is a mark of gene promotors, and H4K4Me3 is associated with enhancers). On the contrary, tri-methylation on K9 and K27 of histone H3 are repressive signals [10]. CUT&Tag or similar methods can measure these histone modifications.

Overall, there is a big variance in histone modifications that add a layer of complexity to genome regulation. Their dysfunctions may lead to errors associated with diseases such as cancer, heart failure, autoimmune diseases, and neurodegenerative diseases like Huntington's, Parkinson´s, and Alzheimer’s disease [17]. Additionally, histone modifications are very important during organism development and stem cell differentiation [16]. A deeper understanding of histone modifications in illnesses may help research groups make novel therapies for targeting specific proteins associated with histone modifications [2]. Thus, histone modifications are essential for improving the understanding of the gene expression machinery that may lead to disease treatments.

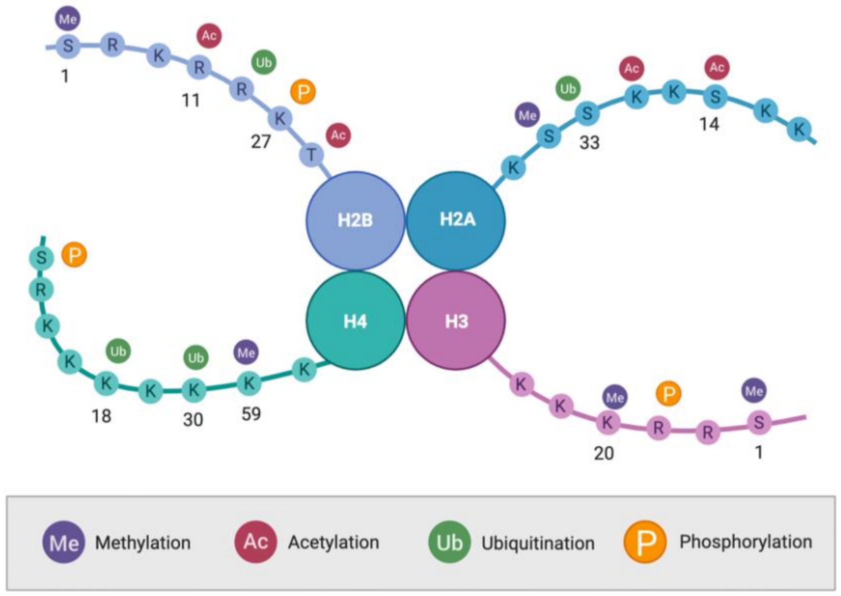


Figure 2.1 *- Figure of a nucleosome, its histone subunits, their histone tails, and common modifications; methylation, acetylation, ubiquitylation, and phosphorylation. Figure made with Biorender.com.*

2.3 Transcription Factor Binding

Transcription factors (TFs) are proteins that read and recognize specific DNA sequences to control chromatin and transcription and regulate gene expression. Mutations in the protein itself or the DNA sequence (e.g., transcription factor binding sites, TFBSs) cause many illnesses. TFBSs, are located at cis-regulatory regions, such as promoters and enhancers. TFs can be categorized into structural classes based on their DNA binding domains (DBDs) that directly interact with DNA, where TFs in the same classes of DBDs have similar DNA binding preferences. TFs control gene expression in multiple ways; for example, some TFs have an inhibitory effect when binding to enhancer regions and downregulating expression; TFs may recruit the RNA polymerase that initiates RNA transcription, mediate interactions with other proteins, or have enzymatic functions themselves, see Figure 2.2 [8].

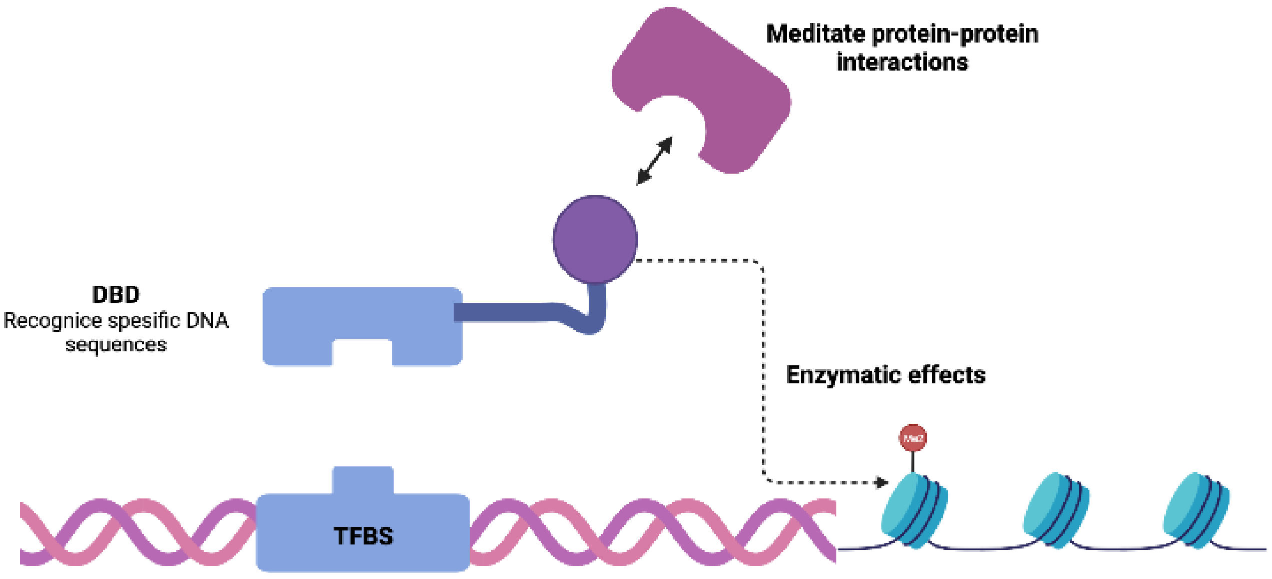

Meditate protein-protein interactions
DBD
Recognice spesific DNA sequences
Enzymatic effects
TFBS

Figure 2.2 *– Simplified transcription factor (TF) binding and effects at enhancer site. The DNA binding domains (DBD), of the TF, will recognize a transcription factor binding site (TFBS) in the DNA. Consequently, the TF can meditate protein-protein interactions or have enzymatic effects. Figure made with Biorender.com.*

TF-DNA interactions depend on multiple factors, TFs being present in the district subcellular regions, cooperation between other DNA-binding proteins, and the chromatin environment. To fully understand the regulation of TFs, it is paramount to gain detailed knowledge about the DNA sequences TFs bind. [8] Thus, there are projects like JASPER and ENCODE curating the location and functionality of TFs and their motifs [3, 18]. Currently, there are several methods for detecting the TF binding sites, three of which will be mentioned later in this thesis: ChIP-seq, CUT&RUN, and CUT&Tag [4, 19, 20]. Altogether, the application of these approaches can have an immense influence on the understanding of the complex gene regulation network.

2.4 Techniques for Profiling Epigenetic Marks and Transcription Factor Binding

Currently, there are several techniques for profiling epigenetic marks (or histone modification) and transcription factor binding, each with its pros and cons. Chromatin Immunoprecipitation followed by Sequencing (ChIP-seq) and Cleavage Under Targets and Release Using Nuclease (CUT&RUN) are the most used methods for studying the chromatin landscape [19, 21].

2.4.2 ChIP-seq

ChIP-seq is the gold standard for profiling epigenetic marks and transcription factor binding. Chromatin immunoprecipitation has been used to profile DNA-binding proteins since the 1980s [22]. With the rise of efficient and accurate sequencing techniques, ChIP-seq was born in 2007 after combining chromatin immunoprecipitation with next-generation sequencing [23]. Massive projects (e.g., ENCODE) leveraged ChIP-seq in the quest to profile the epigenomic profiles in the human genome, which made the method more popular, and more ChIP-seq experimental data become available online [24].

The basics behind ChIP-seq are: 1. Fragmentation; 2. Binding specific antibodies to the desired target, such as histone modification; 3. DNA purification, PCR, and next-generation sequencing, as seen in Figure 2.3. DNA fragmentation can be done in two different ways. For detecting DNA-binding proteins genome-widely, a formaldehyde treatment to cross-link the protein to the DNA, followed by sonic fragmentation, is the standard. For measuring histone modification, micrococcal nuclease (MNase) digestion without cross-linking is preferred. The fragments created are generally 200-600 bp long in both methods. Then, antibodies specific to the target are added to immunoprecipitate the desired mark. Next, the DNA is purified, only leaving what was formally immunoprecipitated DNA. Lastly, the DNA is prepped for next-generation sequencing, often including PCR, however, this may vary depending on the method used. Finally, library preparations involve size selections typically 150-300 bp. Though multiple sequencing platforms may be used, most of ChIP-seq data sets have been produced by the Illumina genome sequencer [20].

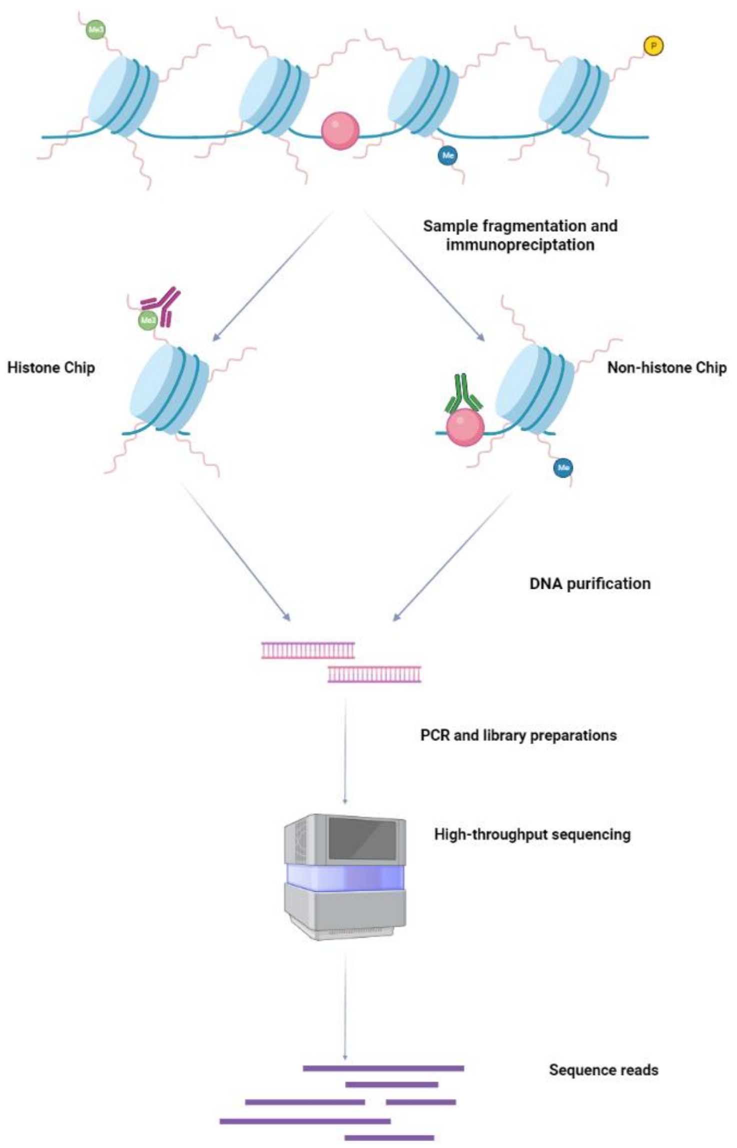

Me3
P
Me
Sample fragmentation and immunopreciptation
Histone Chip
Non-histone Chip
DNA purification
PCR and library preparations
High-throughput sequencing
Sequence reads

Figure 2.3 *– General and simplified flowthrough of Chip-seq analysis for epigenetic profiling, starting with fragmentation either done by sonar or MNase, followed by immunoprecipitation with antibodies specific for the target, either a histone modification or non-histone DNA-binding protein. Lastly, the DNA is purified, copied with PCR, and sequenced. Figure made with Biorender.com.*

Though ChIP-seq may be the gold standard method for profiling epigenetic marks and transcription factor binding, a limitation regarding sample size causes large sample numbers (10^5 –10^7 cells) to be needed to compensate for the loss in the immunoprecipitation process [25]. This is caused by possible deficiency of the antibody proteins used for immunoprecipitation in ChIP-seq experiments. Firstly, the antibodies may have poor reactivity to the intended target, leading to low sequencing depth. Secondly, cross-reactivity with other DNA-associated proteins is also a concern when using antibodies. Therefore, false positives, genomic locations not associated with the intended target, may appear in the sequencing data [26]. Moreover, cross-linking with formaldehyde may mask antigenic epitopes of transcription factors. On the other hand, using MNase to fragment the genome may lead to the loss of target protein binding sites where the target does not bind strongly to DNA. Therefore, MNase fragmentation is generally used for measuring histone modifications. Additionally, the PCR step may introduce potential bias and length limitation [25]. Though there are methods to combat these sources of error, they lead to high costs and are time-consuming compared to other techniques. In conclusion, ChIP-seq is a widely used method for profiling the chromatin landscape, a substantial amount of data generated from ChIP-seq is publicly available [26]. However, its pitfalls call for newer and more efficient methods for genome-wide chromatin profiling.

2.4.3 CUT&RUN

CUT&RUN has emerged as a successful method for identifying the genetic locations of DNA-binding proteins in high resolution. The method was created as a higher resolution, higher signal-to-noise, and lower cost alternative to ChIP-seq [27]. Though CUT&RUN uses immunoprecipitation via antibodies to find desired targets in the genome, its fragmentation method has significantly higher specificity than that of Chip-seq Particularly, CUT&RUN utilizes a recombinant proteinA-microccocal nuclease (pA-MNase) that specifically digests DNA surrounding the target protein. It results in a higher signal-to-noise ratio because of the specific/non-randomized fragmentation [19]. Additionally, it allows for the detection of signals from smaller sample sizes, which is a great advantage over other methods when samples are limited [27].

The CUT&RUN process can be summarized in the following steps in Figure 2.4. Firstly, the nuclei are isolated and immobilized, usually using magnetic beads. Secondly, specific antibodies are added, diffuse into the nuclei, and bind to the desired objective. Furthermore, pA-MNase is added; here, the protein A part of the recombined protein will mediate protein-protein interaction with the already bound antibodies. To activate the enzymatic effects of MNase, Ca^{2+} is added.

Subsequently, the MNase digests the DNA strands surrounding the desired target. Finally, the DNA is purified, prepped for sequencing, and sequenced using high-throughput sequencing [27].

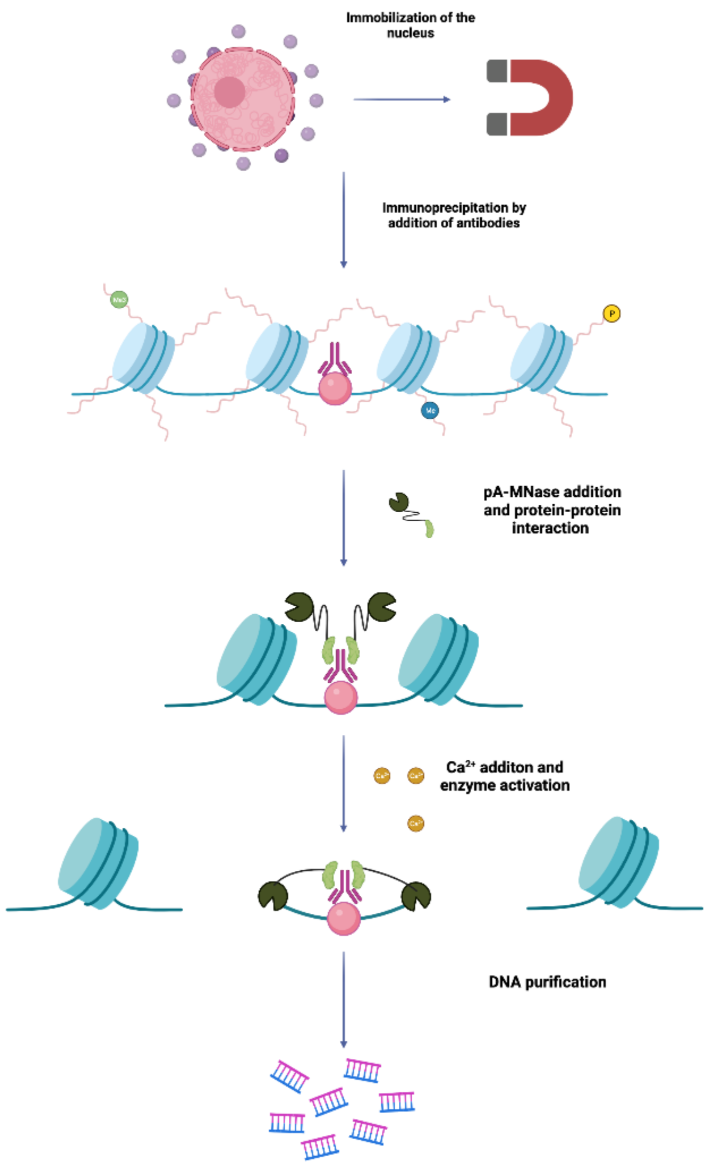

Immobilization of the nucleus
Immunoprecipitation by addition of antibodies
pA-MNase addition and protein-protein interaction
Ca²⁺ additon and enzyme activation
DNA purification

Figure 2.4 *– General and simplified flowthrough of Cut&Run analysis for epigenetic profiling, starting with immobilization of nuclei, following immunoprecipitation of desired target by antibody binding. Next, proteinA-microccocal nuclease (pA-MNase) is added, adheres to the bound antibody, and when exposed to Ca^{2+}, digests the DNA sounding the desired target. Finally, the DNA is purified and ready for amplification, library prep and sequencing. Figure made with biorender.com.*

Though CUT&RUN is a powerful, accurate, and high-resolution method for gaining knowledge about the epigenetic landscape, it has a complex procedure, and the experimental protocol may hinder new users (e.g., some protocols have close to eighty steps). [27]. Furthermore, the lab work may take several days to perform, creating greater costs [19]. Thus, a more efficient method, with the same accuracy as CUT&RUN, is needed to investigate epigenetic marks and transcription factor binding.

2.5 CUT&Tag Analysis

CUT&Tag was developed by the same laboratory team that started CUT&RUN and allows lower starting material and a more efficient procedure. Like CUT&RUN and ChIP-seq, CUT&Tag uses antibodies to immunoprecipitate targeted DNA-binding proteins. However, CUT&Tag includes tagmentation of the desired sequences in the fragmentation step, resulting in higher accuracy. Consequently, the required sequencing depth can be reduced, allowing for a smaller starting sample. Thus, CUT&Tag is an excellent method for single-cell experiments that allows the exploration of tissue complexity, the in-depth analysis of gene regulation, the imputation of developmental trajectories, and the prediction of further cell states [4].

The CUT&Tag experimental protocol starts with the immobilization of the nucleus. Next, antibodies specific to the target protein are added for immunoprecipitation. Then a secondary antibody is added and binds to the target protein-antibody complex. This helps to amplify the signal. Fragmentation and tagmentation are done in one step by the hyperactive fusion protein Tn5 transposase-proteinA (pA-Tn5), which is preloaded with sequencing adapters. This recombinant protein binds to the secondary antibody, and after enzymatic activation by Mg^{2+} addition, the fusion protein digests the DNA sequence around antibodies bound to the desired target proteins. In the same process, pA-Tn5 ligases the sequencing adapters to the fragmented DNA, resulting in already tagged DNA fragments. This eliminates the need for library preparation, saving laboratory work and time. Thus, CUT&Tag is a highly efficient and cost-effective method for profiling epigenetic marks and DNA-binding proteins. It can generate significant results with small starting materials [28].

CUT&Tag experiment can be performed in 1-2 days, depending on the number of samples and laboratory experience of the user [29]. The experimental procedure can be summarized in the following steps, shown in Figure 2.5. As with CUT&RUN, the cells are immobilized and permeabilized, adding target-specific antibodies. After incubation, a second antibody is added that binds to the first one. Then, the addition of pA-Tn5 and Mg^{2+} causes fragmentation and tagmentation of the DNA sequence. Finally, the tagged fragments are upscaled with PCR and ready for high-throughput sequencing [28].

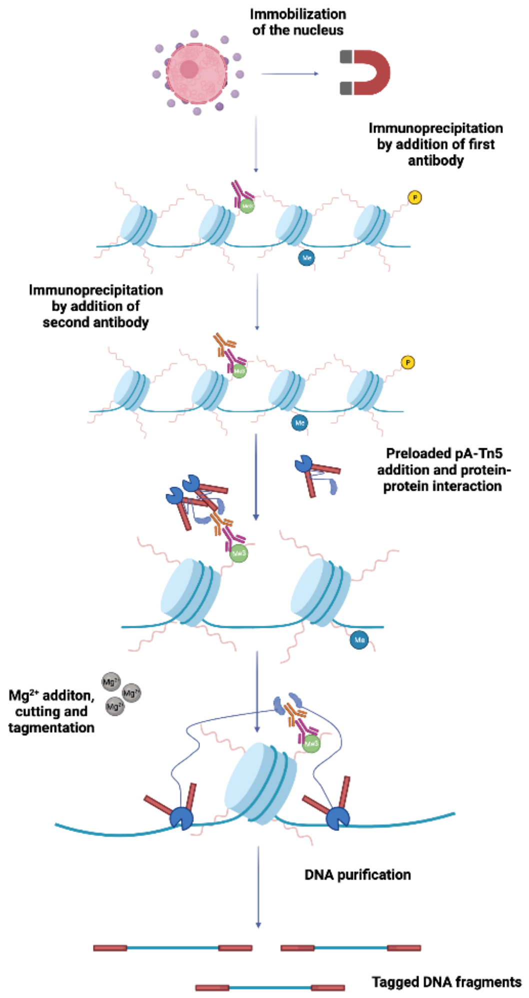

Immobilization of the nucleus
Immunoprecipitation by addition of first antibody
Immunoprecipitation by addition of second antibody
Preloaded pA-Tn5 addition and protein-protein interaction
Mg²⁺ additon, cutting and tagmentation
DNA purification
Tagged DNA fragments

Figure 2.5 *- General and simplified flowthrough of CUT&Tag analysis for epigenetic profiling, starting with immobilization of nuclei, following immunoprecipitation of desired target by secondary antibody binding. Next, Tn5 transposase-proteinA (pA-Tn5) preloaded with tags is added, adheres to the bound antibody, and when exposed to* Mg^{2+}*, digests and tags the DNA sounding the desired target. Finally, the DNA is purified and ready for amplification and sequencing. Figure made with biorender.com.*

CUT&Tag is a powerful tool to measure epigenetic marks, but it has some limitations in detecting weakly bound proteins. For example, the CUT&Tag protocol does not involve cross-linkage, making it more efficient than Chip-seq [28]. However, without cross-linkage, weakly bound proteins may detach from the DNA, causing lower signal densities [29]. On the other hand, CUT&Tag can detect strong interactions between transcription factors or other proteins and DNA sequences [4]. Thus, it is a highly accurate, low-cost, and efficient single-cell compatible genome-wide chromatin profiling technique.

2.6 Low-level and High-level Analysis in High-throughput Sequencing Experiments

The sequencing data from ChIP-seq, CUT&RUN, and CUT&Tag contain important information about protein binding sites and chromatin modifications related to gene regulation [4] [19, 20]. These methods have specific considerations when analyzing the data; however, the general flowthrough can be seen in Figure 2.6.

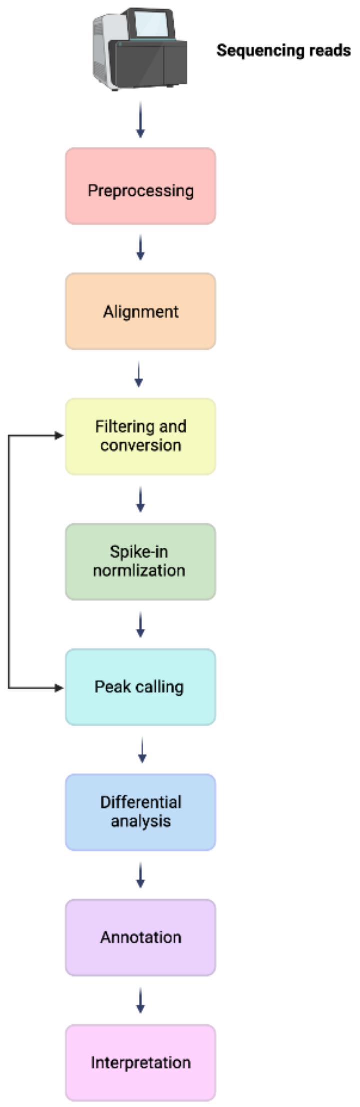


Figure 2.6 – *Simplified and generalized pipeline used for Chip-seq, Cut&Run, and Cut&Tag data analysis. Figure made with biorender.com*

2.6.1 A Low-level Sequencing Data Analysis – preprocessing of raw reads

Often the preprocessing of raw reads (e.g., quality control, sequencing alignment, or filtering of raw reads) is similar among various high-throughput experiments [21]. Quality control of raw reads is essential for downstream analysis because, without a clear confirmation of the experiment´s quality, the results or hypothesis generated from the study will not be scientifically sound (garbage in = garbage out). After preprocessing the raw sequencing reads (quality control and filtering), they are aligned to a reference genome. The alignment step maps reads to a known reference genome, which may involve the filtering of poorly mapped reads. Here, some statistical criteria (e.g., uniquely mapped reads, sequencing depth, and alignment rate) are used to assess sequencing quality: for example, a low number of uniquely mapped fragments may indicate excess amplification in the PCR step, inadequate read length or complication in the sequencing step [21].

Removal of duplicated reads is only recommended for experiments with minimal material or if PCR duplication is suspected. In CUT&Tag experiments, some of the “apparent” duplicated fragments/reads are likely to be true fragments: for example, CUT&Tag adapters are inserted into the DNA, and the exact sites of integration are affected by the accessibility of surrounding DNA. Therefore, fragments with the same starting and ending positions are expected to be common. Thus, duplication removal should be performed with caution. Nevertheless, Cut&Tag is a method that yields results even with minimal starting materials, where duplication removal might be helpful.

2.6.2 A low-level Sequencing Data Analysis - Spike-in Calibration

Spike-in calibration is a normalization method that adjusts relative abundance based on control abundance. Usually, some form of bacterial DNA is added to the samples throughout the experimental protocol since microorganisms are used to produce recombinant proteins. After sequencing the samples, alignment to a spike-in genome may be performed. Usually, the selected spike-in genome is from a bacteria (e.g., *Escherichia coli E. coli*). A spike-in sequencing depth may be found from the spike-in alignment. This spike-in sequence depth is used to scale the sequencing depth from normal reference genome alignment, therefore, adjusting the relative abundance. Moreover, global signal normalization should be avoided if samples have signal variations due to condition differences, as it may falsely introduce signal variations in regions without changes. Therefore, a spike-in normalization should be applied if the total amount of signal is expected to change between conditions, e.g., sick/healthy cells [30]. Nevertheless, spike-in normalization is an optional step in Cut&Tag data analysis.

2.6.3 A Low-level Sequencing Data Analysis - Blacklist Removal

Removing blacklisted regions from a reference gnome is an additional filtering or quality control of the sequencing data since some parts of the genome may cause difficulty in genome annotation, experimental measurement, or sequencing alignment. For example, mapping reads to repetitive regions is challenging and may lead to the amplification of background noise if not removed. [31]. Therefore, removing blacklisted regions based on known information (e.g., ENCODE) will increase accuracy in epigenetic profiling in high-throughput sequencing.

2.6.4 A Low-level Sequencing Data Analysis -Peak Calling

Identifying genomic regions where signals are enriched compared to background noise (e.g., non-targeting antibody or control DNA input) is essential for downstream analysis of epigenomic marks. This is often called "peak calling," as the enriched regions will have "peaks", a high number of reads in the target binding area. There are different methods of peak calling for ChIP-seq, CUT&RUN, and CUT&Tag. For instance, a peak caller for ChiP-seq data utilizes highly sensitive models optimized for high recall to differentiate signal from noise. [32]. For the other experiments (CUT&RUN, CUT&Tag) with lower background noise and lower sequencing depths [4, 19], the peak calling method requires high specificity rather than high sensitivity, as highly sensitive methods may result in false positives – background noise being included in peaks [32]. Thus, peak calling is a critical step for identifying enriched genomic regions in sequencing experiments.

2.6.5 A High-level Sequencing Data Analysis - Differential Analysis and Annotation

Differential analysis of multiple samples with various proteins as targets will reveal peak signals with statistically significant changes between them [33]. Since epigenetic marks are dynamic, many studies aim to find epigenetic differences between samples /conditions, either between sick and healthy cells or between various cell types [2, 6]. Advanced statistical methods are needed to perform differential analysis based on sequencing results from various experiments. A good understanding of where a specific epigenetic mark is located or transcription factor is bound in the genome will give insights into the overall chromatin landscape and gene regulation [6]. Therefore, differential analysis and the corresponding genomic annotation of these changes (e.g., mapping significantly changed peaks to promotors, enhancers, or genes, et.al) is a good way to gain knowledge about gene regulation. Thus, this high-level sequencing data analysis step is often performed after obtaining results from ChIP-seq, CUT&RUN, or CUT&Tag experiments [34].

2.6.6 Interpretation

After obtaining meaningful results from sequencing experiments, such as signals, peaks, or dynamic changes of peaks, there is a need to interpret and explore the hidden knowledge behind them. This is another high-level sequencing data analysis essential for gaining knowledge from a sequencing experiment. For example, comparing results with similar studies or other molecular measurements, such as RNA sequencing of gene expression profiles, searching for the biological truth of the observations, and creating meaningful plots, may aid in interpreting the results. Heatmaps of the selected results or data mining methods (e.g., clustering) are often applied for such interpretation [4, 5, 19]. A good visualization of the results may help scientists squire or understand new research concepts. At this stage of high-level sequencing analysis, more data mining and machine learning methods may be used to accomplish scientific goals. In conclusion, the interpretation involves putting the results in a context with previous knowledge, and visualization of the results is beneficial assistance.

2.7 Existing Tools for Cut&Tag Data Analysis

Analyzing Cut&Tag data may reveal crucial information about epigenetic marks [4]. Currently, several tools are available which may be used to complete the analysis of CUT&Tag data. From low-level to high-level analysis, it includes: FastQC for examining raw reads quality [35], bowtie2 for sequencing alignment [36], SEACR for peak calling [32], and deepTools for visualization and plots, etc. [37]. Both statistical and programming expertise is necessary to perform the data analysis from scratch. Though there is a tutorial for complete CUT&Tag data analysis available online, many biologists or wet lab scientists do not have the skillset (e.g., working in Linux/Unix environments, making necessary scripts for analyzing the data, etc.) to do it by themselves. [38] Thus, there is a need for a user-friendly data analysis package, such as Cut&RunTools [39], to simplify the CUT&Tag data analysis and help researchers to interpret/visualize results from the sequencing experiments.

2.8 Statistical methods

Statistical methods are often used in data analysis. In this Chapter, a few of them used in CUT&Tag data analysis, such as normalization and hypothesis testing, will be introduced.

2.8.1 Quantile normalization

Quantile normalization is a statistical normalization method for making two or more distributions identical in statistical properties. It can be used to normalize the raw signals from sequencing measurements and assumes an equal distribution of detected signals across all samples or conditions. Quantile normalization can be achieved by normalizing arrays to each other. To accomplish quantile normalization among samples, the arrays are first independently sorted (e.g., in ascending order). Now, the mean at each entry/row may be found such that the biggest value of all arrays has a mean, the second biggest entry has a mean, and so on. This creates an array of means that is used to replace every entry in the original set at the corresponding entries so that every array has the exact ordering as the origin. In short, the collection of arrays is transformed to have the same distribution of values [40]. In this thesis, sets of arrays will be saved in matrixes therefore the algorithm will consist of the following steps given a matrix M of size $n \times k$, where each column is an array*:*

1. Sort each column in M to make M_{sorted}
2. Find the means of across each row in M_{sorted} and store this mean to each element at each row to get M'_{sorted}, the mean being the arithmetic mean, defined by the formula:

$$X = \frac{1}{n}\sum_{i=1}^{n} x_i = \frac{x_1 + x_2 + \cdots x_n}{n}$$

3. Create $M_{normalized}$ by rearranging each column of M'_{sorted} to have the same ordering as the original matrix M

2.8.2 Principal component analysis

Principal component analysis, or PCA, is a linear dimensionality reduction method used to reduce data dimensionality while preserving essential features. It is often applied to high-dimensional data sets in bioinformatics, making it easier to visualize and explore key features of big data. Though there are multiple methods to reduce data dimensions, PCA is a simple and efficient method to project original data into a lower dimensional space. PCA achieves this by transforming the original data into a new set of variables called principal components. These principal components each explain some of the variance found in the data and are often sorted by the amount of the variance they explain. Thus, if the first two or three principal components can explain most of the data variance, then a plot based on these will faithfully reflect the original data in a 2D or 3D space. Such visualization of multidimensional data may aid in the identification of important variables and underlying structures or patterns in the data [41].

2.8.3 Hypothesis testing

A statistical hypothesis test evaluates whether an observation supports a hypothesis (e.g., the distribution of two populations is significantly different). There are two hypothesis testing: a null hypothesis, H_0, and the alternative hypothesis H_1. By testing if H_0 is true, it is possible to acquire information about the alternative hypothesis. The test can be one-tailed, allowing statistical significance testing in one direction or two-tailed, making it possible to test for statistical significance in both directions. Here, a p-value is the probability of finding the observed or a more extreme event under the assumption that H_0 is true, which helps to determine the strength of the null hypothesis. In a hypothesis test, H_0 is dismissed if the p-value is less than a specific cutoff, referred to as a significance level (e.g., P-value < 0.05 or 0.01 indicates either a 5% or 1% error margin)[42].

2.8.4. T-test

The t-test, often called the student´s t-test, is a statistical test used to determine if there is a significant difference between the means of two groups or samples. The test assesses whether the observed differences are likely due to change or if they are statistically significant. Though there are several types of t-tests, the most common ones are the independent t-test and the paired samples t-test. An independent t-test compares the means of two independent groups to determine if they are significantly different. On the other hand, the paired samples t-test compares the means of two related groups or measures taken from the same sample at different times under different conditions[42]. In this thesis, the independent t-test is used to find differential peaks since the samples are independently treated when the specific antibodies are added.

3. Methods

3.1 Reference and Test Data

This chapter will describe the input data, their format, and sources of the data used in this thesis. The goal of most biomedical research is to find a therapy for humans. Therefore, when making this pipeline, the main goal was to make it compatible with the human genome. If more time were available, then it would be ideal for the pipeline to be expanded to model organisms (e.g., mice and zebrafish). The following sections will introduce a complete workflow and application of data based on human cells.

3.1.1 Reference Genome File

A reference genome sequence file is a representation of a specie´s genetic information, which is used as a comparison or guide in biological research. Reference genome files are not created by extracting and assembling one donor´s DNA but by combining multiple genomic clone libraries. Thus, a reference genome provides a haploid mosaic of diverse DNA sequences that should be representable for the species as a whole. Species-specific can be downloaded from multiple online databases, such as the National Center of Biotechnology Information - NCBI, which is used in this project [43]. The specific human reference genome used in this thesis is the Genome Reference Consortium Human Build 38 - hg38. The human genome consists of 23 chromosome pairs, where 22 are autosomes, not sex-dependent, and one is allsome, sex-dependent. The autosomes are typically named by number, such as chr1, chr2, chr3, etc. The allsome is commonly called X for female and Y for male, chrX or chrY. Moreover, all eukaryotes have a circular chromosome located in the mitochondria. This is also included in the reference genome used in this thesis. Thus, the human reference genome will comprise 25 chromosomes, 22 autosomes, two allosomes, and one mitochondrial chromosome. The reference genome will be used in three formats in this pipeline: FASTA, BED, and refFlat.

FASTA is a text-based file format for representing nucleotide or amino acid sequences. Every sequence begins with a greater than character “>” followed by a header describing the sequence. The sequence representation follows immediately after the header, consisting of a letter for each nucleotide or amino acid. The **FASTA** file of a human reference genome will be used in the alignment step in this pipeline. For example, the reference genome will be used to create indexing files required by the alignment software Bowtie2. The creation of these indexing files will not be part of the pipeline functions. The **BED** file format is a text file format used to store genomic regions as coordinates and associated annotations, making easy comparisons between files. The data is presented in tab or space-separated columns, where the number of columns may vary depending on the reference genome. The **refFlat** is a text-based file containing gene annotation information. Each line represents a gene prediction, with eleven tab-separated columns. These columns contain various information about the predicted genes, as seen in Table 3.1. In this thesis, only the first six columns will be evaluated when annotating genes.

Column	**Type**	**Description**
geneName	String	Name of gene as they appear in genome browser
name	String	Name of gene
chrom	String	Reference sequence chromosome or scaffold
strand	Character	+ or - strand
txStrand	Integer	Transcription start position (or end position for minus stand items)
txEnd	Integer	Transcription end position (or start position for minus stand items)
cdsStart	Integer	Coding region start (or end position for minus strand item)
cdsEnd	Integer	Coding region end (or start position for minus strand item)
exonCount	8	Number of exons
exonStarts	List of integers separated by ,	Exon start positions (or end positions for minus strand item)
exonEnds	List of integers separated by ,	Exon end positions (or start positions for minus strand item)

***Table 3.1** – Description of each column in the gene prediction file format refFlat.*

In this thesis, a reference genome for *Escherichia coli* will be used for spike-in normalization, which relies on alignment against another genome for signal calibration. Please refer to section 2.6.2 for details. Here, the reference genome Escherichia coli str.K-12 substr. MG1655 (E. coli) will be used in the spike-in normalization. The reference file is in FASTA format and will be used to create indexing files for genome alignment, likewise to the human reference genome. The user must create these indexing files before utilizing the proposed pipeline.

3.1.2 Chromosome Size File

The chromosome size file is a text file describing the length of each chromosome, which consists of two tab-separated columns, one with the chromosome names and the other with the size of the chromosome (the number of base pairs). The chromosome size file can be downloaded from the UCSC genome browser [44]. However, it contains more than the common 25 chromosomes (e.g., unlocalized scaffolds, alternative loci scaffolds etc.), which must be cleaned and sorted before usage.

3.1.3 Blacklisted regions

The ENCODE project has proposed a list of genomic (backlisted areas) to be excluded when analyzing ChIP-Seq and other similar high-throughput experiments [31]. Please refer to Chapter 2.6.3 for a more detailed description of the improved data quality after removing blacklisted genome regions. The users may choose the genomic regions to be excluded from the analysis by using a file with the same BED format.

3.1.4 CUT&Tag Sample Data

The data used in this thesis is obtained from a previous CUT&Tag publication by Kaya-Okur et al. (2020) [4] which is publicly available in NCBI´s Gene Expression Omnibus, GEO [45]. The corresponding SRA entries are listed in Table 3.2. The cell line used in this experiment was K562 (RRID:CVCL 0004), human leukemia blood cells. The current project will mainly focus on two histone modifications, H3K27me3 and H3K4me3. Furthermore, immunoglobulin G (IgG) control samples (non-specific binding) are also included in the pipeline example run, as control samples are essential to evaluate the quality of any experiments.

For this proposed data analysis pipeline, the input data must be in fastq format, a standard output data format from high-throughput sequencing experiments, which is a text-based format that stores both the biological sequence and a corresponding score sequence. For clarity in the data analysis, the sample file names must follow the pattern:

[sample name/histone]_rep[replication number]_[R1/R2]_[(technical replicates]].fastq

For instance: H3K4me3_rep1_R1.fastq or H3K27me3_rep2_R1_001.fastq. If not in the right format, the user needs to change their file names manually before using the proposed pipeline.

Sample type	**Name**	**GEO association**	**SRA entry**	**Given name**
H3K27me3	SH_Hs_k37m3_NX_0918	GSE145187	SRX8754646	H3K27me3_rep1
H3K27me3	SH_Hs_K27m3_Xpc_0107	GSE145187	SRX7713678	H3K27me3_rep2
H3K4me3	SH_Hs_K4m3_NX_0918	GSE145187	SRX7713692	H3K4me3_rep1
H3K4me3	SH_Hs_K4m3_Xpc_0107	GSE145187	SRX7713696	H3K4me3_rep2
IgG	SH_Hs_IgG_1x_0924	GSE145187	SRX8468909	IgG_rep1
IgG	SH_Hs_IgG_20181224	GSM3680227	SRX5545346	IgG_rep2

***Table 3.2** – Describes the origins of the Cut&Tag sequencing data used in this thesis, with name, GEO association, and SRA entry. Furthermore, the given names are used for the rest of the thesis as well as the pipeline examples.*

3.2 Hardware and software

3.2.1 Programming Language and Libraries

This project aimed to create a user-friendly command-line package for CUT&Tag data analysis, which allows scientists to perform the data analysis without extensive programming experience. To increase accessibility, the pipeline is designed as a Python [46] package (Python 3.8). Python is a popular and free programming language applied for many purposes, from machine learning to websites. Python is especially widely used for big data analysis in the scientific community since it performs excellently when working on massive data sets, has a simple structure, and is easily learned. Additionally, many useful bioinformatics packages are available in Python, providing an optimal solution for the current project goal.

Table 3.3 lists Python libraries required by the current pipeline package. Usually, Python includes a standard library including models that the users do not need to download or install again, and the proposed pipeline uses some of them. For example, “argparse”, “multiprocessing”, and “subprocess” are standard modules in Python; modules like “os” and “shutil” are used to help with operating system interaction; “Pathlib” and “glob” are standard Python modules used for path handling; module “re” is utilized to access regular expressions. Though these packages are part of the standard libraries in Python 3.8, users should check if the correct versions of these packages are available in their Python installation.

Further, some non-standard Python libraries will be needed in the current pipeline and can easily be downloaded and installed by a package managing system (e.g., Conda, pip). For example, “Numpy” [47] and “Scipy” [48] are used for various calculations, while “Panadas” is a wildly used package for data analysis and manipulation [49]. For gene annotation, a new similar function from HMST-Seq-Analyzer [50] is adopted, which may easily be downloaded and installed from GitHub (https://hmst-seq.github.io/hmst/). The proposed pipeline creates many plots for interpreting or visualizing the results, which requires some additional Python modules such as “seaborn” [51], “matplotlib” [52], and “plotnine” [53]. Lastly, “setuptools”, a standard part of the Python library, is used to install the pipeline package itself, which is simple and straightforward. Please see the example below for using the “setup.py” script to build and install the pipeline package.

```
$ python setup.py install
```

Python Package	Version	Usage	Standard
argparse	-	Make command-line	Yes
multiprocessing	-	Preform multiple processes	Yes
subprocess	-	Start a new subprocess	Yes
os	-	Operating system interaction	Yes
pathlib	-	Path handling	Yes
re	-	Regular expressions	Yes
shutil	-	File and directory operations	Yes
setuptools	-	Installation	Yes
glob	-	Retrieve file/pathnames	Yes
numpy	1.23.0	Calculations	No
pandas	1.5.2	Data analysis and manipulation	No
seaborn	0.12.2	Visualization	No
matplotlib	3.6.2	Visualization	No
plotnine	0.10.1	Visualization	No
	1.0	Gene annotation	No

Table 3.3 *-Python packages used in the pipeline created in this thesis, their version, usage, and if they are a part of the Python 3.8 standard library. Also, the Python packages are required for running the pipeline.*

3.2.2 Software

In addition to Python modules, several other public packages are needed in the analysis pipeline, please refer to Table 3.4. These external modules can easily be downloaded and installed by conda or pip. For example, FastQC [36] is a tool to run quality checks on the input reads and generate quality reports. Bowtie2 [48] is a sequencing alignment tool for mapping reads to a reference genome, as well as aligning control reads to a spike-in genome if that is part of the experimental protocol; Bowtie2 needs to have genome indexing files of the reference genome, which must be produced by the user before running the pipeline. PicardTools [48] is a JAVA package to mark and remove duplicated reads. SAMtools [54] is a package for interacting with high-throughput sequencing data for reading, writing, editing, indexing, and viewing different file formats; in the current project, it is mainly used for filtering and format conversion. BEDtools [55] is utilized to search for overlaps between genomic features and is used multiple times in the current pipeline for comparing and intersecting different genomic regions/features.

SEACR [56] is a peak calling program aimed at CUT&RUN and CUT&Tag data, which uses a global distribution of background signal to calibrate a simple threshold for peak calling. The SEACR software must be downloaded as a shell script file and be provided by the user. Lastly, the Galaxy-based software deepTools [37] is utilized to create heatmaps for the visualization of called peaks. Although installing and setting up all the aforementioned software might be a daunting task for users with limited informatics skills, the pipeline makes them easily useable without having to read up extensively. Furthermore, the pipeline developed in this thesis is an end-to-end application, starting with raw reads and ending with annotated enrichments. Thus, it is a simple and efficient tool for most users with limited bioinformatics skills.

Software	**Version**	**Usage**
FastQC	0.11.9	Quality control
Bowtie 2	2.2.5	Alignment
Picard	2.27.5	Duplication removal
SAMtools	1.6	Filtering and file conversion
BEDtools	2.30.0	Searching for overlaps
SEACR	1.3	Peak calling
deepTools	3.5.1	Visualization

Table 3.4 – *Software used in the pipeline created in this thesis, the versions, and their usage. Also, the software required for running the pipeline.*

3.2.3 Hardware for Development and Testing

The proposed pipeline was created and continuously tested on a MacBook Pro with an Apple M1 Pro chip, 16 GB memory, 1 TB disk, and macOS Ventura 13.1., therefore the pipeline is compatible with the MacBook operating system. The pipeline was also evaluated on SAGA – a supercomputer located at NTNU in Trondheim. SAGA is provided by Hewlett Packard Enterprise and has a computational power of roughly 140 million CPU hours per year. The computer consists of 364 computational nodes with various amounts of cores and 192 GiB. The test run was executed on a standard computer node that runs Intel Xeon-Gold 6138 2.0 GHz/ 6230R 2.1 GHz with CentOs Linux. The queue system Slum Workload Manager is used in Saga for effective job handling.

3.3 The Full Analysis Pipeline for CUT&Tag

The pipeline package implemented in this thesis performs several tasks in analyzing CUT&Tag sequencing data, which is inspired by the online data analysis tutorial created by Ye Zheng [38], but with a focus on designing a user-friendly command-line Python package. Nevertheless, some moderations were made regarding the correlation plots, removal of blacklisted regions, gene annotation, differential peak analysis, etc. In the following sections, a workflow of the pipeline will be described.

3.3.1 An Overview of The Pipeline

The proposed pipeline consists of nine main modules, each performing one task of the data analysis as described in Figure 2.6, except for the interpretation step. The figure illustrates the workflow of the whole pipeline and its running order. The first step, `fastqc`, quality control of raw sequencing data may be skipped or conducted at any time in the pipeline; nevertheless, it is highly recommended to check the data quality of the reads before continuing the rest of the analysis. Please note that the last eight modules depend on each other, and their running order should not be changed, and the order should be:
`bowtie2_alignment` -> `remove_duplicates` -> `fragment_length` -> `filtering` -> `spike_in_calibration` (if part of the experiment composition) -> `peak_calling` -> `differential_analysis`. Nevertheless, there is a possibility to utilize one or a few of the functions when it is necessary. If that is the case, then the user must ensure that the input files have the correct formats and file names for the functions, which will be described further in Chapter 3.4.

3.3.2 `fastqc`

In this function, the names of the samples are extracted and paired according to reads numbers. For instance, H3K4_rep1_R1 and H3K4_rep1_R2 are paired reads for the H3K4me3 experiment. Then, the software FastQC [35] is utilized to generate quality reports for the input files. Reports of both reads of the same sample are exported in a common folder. Please refer to Chapter 3.4 for a more detailed description of the pipeline architecture, input, and output files.

3.3.3 `bowtie2_alignment`

The `bowtie2_alignment` function is used to align sequencing reads to a reference genome or spike-in genome if the spike-in examination is part of the experimental protocol. The Bowtie2 [36] indexing files must be created before using this function (e.g., by using Bowties´2 **build** command), which takes some time to generate the indexing files. The `bowtie2_alignment` function will extract the file names of the samples and pair them according to the read files. Subsequently, the function utilizes Bowtie2 to align the reads to a reference genome.

Below is an example of the Bowtie2 alignment command with all parameters utilized in this pipeline:

```
bowtie2 --end-to-end --very-sensitive --no-mixed --no-discordant --phred33 -I 10 -X 700
```

The --*end-to-end* parameter means that Bowtie2 searches for alignments involving all the read characters.
The --*very-sensitive* is a collection of variables implicating many others such as: -*D* 20, -*R* 2, -*N* 0, -*L* 20 -*i* S,1,0.50. where:
-*D* is the number of consecutive seed extension attempts that may "fail" before Bowtie2 moves. For a seed extension to "fail", it must not yield a new best or second-best alignment. It is set to 20, which increases sensitivity compared to the default value of 15.
-*R* is the maximum number of times the software will "re-seed" reads with repetitive seeds. "Re-seeding" means that Bowtie2 chooses a new set of reads with the same length and same number of mismatches allowed at different offsets and searches for more alignments. If the number of seed hits divided by the number of seeds aligned at least once is greater than 300, then a read is considered to have repetitive seeds. Here, it is set to the default value of 2.
-*N* sets the number of mismatches allowed in seed alignment during multiseed alignment. In this pipeline, this is set to the Bowtie2 default value of 0.
-*L* sets the length of the seed substrings to align amid multiseed alignment. This is also set to the default value of 20 for end-to-end aligning.

-i determines a function governing the interval between seed substrings to use during multiseed alignment. Here, the function is set to be *f(x) = 1 + 0.5*sqrt(x),* where x is the read length. This function creates longer intervals than the default value function *(f(x) = 1 + 1.5*sqrt(X))* which increases sensitivity.

The *--no-mixed* parameter disables Bowtie2´s default behavior of finding alignments for individual pairs when the software cannot find a concordant or discordant alignment for a read pair.

The *--no-discordant* disables Bowtie2´s default behavior of looking for discordant alignments when no concordant alignments are found. A discordant alignment is an alignment where both pair-mates align uniquely; however, it does not satisfy the paired-end constraints.

The *--pherd33* parameter signifies that the input ASCII equals the Pherd quality plus 33. Pherd quality scores are the quality scores that are encoded as ASCII characters in fastq files.

Lastly, the *-I* and *-X* parameters decide the minimum and maximum fragment length for valid pair-end alignments. Here, the interval is set to 10-700, which includes the typical fragment lengths of 200-400, where Bowtie2 alignment is very efficient.

After the alignment, the pipeline function will create summary tables and plots These depend on whether the alignment was normal or spike-in, yet all contain information on sequencing depth, alignment rate, and the number of mapped fragments. The output is further discussed in Chapter 3.4.

3.3.4 `remove_duplicates`

The `remove_duplicates` function utilizes two functions from the Picard software [57]; **SortSam** and **MarkDuplicates**. Firstly, the input files are sorted with **SortSam**, the sorting parameter is set to: *SORT_ORDER=coordinate*, which means the input files are sorted by coordinates. Next, the function calls **MarkDuplicates** to mark all the duplications present in the input files. Then, **MarkDuplicates** is called again, now with the parameter *REMOVE_DUPLICATES = true*, which removes the duplicates. After utilizing Picard, `remove_duplicates` will provide information of library size and duplicated fragments in each sample based on Picard summary text files. Subsequently, it calculates the number of unique fragments in each sample based on Formula (I). Finally, it creates a summary table and plots figures for visualizing the library size, number of unique fragments, and duplicated fragments in the samples.

(I)

$$Number\ of\ Unique\ Fragments = Number\ of\ Mapped\ Fragments\ \times \frac{1 - Duplication\ rate}{100}$$

3.3.5 `fragment_length`

This `fragment_length` function assesses the fragments' length after sequencing, indicating whether the experiment was successful or not. The function uses SAMtools [54] and its function **view** for the analysis. If no parameters are specified, **view** will export all alignments in the input file. Here, the *-F* parameter of **view** excludes any bits set present in the FLAG field. For example, *-F 0x04* is set to exclude unmapped reads since they should not be included in the fragment length assessment. Next, the 9th column of the input data is extracted, which is the fragment length information. The fragment lengths greater than zero are counted based on the frequency of a specific length appearing in the input file. The fragment length and the corresponding count number are exported to a text file for making a weighted violin plot, where the weights are calculated by formula (II) for each fragment length. Finally, all samples are combined into one text file for plot-making.

(II)

$$Weight = \frac{Number\ of\ Fragments}{All\ Fragments}$$

3.3.6 `filtering`

The `filtering` function is used to filter alignment files and convert them into BED format by using SAMtools [54] and BEDtools [58]. Similar to `fragment_length`, SAMtools **view** is used to filter out unwanted contents. Firstly, **view** is utilized with a *-q* parameter to eliminate alignment results below a predefined quality score (e.g., a minimum score equals two in the current work). Furthermore, **view** with the *-F* parameter set to *0x04* is used to remove unmapped reads and with the *-bs* parameter to convert the input file to BAM format. For BEDtools the **bamtobed** option is used with a *-bedpe* parameter to convert BAM format files to BEDPE format, which allows for reporting two paired-end alignments in one single text line. It is easy to filter paired alignments from different chromosomes based on BEDPE format. Finally, several Unix command-line functions are used to remove fragments with more than 1000 base pairs to increase specificity and wrongfully paired reads. The fragment-related columns from the BED files are exported for further data analysis.

Additionally, the `filtering` function can evaluate the reproducibility between biological replicates or conditions. Reproducibility is the consistency of measurement between replicated experiments which is used to validate the similarity between biological replicates and evaluate the quality of experiments. In the current work, blacklisted genome regions are removed before checking the reproducibility. Here, functions **makewindows**, **intersect**, and **map** from BEDtools are utilized for removing blacklisted regions.

First, **makewindows** is used to create 500 base pair long bins based on the chromosome sizes of the reference genome. Then, a BED format reference genome file without blacklisted regions is generated by applying **intersect** with the *-v* parameter to both the window bins and the blacklisted regions, where the *-v* parameter reports entries that are not overlapping between the two files. Finally, the **map** function is used to map overlapping features between the blacklist-removed reference genome file and the fragment-related BED files. Here, the parameter *-o mean* is used, meaning that if multiple features from the sample data overlap with a bin, the mean overlap score is used.

Now, the reproducibility can be assessed based on the aforementioned cleaned and filtered fragment files. A count matrix based on the number of overlaps between a sample feature and the blacklist-removed reference genome. Then, all bins with zero overlaps across all samples are removed, and the matrix is log2 transformed before it is used to calculate a Person´s correlation matrix for plotting the figure. Here, a secondary correlation matrix is also made, where all bins with a total count number less than eight across samples are removed before calculating correlation coefficient.

3.3.7 `spike_in_calibration`

The `spike_in_calibration` function normalizes the signal by calculating normalization factors from spike-in alignment. Though the spike-in normalization is optional, it is recommended when the experimental protocol is available. It uses the spike-in sequencing depth to create normalization factors for the corresponding sample, see Formula (III), where C is a constant in the pipeline (e.g., 10,000) to avoid small fractions in the normalized data set. The normalization method assumes that the ratio of fragments mapped to the spike-in genome is the same in all samples. This is a reasonable assumption since each sample did contain the same number of cells. The raw measurements will be multiplied by this scaling factor to obtain normalized coverage, as in Formula (IV). In this work, the normalization is done by using BEDtools **genegov** function with parameters *-scale* and *-bg*, which scale the data and convert them to BedGraph files, respectively. In the end, the `spike_in_calibration` function plots scaling factors and normalized fragment counts of all samples.

(III)

$$Scaling\ factor = \frac{C}{Mapped\ fragments\ to\ spike-in\ genome}$$

(IV)

$$Normalized\ coverage = (Primary\ genome\ coverage) \times (Scaling\ factor)$$

3.3.8 `peak_calling`

The `peak_calling` function uses SEACR (Sparse Enrichment Analysis for CUT&RUN) [32] to find enriched regions of the genome, followed by an assessment of the reproducibility and the proportion of fragments in peaks regions (FRiPs). SEACR is designed to call peaks and enriched areas from chromatin profiling tools (e.g., CUT&Tag) with very low background data (i.e., regions with no read coverage). SEACR defines peaks as continuous blocks of base pair coverage that do not overlap with blocks of background signal from the IgG control samples. Thus, the function will first run peak calling on the experimental samples by using the control IgG samples as a reference for background noise. Then, SEACR finds the 1% most enriched peaks from the experimental samples. Here, the *non* parameter is utilized because the input data is not normalized. After the peak calling, the pipeline function assesses the reproducibility between the biological replicates by computing the number of peaks overlapping between the replicates and then dividing it by the number of peaks from one sample multiplied by 100, this is to get the percentage, please refer to Formula (V). In each sample, it is done for both the SEACR results with IgG as background reference and the top 1% SEACR ranked peaks. To find the proportion of fragments in peak regions, BEDtools [55] **intersect** is used to find overlaps between the fragments and peaks for each sample. Then, for each sample, the number of fragment-peak overlaps is divided by the number of fragments and multiplied by 100, this is again done to get the percentage; please refer to Formula (V). Lastly, `peak_calling` plots the number of peaks, the peak width, the reproducibility percentage, and the FRiP percentage.

The reproducibility analysis is done for both the SEACR results with IgG as background reference and the top 1% of peaks in each sample. To find the fragment proportion in peaks regions (FRiPs) BEDtools function **intersect** is utilized to find overlaps between the fragments and peaks of a sample. Finally, the fragment-peak overlaps are divided by the sample´s total number of fragments and multiplied by 100 to get the percentage, as in formula (VI). As the last step, the pipeline function creates plots to display these results as well as the widths of the peaks.

(V)

$$Reproducibility\ (\%) = \frac{Number\ of\ Overlaps\ between\ Replicates}{Number\ of\ Peaks\ for\ One\ of\ the\ Replicates \times 100}$$

(VI)

$$FRiP = \frac{Numner\ of\ Fragment - Peak\ overlaps}{Number\ of\ Total\ Fragments\ \times\ 100}$$

3.3.9 `heatmap`

The `heatmap` function of the pipeline can be used to visualize the enrichment of target protein in predefined genomic regions or the called peaks in heatmaps. First, `heatmap` uses Samtools [54] **sort** and **index**, to sort and index aligned BAM files before converting them to bigWig files. It utilizes three functions from the deepTools [37] software: **bamCoverage**, **computeMatrix**, and **plotHeatmap**. **bamCoverage** is used to convert sorted and indexed BAM files into bigWig format. After the conversion, reference genomic regions and bigWig files are inputted to **computeMatrix,** which calculates scores per genome region based on the bigWig scores, and then creates a scoring matrix. This scoring matrix will be used to visualize the enrichment of the target protein in the specific genomic regions. The **computeMatrix** parameters to create such a matrix are:
-beforeRegionStartLenght 3000, --regionBodyLength 5000, --afterRegionStartLength 3000, --skipZeros, -p 8, which implies that 3000 base pairs upstream and downstream of the 5000 base pairs genome region are included in the calculations. The calculation will not include regions with only zero scores. The *-p 8* parameter makes the function utilize eight cores for the calculations, which may be changed by the user. Finally, visualize the results, the aforementioned enrichment/genome score matrix is inputted to **plotHeatmap** with the parameter *-sortUsing sum*, where scores are sorted based on their sums before generating the heatmap.

To visualize the enrichment of target protein in the peaks, the midpoint of a signal block (i.e., called peaks) obtained from SEACR is used to align signals in heatmaps. To create an individual matrix for each sample, **computeMatrix** is used with the same bigWig files, but now the peak signals are used as the reference genomic regions. Here, the parameters are: *skipZeros, -p 8, -beforeRegionStartLenght 3000, --afterRegionStartLength 3000* are used; Please note that the parameter *--referencePoint center* is used to use the midpoint of the peak signal block as reference.. In the end, **plotHeatmap** is used to generate heatmaps based on these score matrixes.

3.3.10 `differential_analysis`

The `differential_analysis` function is the final function in the pipeline, which performs differential analysis of called peaks between samples. There are several steps in this function, where the three main steps are: 1. Combining called peaks from multiple samples and merging them into a combined BedGraph file; 2. Differential analysis, 3; Annotation and plotting. First, the top 1% called peaks from each sample will be combined into one BED file, where the overlapping features are found and combined into one single feature by using BEDtools [55] **merge** with the *-o* parameter. Subsequently, blacklisted regions are removed from the fragment BedGraph files with the same procedure as in `filtering`; where, 100 bp window bins are created for each chromosome. The small window bin size makes for easier annotation of the called peaks. These window bin files with mapped peaks are combined into one file, where peak signals may be quantile normalized and logarithmically transformed if not already normalized by spike-in normalization. At the end of step 1, all peak signals are mapped to the 100 bp window bins by BEDtools **intersect** with *-wa -wb -loj* parameters. These parameters ensure that both the original entries of each file and the overlap features are exported.

In the second step, the differential analysis, a two-sided students t-test is used to find statistically significant differential peaks between the two conditions/samples (e.g., a t-test with p-value < 0.05 as significance level). To illustrate the sample based on their called peak signals, PCA is applied to the data sets, and the first three principal components are used to generate a 3D plot for visualizing the samples. Finally, the HMST-seq-analyzer [50] package is used to annotate the location of differentially enriched regions to various predefined genomic regions (e.g., transcription start site (TSS), transcription end site (TES), genes, 5´ distance regions, enhancers etc.).

3.4. Pipeline Architecture and User Interaction

The workflow, pipeline tasks, and functions were described in the earlier chapter. Here, the focus will be the description of pipeline architecture and user interaction, such as the input/output and options/parameters of functions.

3.4.1 General architecture

As mentioned previously, there are nine major tasks in the pipeline, from the quality control of data to the annotation of results. Except for `differential_analysis`, which consists of various calculations (e.g., hypothesis testing and PCA analysis) and needs several scripts to accomplish the goal, all other tasks consist of one Python script using one or more external analysis tools. The pipeline has a simple architecture with a total of 19 Python scripts. Thus, it is relatively easy for users to modify the pipeline if it is needed in the future. The architecture, script relationships, input and output files of the proposed pipeline are shown in Figure 3.1. Please note that the pipeline follows a specific workflow, and two of the functions (`remove_duplicates`, and `spike_in_`calibration) are optional.

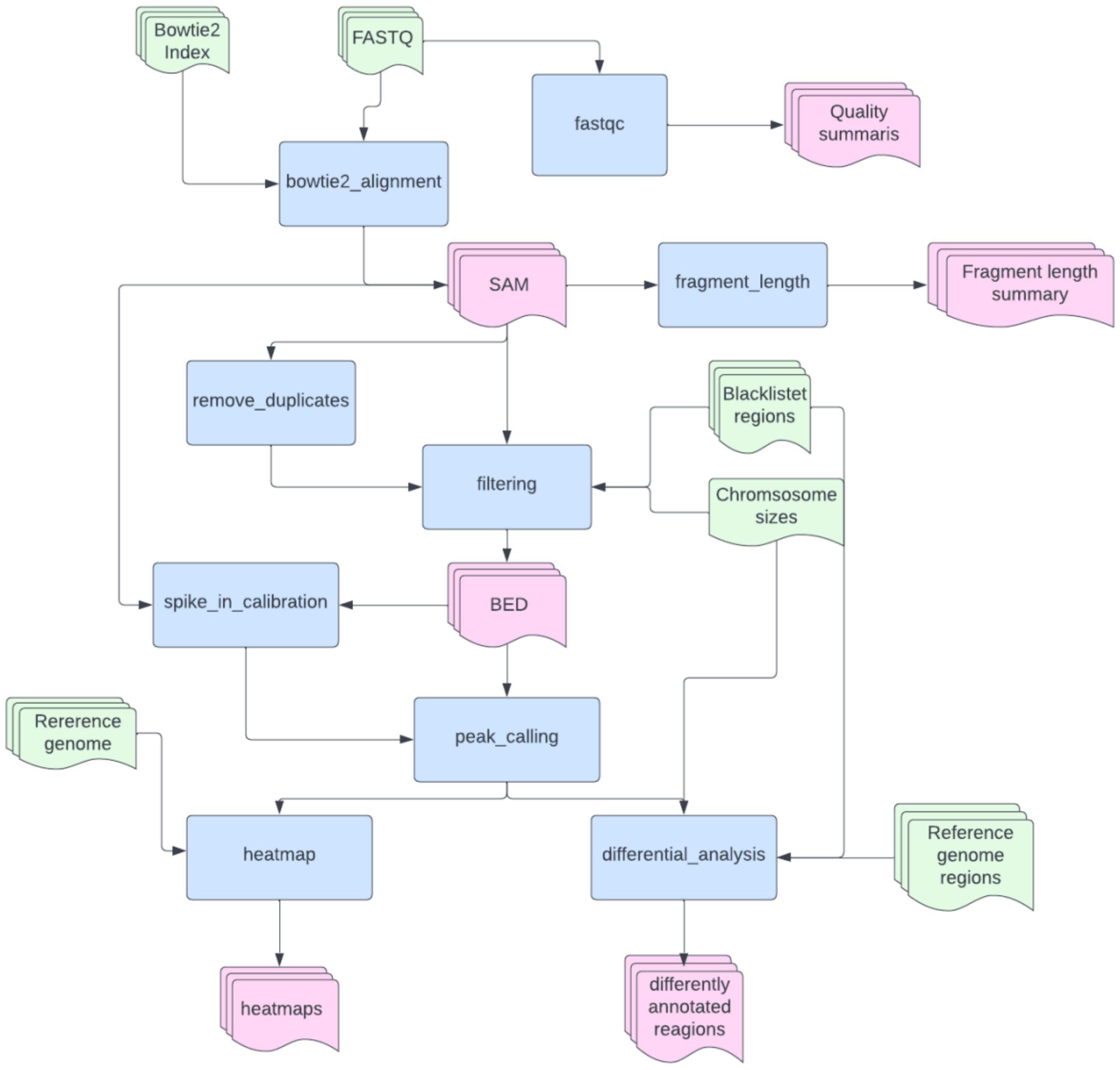


Figure 3.1 *- The pipeline architectural design. The main input data files that must be provided by the user are depicted in green. The blue boxes represent the different tasks, and the pink shapes depict their output. Not all outputs are included in this figure for simplicity.*

Most of the functions in the pipeline are dependent on each other to generate the correct file format for the rest of the analysis where different input files are needed in the different steps. Please refer to Chapters 3.4.2 to 3.4.10 for detailed descriptions.

All results from the analysis will be exported to a main output folder, “CutAndTagAnalyzer”, which contains several sub-directories. The main output folder is created automatically by the pipeline, where the results from different tasks are organized and easily accessible in the sub-directories. Figure 3.2 shows an example of the “CutAndTagAnalyzer” main folder and subfolders generated by a full run of the pipeline. Since many results files are exported by the pipeline, the “summary_tables” folder stores figures and summary tables, which may be easily checked during the data analysis.

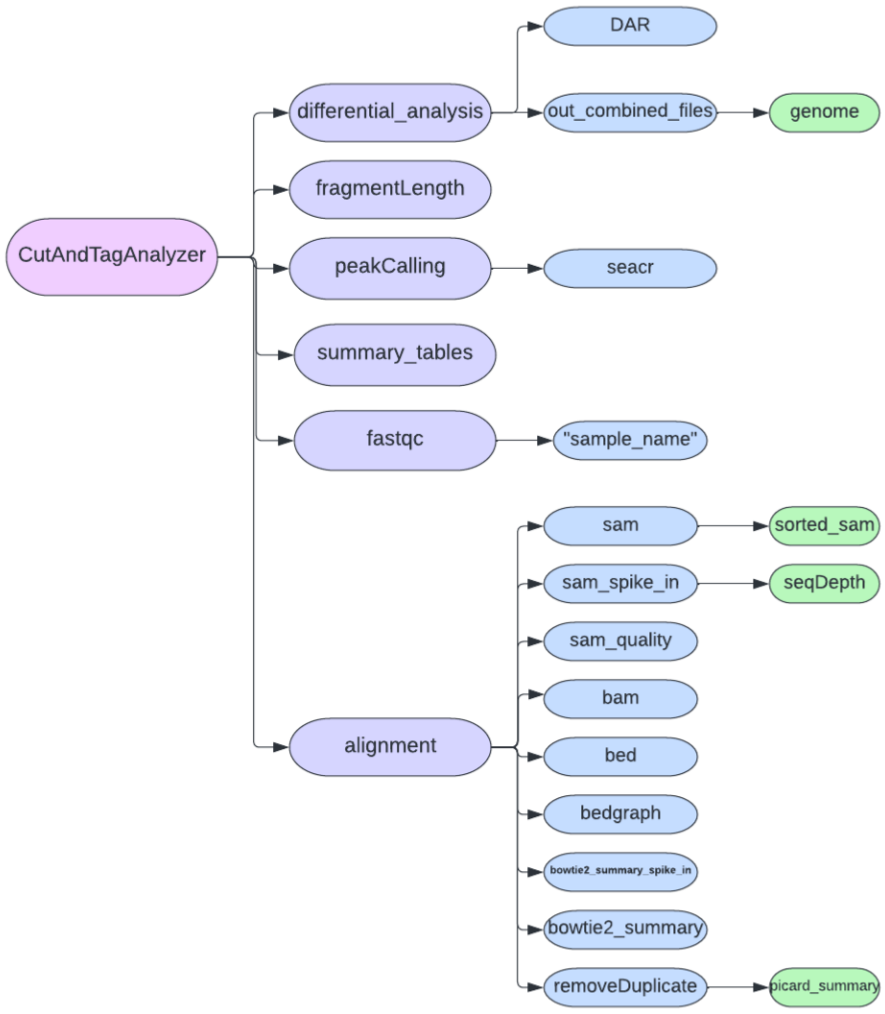


Figure 3.2 *– Architecture of output directories created by the pipeline implemented in this thesis.*

3.4.2 Data quality control - `fastqc`

As described in Chapter 3.3.2, this function aims to check the quality of raw reads by inputting fastq files obtained from high-throughput sequencing. All parameters/options for this function are shown in Table 3.5. For example, the *-f/--fastq* argument represents the whole pathway of the directory containing fastq files, where files should be named as follows:

[sample name/histone]_rep[replication number]_[R1/R2]_[(technical replicates]].fastq

By using fastQC [35], this function creates several quality control reports for each fastq file, which are stored in the subfolder "fastqc" of "CutAndTagAnalyzer" with the same name as the input fastq file. For example, HTML summary reports containing several figures related to experimental quality, including:

- Per base sequence quality
- Per base sequence quality score
- Per base sequence content
- Per base GC content
- Per base N content
- Sequence length distribution
- Sequence duplication levels
- Overrepresented sequences
- Adapter content

These plots and a summary table are also available in a zipped file with the same name as the input fastq file. The user may change the location of the output directory by using the *-o/--out_dir* argument, the default being the user´s current working directory.

Default	**Argument**	**Description**
None	*-f / --fastq*	Directory containing fastq files
Current working directory	*-o / --out_dir*	Output directory

Table 3.5 *– The `fastqc` function input, both required (first one) and optional.*

3.4.3 Mapping to reference genome - `bowtie2_alignment`

The `bowtie2_alignment` function aligns input fastq files to a reference, from which summary table(s) and plots will be generated. There are two required and three optional arguments in the function, please refer to Table 3.6. Here, the *-f/--fastq* argument must be provided with the full pathway of the directory containing fastq files that should be aligned to the reference genome. The *-i* argument is the full path of Bowite2 [36] indexing files, which must be provided. For optional arguments, the *-o/--out_dir* is the full path of an output directory. If the input data contains technical replicates, the argument *-m*/*--merge_replicates True* should be used as it will merge files with the same sample name and read-number to one single file before the alignment. If the fastq files should be used for spike-in alignment, the argument *-s*/*--spike_in* must be used, and the alignment will be performed with the same parameter as the common reference genome. However, the output files from a spike-in alignment will have a different name. All output files of the alignment will be stored in subdirectories of the "CutAndTagAnalyzer" main directory.

SAM files and Bowtie2 alignment summary text files will be exported from the Bowtie2 alignment. If the *-s/spike_in* argument is set to *True*, then the results will be stored in "CutAndTagAnalyzer/alignment/sam_spike_in," and "CutAndTagAnalyzer/alignment/bowtie2_summary_spike_in", respectively. Please refer to Figure 3.2 for an illustration of such output. However, if the *-s/spike_in* argument is set to *False* (default), then these outputs will be stored in "CutAndTagAnalyzer/alignment/sam" "CutAndTagAnalyzer/alignment/bowtie2_summary". If the alignments of the reference genome and spike-in genome are performed, then the alignment summary table will contain both results. Plots visualizing the alignment summary will be generated and exported in one PNG file, containing the following plots:

- Boxplot of sequencing depth to the human reference genome (per million)
- Boxplot of mapped fragments to the human reference genome (per million)
- Boxplot of alignment rate to the human reference genome (% of mapped fragments)
- Boxplot of alignment rate to the spike-in genome (% of mapped fragments)

Default	**Argument**	**Description**
None	*-f / --fastq*	Path to directory containing FASTQ files
None	*-i / --bowtie2_index_pathway*	Path to directory containing Bowtie2 index files
Current working directory	*-o / --out_dir*	Path to output directory
False	*-s / --spike_in*	Spike-in alignment or not
False	*-m / --merge_replicates*	To merge technical replicates or not

***Table 3.6** – The* `bowtie2_alignment` *function input, both optional (last two) and required.*

3.4.4 Removal of duplicated reads after alignment - `remove_duplicates`

The function is used to remove duplicated reads mapped to the same location of a reference genome after alignment. Here, the user must provide the full path of a directory containing SAM files in the *-s*/*--sam* argument. The *-o*/*--out_dir* can be used to set an alternative output directory. More description of arguments for this function is shown in Table 3.7. The `remove_duplicates` will create a "sorted_sam" directory under the SAM-file input directory, where sorted by coordinates are stored. It will also create a "removeDuplicates" folder under the main output directory "CutAndTagAngalyzer", where SAM files after removing duplicates will be stored (e.g., file names with postfix string "rm.Dup.sam"). Finally, it creates "picard_summary" under the "removeDuplicates" folder, where Picrad summary text files will be stored. These summary files will be used to generate a summary table and corresponding plots, which will be stored in "summay_tables". The plots will consist of one PNG file containing the following:

- Boxplot of duplication rate (%).
- Boxplot of estimated library size.
- Boxplot of the number of unique fragments.

Default	**Argument**	**Description**
None	*-s / -- sam*	Directory containing SAM files
Current working directory	*-o / --out_dir*	Output directory

***Table 3.7** – The* `remove_duplicates` *function input, both required (first one) and optional.*

3.4.5 Assessment of Fragment Length Distribution `fragment_length`

The `fragment_length` function evaluates the distribution of mapped fragment length in input SAM files and shows the results in a violin and a line plot. There is only one required argument -*s*/*--sam*, which is the full path to a directory containing SAM files exported by alignment. An optional argument *-o*/*--out_dir* may be used to change the output directory. Please refer to Table 3.8, where all arguments for this function are shown. This function will create a folder "fragmentLength" under the main output folder "CutAndTagAnalyzer" for storing the results, such as fragment-length information of each input file. In addition, a summary file with information from all the input samples is exported to the "summary_tables" folder and will be used for generating two PNG files with the plots shown below:

- A weighted violin plot of the fragment length for each sample.
- A line plot of the fragment length for each sample.

Default	**Argument**	**Description**
None	*-s / --sam*	Directory containing SAM files
Current working directory	*-o / --out_dir*	Output directory

Table 3.8 *– The* `fragment_length` *function input, both required (first one) and optional.*

3.4.6 Alignment Results Filtering - `filtering`

As described in Chapter 3.3.6, the `filtering` function performs data filtering for mapped reads based on their alignment quality, and file format conversion before high-level data analysis. This function has multiple input arguments, as shown in Table 3.9. First, the full path to a directory containing SAM files exported from an alignment must be provided by the user in the -*s*/*--sam* argument. Secondly, in the *-bl*/*--blacklist,* the user must input the path to a BED file of blacklisted regions that will be used to filter sample data. Finally, in the *-cs/--chromosome_sizes* argument, the path to a sorted chromosome size file must be provided. An optional argument, -o/--*out_dir* can be used to set an alternative path for the outputted data. The `filtering` function exports result files in several directories (e.g, sam_quality, bam, bed, and summary_tables) Under the "sam_quality" folder, quality filtered SAM files are stored. There are three types of BAM files in the "bam" directory (e.g., the BAM files of the quality filtered SAM files – ".bam", the filtered BAM files with only uniquely mapped reads – ".mapped.bam", and the sorted uniquely mapped reads – ".mapped_sorted.bam"). In the "bed" folder, there are four types of BED files (e.g., BED format of ".mapped_sorted.bam" – ".bed" and its clean version – "clean.bed" that contains only reads from the same chromosome with fragment length smaller than 1000 bp; the sorted fragment file – ".framents.bed", and predefined windows of 500 bp bin size file for all chromosomes with mapped fragments after removal of blacklisted regions – "500.b.windows.BlacklistFiltered.bed") In the "summary_tables" folder, two JPG files of correlation coefficient heatmaps are exported: "corrcoef_heatmap4logCount_all.jpg" (full data plot) "corrcoef_heatmap4logCount_filtered_gt8.jpg" (filtered data plot), which may be used to evaluate the quality of biological replicates as described in Chapter 3.3.6.

Default	**Argument**	**Description**
None	*-s / --sam*	Path to directory containing SAM files
None	*-cs/ --chromosome_sizes*	Path to file with sorted chromosome size information
None	*-bl/ --blacklist*	Path to file BED with blacklisted regions
Current working directory	*-o / --out_dir*	Output directory

Table 3.9 *– The `filtering` function input, both optional (last one) and required.*

3.4.7 Spike-in Calibration - `spike_in_calibration`

The `spike_in_calibration` function is used to remove experimental bias by normalizing fragment counts based on sequencing depth to a spike-in genome. Its arguments are listed in Table 3.10. The *-ss/--sam_spike_in* argument is the path to SAM files exported from spike-in alignment (e.g., "sam_spike_in" from `bowtie2_alignment`) and must be provided by the user. The *-cs/--chromosome_sizes* argument is a sorted chromosome sizes file, which is also a required input from the user. The *-tbl/--fragment_table* is an input file, the default being the summary table "bowtie2_alignment_ref_and_spike_in.csv" exported by `bowtie2_alignment`, but it accepts other CSV tables if the column names are ["Sample", "Replication", "SequencingDepth", "MappedFragments", "AlignmentRate", "MappedFragments_SpikeIn", "AlignmentRate_SpikeIn"]. The *-o/out_dir* may change the output directory from the current working directory, which is the default. The `spike_in_calibration` export the normalized data in BedGraph format files under the

"CutAndTagAnalyzer/alignment/bedgraph" folder. Results from this function will be found in "summary_tables" and include a summary table based on the input file from the *-tbl/--fragment_table* argument and one PNG file with the following plots:

- Boxplot of spike-in scaling factors, for each sample
- Boxplot of normalized fragment counts, for each sample

Default	**Argument**	**Description**
None	*-b / --bed*	Path to directory containing BED files
None	*-cs/ --chromosome_sizes*	Path to file with sorted chromosome size information
None	*-ss/ --sam_spike_in*	Path to directory with SAM files from spike-in alignment
summary_tables/ bowtie2_alignment_ref_and_spike_in.csv	*-tbl / --frament_table*	Alignment summery table
Current working directory	*-o / --out_dir*	Output directory

Table 3.10 *– The* `spike_in_calibration` *function input, both optional (last one) and required.*

3.4.8 Calling Enriched Regions from Chromatin Profiling Data - `peak_calling`

As described in 3.3.8, the `peak_calling` function is used to find enriched regions or call peaks from chromatin profiling data. The input arguments of this function are seen in Table 3.11. The *-bg/--bedgraph_ex* and the *-c/--bedgraph_control* are the required input path of BedGraph files and the path to a control BedGraph file, respectively. Here, the files with the same sample name as the control file will be excluded from further analysis if both the control and conditional samples are stored in the same folder. The *-s/--seacr_path* is the full path to the SEACR shell script, which the user must provide. Lastly, the user must input the pathway to BED files containing the same fragments as the BedGraph files in the *-f/--fragments* argument. For the optional arguments, the default setting of *-tbl/--fragment_table* is the summary table (bowtie2_alignment_ref.csv) from `bowtie2_alignment` but can be changed to a file with these column names: ["Sample", "Replication", "SequencingDepth", "MappedFragments", "AlignmentRate"]. The *-o/--out_dir*, may be used to set up the output path, with the default setting being the user´s current working directory. All results or output BED files from the SEACER peak calling are stored directory: "CutAndTagAnalyser/PeakCalling/seacr", see Figure 3.2 for visualized output structure. Finally, a summary table based on the *-tbl/--fragment_table* argument is exported in the "summary_tables" folder, as well as the following plots in PNG format:

- Boxplot of the number of peaks found, both control and top 0.01, for each sample.
- Violin plots of the peak widths, both control and top 0.01, for each sample.
- Boxplots of Fragment proportion in peaks regions (FRiPs)
- Barplots of the peak reproducibility, both control and top 0.01, for each sample.

Default	**Argument**	**Description**
None	*-bg / --bedgraph_ex*	Path to directory containing BedGraph files
None	*-c / --bedgraph_control*	Path to control BedGraph file
None	*-s / --seacer_path*	Path to SEACR shell script
None	*-f / --fragments*	Path to directory containing BED files
summary_tables/ bowtie2_alignment_ref.csv	*-tbl / --frament_table*	Alignment summery table
Current working directory	*-o / --out_dir*	Output directory

Table 3.11 *– The* `peak_calling` *function input, both optional (last one) and required.*

<u>3.4.9 Visualization of Results - `heatmap`</u>

As explained in Chapter 3.3.9, the `heatmap` function generates heatmaps of enrichment of chromatin profiling or protein binding data in specific genomic regions or peaks. There are four required arguments and two optional arguments in this function, which are listed in Table 3.12. The matrices used to make these heatmaps are based on BigWig files that are converted from BAM files. The user must provide an input path to BAM files by the *-b/--bam_dir* argument, give a BED format reference genome file in the *-r/--ref* argument, and input a BED format file of blacklisted regions in the *-bl/--blacklist* argument. Finally, the *-p/--peaks* argument, an input path to BED files obtained from SEACR peak calling must be provided. Other optional arguments are the *-c/--cores* argument for defining the number of parallel processes (default = 8) used in the calculations; the *-o/--out_dir* argument for setting an alternative output path. In this function, the input BAM files are converted to BigWig files and stored in the "bigwig" folder, which is under "CutAndTagAnalyzer/alignment". Please refer to Figure 3.2 for a complete output overview. The function generates several matrices that are used to create heatmaps, these are stored in "bigwig" and “seacr”, respectively. The heatmaps plots are exported as PNG plots in the “summary_tables” folder.

Default	**Argument**	**Description**
None	*-b/ --bam_dir*	Path to directory containing BAM files
None	*-r / --ref*	Path to reference genome BED file
None	*-bl / --blacklist*	Path to file BED with blacklisted regions
None	*-p / --peaks*	Path to directory containing BED files from SEACR peak calling
8	*-c / --cores*	Number of cores used in the calculation
Current working directory	*-o / --out_dir*	Output directory

Table 3.12 *– The `heatmap` function input, both optional (last two) and required*

3.4.10 High Level Data Analysis - `differntial_analysis`

As mentioned in Chapter 3.4.1, the last task in this pipeline, `differntial_analysis`, includes several scripts with many input arguments and output result files. For this function, there are seven compulsory and one optional argument, please refer to Table 3.12 for a detailed description. Some of the arguments are introduced here: the *-b/--bedgraph* argument inputs BedGraph format files of BED files obtained from SEACR peak analysis; the -*cs/chromosome_sizes* and *-bl/blacklist* arguments input a chromosome size file and a BED format file of blacklisted regions, respectively; the *-d/--data* argument is a full path of a data folder that contains the BED format files of the regions which will be studied, a text file of paths for the region BED files, and a BED format RefFlat file for the human genome. Finally, a list of sample names that will be used to do differential analysis must be provided in the *-la/--list_a* and the -lb/--*list_b* argument, respectively. All results are stored in the “differential_analysis” folder under the main output folder “CutAndTagAnalyzer”, please refer to Figure 3.2. In “differential_analysis”, there are window bin size files generated based on input BedGraph files after the removal of blacklisted regions and two subfolders (“DAR” and”out_combined_files”). In the ”out_combined_files” folder, there is an outputted BED file after combining the BedGraph files, the combined and merged BED peak files, and the overlap file between the two conditions. Here, there is a subfolder “genome” with result files from the genome annotation and the t-test result summary file. In the “DAR” subdirectory, there are annotated regions files. In the end, several plots are generated to visualize the results, these are exported to the “summary_tables” folder:

- 3D PCA plot based on the peaks mapped to a predefined window bin size (e.g., 500 bp in current work)
- Pie plot of the differently annotated regions (e.g., TSS, TES, enhancers, etc.)
- Bar plot of the differently annotated regions and whether they are differently up or down changes.

Default	**Argument**	**Description**
None	*-b/ --bedgraph*	Path to directory containing BedGraph files
None	*-cs / --chromosome_sizes*	Path to reference genome BED file
None	*-bl / --blacklist*	Path to file BED with blacklisted regions
None	*-d/ --data*	Path to directory with reference region files, and RefFlat file.
None	*-p / --peaks*	Path to directory containing BED files from SEACR peak calling
None	*-la / --list_a*	List of sample names from first group
None	*-lb / --list_b*	List of sample names from second group
Current working directory	*-o / --out_dir*	Output directory

Table 3.13 *– The* `differential_analysis` *function inputs, both optional (last one) and required.*

4. Results and Discussion

In this chapter, a demonstration of the CUT&Tag analysis pipeline in SAGA supercomputer will be described. It is an example run of the full pipeline where results and plots will be explained. Furthermore, some of the results from the current Python package will be compared with the corresponding figures in the Zheng Y et al. (2020) protocol, because both works implemented the same modules and analyzed the same experimental data sets.

As seen in Table 2.1, both histone modifications are customarily found in promotor areas, yet they have opposite effects on gene expression. Mainly, H3K27me3 is associated with gene repression, while H3K4me3 is linked to activation. Therefore, both data are expected to be present at the promotor and other gene regulatory regions. Though these modifications affect gene regulation differently, they may occur in the same areas called bivalent domains [10, 59].

4.1 Pipeline Environment and Set Up

A Conda package manager is needed (e.g., Miniconda https://docs.conda.io/en/latest/miniconda.html) for setting up the pipeline, from which the Python 3.8 running environment will be installed first. Then, several packages from the standard Python 3.8 library and some non-standard ones will be installed before utilizing the pipeline. As described in Chapter 3.2.1, a Python library installation can be done with the following command:

```
$ conda install pandas
```

Additionally, some external software will be installed through Conda through a specific channel, “-c bioconda” (installing through the bioconda channel [60]), which is specialized in bioinformatical software. Below is an example of installing one of the software, FastQC [35]:

```
$ conda install -c bioconda fastqc
```

Python packages, external software, and their version requirements for running the pipeline are listed in Tables 3.3 and 3.4, respectively. Nevertheless, there are two external analysis packages (SEACR [32] and HMST-seq-Analyzer [50]) that need to be installed manually. SEACR is a peak caller program for CUT&RUN and CUT&Tag experiments, which need to be downloaded (one shell script and one R script (https://github.com/FredHutch/SEACR)). HMST-seq-Analyzer is a differential methylation analysis tool that is not available in the Conda package library but needs to be downloaded through GitHub (https://hmst-seq.github.io/hmst/). Its installation is similar to that of other Python packages, such as described in Chapter 3.2.1 by running the setup.py script.

4.2 Demo Data

Histone modification data generated by Cut&Tag experiments from Kaya-Okur et al. (2020) are in the demo folder of the cut_and_tag_analyzer pipeline which are used in work. As mentioned in Chapter 3.1.4, the experimental datasets include two biological replications of both histone modifications (e.g., H3K27me3 and H3K4me3) and the IgG control All input files were in FASTQ format, which may need to be renamed according to the file naming protocol in Chapter 3.1.4 (Table 3.2). For the other input files and their configuration please refer to Figure 4.1

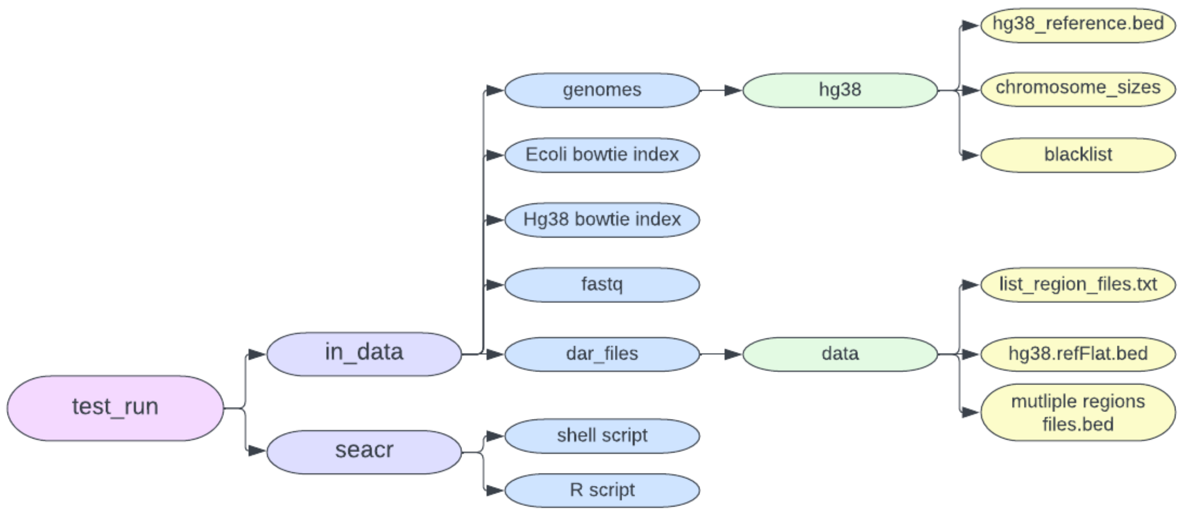
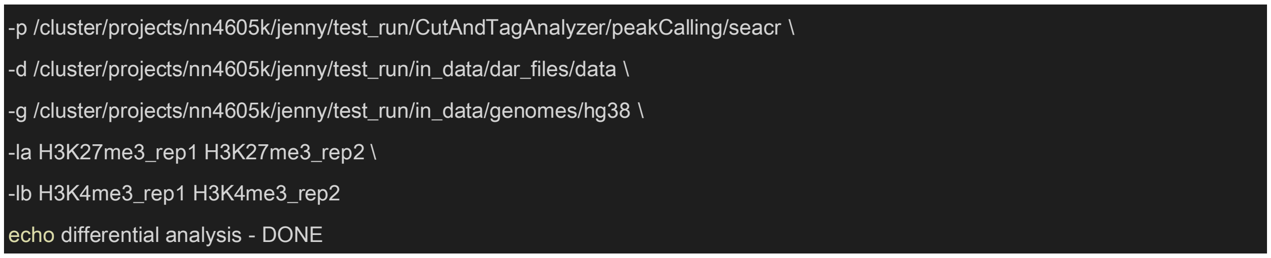


***Figure 4.1** – Architecture of input directories and files for the test run performed in this thesis.*

4.3 Demo Run

A sbash script in Figure 4.2 demonstrates a simple run of the pipeline, where all nine functions from the package are illustrated. Here, an “echo – DONE” line was added after each function/task to track the demo run.

```
#!/bin/csh

#SBATCH --account=nn4605k

#SBATCH --job-name=test_run

#SBATCH --time=1-0:0:0

#SBATCH --mem-per-cpu=15G --partition=bigmem

#SBATCH --cpus-per-task=10

## Do the job:

cut_and_tag_analyzer fastqc -f \
/cluster/projects/nn4605k/jenny/test_run/in_data/fastq
echo fastq - DONE


cut_and_tag_analyzer bowtie2_alignment \
-f /cluster/projects/nn4605k/jenny/test_run/in_data/fastq \
-i /cluster/projects/nn4605k/jenny/test_run/in_data/bowtie2_index_hg38/bowtie
echo bowtie2 alignment - DONE


cut_and_tag_analyzer bowtie2_alignment \
-f /cluster/projects/nn4605k/jenny/test_run/in_data/fastq \
-i /cluster/projects/nn4605k/jenny/test_run/in_data/bowtie2_index_ecoli/bowtie \
-s True
echo bowtie2 spike-in alignment - DONE


cut_and_tag_analyzer remove_duplicates \
 -s /cluster/projects/nn4605k/jenny/test_run/CutAndTagAnalyzer/alignment/sam
echo remove duplicates - DONE


cut_and_tag_analyzer fragment_length \
-s /cluster/projects/nn4605k/jenny/test_run/CutAndTagAnalyzer/alignment/sam
echo fragment length - DONE


cut_and_tag_analyzer filtering -s \
/cluster/projects/nn4605k/jenny/test_run/CutAndTagAnalyzer/alignment/sam \
-cs /cluster/projects/nn4605k/jenny/test_run/in_data/genomes/hg38/hg38.chrom.sizes.clear.sorted \
-bl /cluster/projects/nn4605k/jenny/test_run/in_data/genomes/hg38/blacklist.bed
echo filtering - DONE


cut_and_tag_analyzer spike_in_calibration \
 -b /cluster/projects/nn4605k/jenny/test_run/CutAndTagAnalyzer/alignment/bed \
-cs /cluster/projects/nn4605k/jenny/test_run/in_data/genomes/hg38/hg38.chrom.sizes.clear.sorted \
-ss /cluster/projects/nn4605k/jenny/test_run/CutAndTagAnalyzer/alignment/sam_spike_in
echo spike-in calibration - DONE


cut_and_tag_analyzer peak_calling \
-bg /cluster/projects/nn4605k/jenny/test_run/CutAndTagAnalyzer/alignment/bedgraph \
-c /cluster/projects/nn4605k/jenny/test_run/CutAndTagAnalyzer/alignment/bedgraph/IgG_rep1.fragments.normalized.bedgraph \
-f /cluster/projects/nn4605k/jenny/test_run/CutAndTagAnalyzer/alignment/bed \
-s /cluster/projects/nn4605k/jenny/test_run/SEACR/SEACR_1.3.sh
echo peak calling - DONE


cut_and_tag_analyzer heatmap \
-b /cluster/projects/nn4605k/jenny/test_run/CutAndTagAnalyzer/alignment/bam \
-r /cluster/projects/nn4605k/jenny/test_run/in_data/genomes/hg38/hg38_ref.bed \
-bl /cluster/projects/nn4605k/jenny/test_run/in_data/genomes/hg38/blacklist.bed \
-p /cluster/projects/nn4605k/jenny/test_run/CutAndTagAnalyzer/peakCalling/seacr
echo heatmap - DONE


cut_and_tag_analyzer differential_analysis \
 -b /cluster/projects/nn4605k/jenny/test_run/CutAndTagAnalyzer/alignment/bedgraph \
-cs /cluster/projects/nn4605k/jenny/test_run/in_data/genomes/hg38/hg38.chrom.sizes.clear.sorted \
-bl /cluster/projects/nn4605k/jenny/test_run/in_data/genomes/hg38/blacklist.bed \
```



```
-p /cluster/projects/nn4605k/jenny/test_run/CutAndTagAnalyzer/peakCalling/seacr \
-d /cluster/projects/nn4605k/jenny/test_run/in_data/dar_files/data \
-g /cluster/projects/nn4605k/jenny/test_run/in_data/genomes/hg38 \
-la H3K27me3_rep1 H3K27me3_rep2 \
-lb H3K4me3_rep1 H3K4me3_rep2
echo differential analysis - DONE
```

Figure 4.2 – *Bash script performing a test run of the entire pipeline created in this thesis.*

The current demo was run on SAGA with a *bigmem* job with 15G per CPU and 10 CPUs per task. It was completed in 7.8 hours and used 125.2 billing CPU hours.

4.4 Demo Figures

After completing the run, all plots are stored in the "summay_tables" subfolder under the main output folder "CutAndTagAnalyzer". In the following sections, a description and a comparison of these plots with the corresponding plots from the Zheng Y et al. (2020) protocol will be provided.

4.4.1Quality Control - FastQC Plots

As described in Chapter 3.4.1, HTML files of various quality summary figures are generated by the FastQC [35] software based on input FASTQ files. Usually, these plots can be used to check the quality of raw input files before preceding to other data preprocessing steps. FastQC provides several plots to help to evaluate the quality of a sequencing experiment. For example, a plot of "per base sequence content" is shown in Figure 4.3, where demo sample (H3K4me3_rep1_R1) from the current run and a successful reference sample are shown in the left and right panel, respectively. It shows that the reference sample has roughly straight lines across all read positions, but the plot of the demo sample displays a strong oscillation at the first 15 nucleotides for all reads. Nevertheless, this is a common phenomenon in CUT&Tag sequencing data because the pA-Tn5 recombinant protein might have preferences when cutting and tagging the DNA sequences [38]. Therefore, this potential warning or error in per base sequence content might be ignored for the current quality evaluation, and further data analysis may proceed.

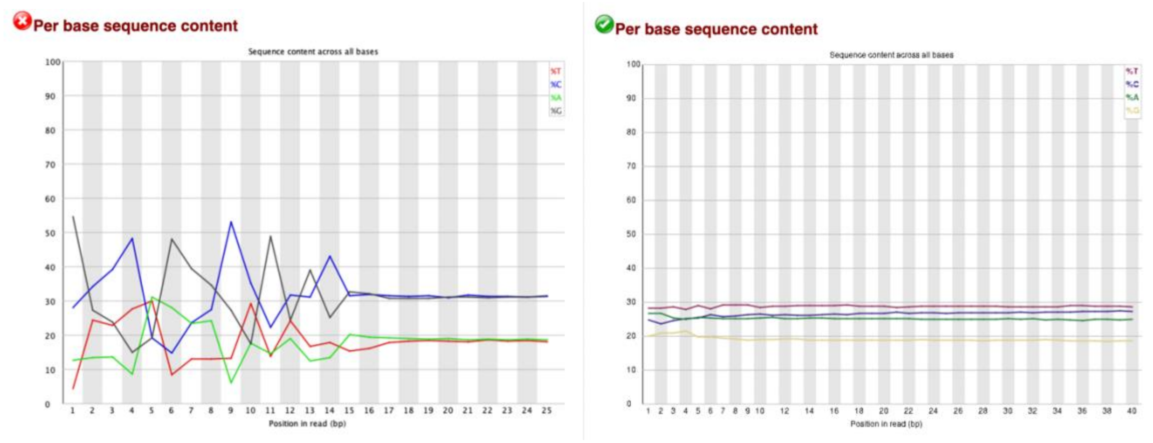


Figure 4.3 *– Per base sequence content plots from H3K4me3_rep1_R1 Cut&Tag experimental sample (left) and FastQC reference plots (right).*

<u>4.4.2 Sequencing Alignment - Bowtie2 Alignment Plots</u>

Alignment is a crucial step in high throughput sequencing analyses, which maps observed sequence reads to a reference genome for further study. Here, it is essential to evaluate the alignment quality before doing the rest of the sequencing analysis. Usually, a good sequencing experiment will have more than 80% of the input reads mapped to a reference genome. As a result of the low background noise in CUT&Tag experiments, around a million mapped reads may be sufficient for robustly profiling epigenetic marks that are in abundance, such as histone modifications. However, for less abundant targets such as transcript factors, at least ten million mapped fragments may be needed for a valid profiling [30]. Thus, roughly knowing the biological abundance of the epigenetic target will help designing an experiment (e.g., the number of cells used) to reach the data expectations.

For aforementioned reasons, a statistical summary of the alignment of reads is provided in the pipeline. At the left side of Figure 4.4, there are boxplots of sequencing depth, mapped fragments, and alignment rate for both human and spike-in (e.g., *E. coli*) genome alignment results from the proposed pipeline. These plots are compared to the results from the Zheng Y. et al (2020) protocol, which used the same input dataset (right side of Figure 4.4). In Figure 4.4 the plots from both works are identical because they used the same alignment tool - Bowtie2 [36], and the same experimental data. Furthermore, these figures indicate that all samples have good sequencing quality because their alignment rates are above 80%, with at least one million mapped fragments to the reference genome. Additionally, Figure 4.4 displays that the alignment rate of the spike-in genome is higher for control samples than that for the other (e.g., H3K27me3 and H3K4me3) samples, which is expected. However, if the epigenetic target is less abounded or has fewer cells in the experiment, then the spike-in fragments could compose as much as 70% of the total mapped reads. Nevertheless, the two histone modifications studied in this work are very common and with abundant signals, the alignment rate to *E. coli* is low as shown in Figure 4.4. Overall, the new Python package reproduced the results from original study.

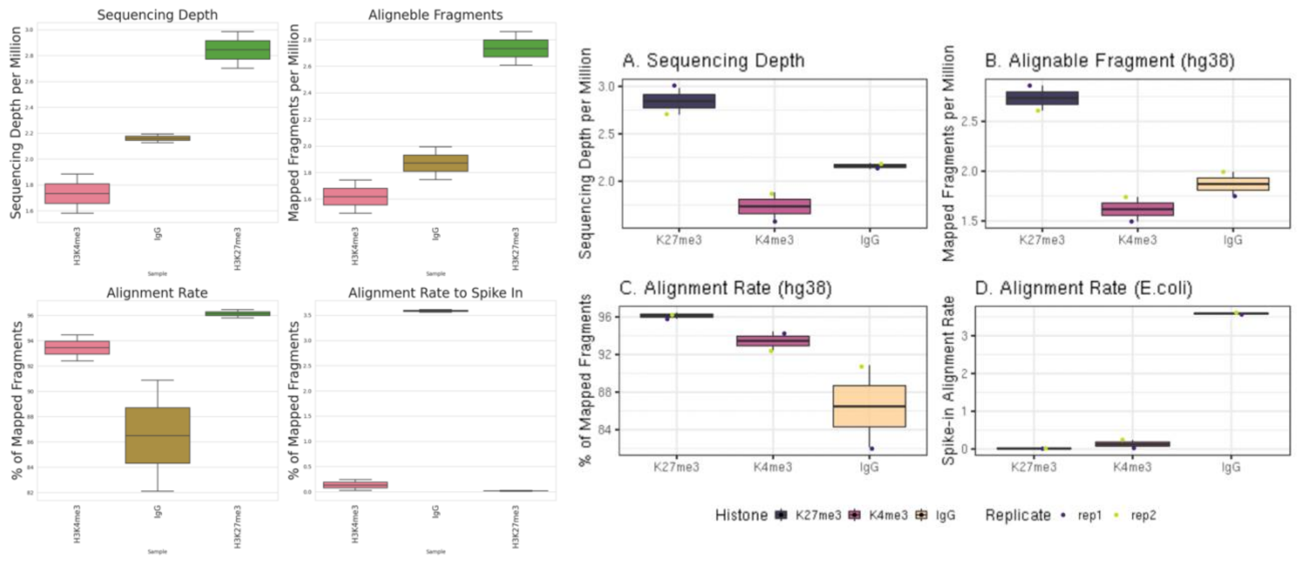


Figure 4.4 *- Boxplots of sequencing depth, alignable fragments, and alignment rate for both human and spike-in, here E. coli, genomes, from alignment summaries from the pipeline created in this thesis (left) and from the inspiration pipeline Zheng Y et al. (2020) protocol. (right)*

4.4.3 Duplication Removal After Alignment - Duplication Removal Plots

CUT&Tag data usually contain very low levels of duplications; therefore it is optional to remove duplicated reads (reads mapped to identical positions in a reference genome) in the proposed pipeline based on the Zheng Y et al. (2020) protocol. The new Python package follows the same procedure as the online protocol by using Picard [57]. Figure 4.5 shows boxplots of duplication rates from the current pipeline (left of the figure) compared to that from the Zheng Y et al.

(2020) protocol (right of the figure). Both plots are identical, which indicates the robustness of the `remove_duplicates` function in the current pipeline.

Furthermore, Figure 4.5 displays expected results from such a CUT&Tag experiment based on common histone modifications. Firstly, the duplication rates are very low for the histone modifications, yet high for the control samples (IgG). This may be explained by the non-specific tagmentation of control samples in CUT&Tag reactions. Therefore, it may not be necessary to perform duplication removal on the two histone modification samples due to the low duplication levels. Conversely, eliminating duplicates found in the IgG control samples would be appropriate before further downstream analysis because of the high duplication rate. Secondly, the estimated library size for the samples is proportional to the abundance of the targeted epitope, as well as to the quality of the antibodies that were used in the experiment. Due to the non-specialized antibody used in the control samples, there is a very small library size [4]. Lastly, the number of unique fragments found in Figure 4.5 were calculated as explained in Chapter 3.3.4, which depends on the mapped fragments and the duplication rate in sequencing. Thus, this causes H3K27me3 to have the highest number and IgG control samples to have the lowest number of unique fragments, which is expected as H3K27me3 has the highest number of mapped fragments among these samples, please refer to Figure 4.4.

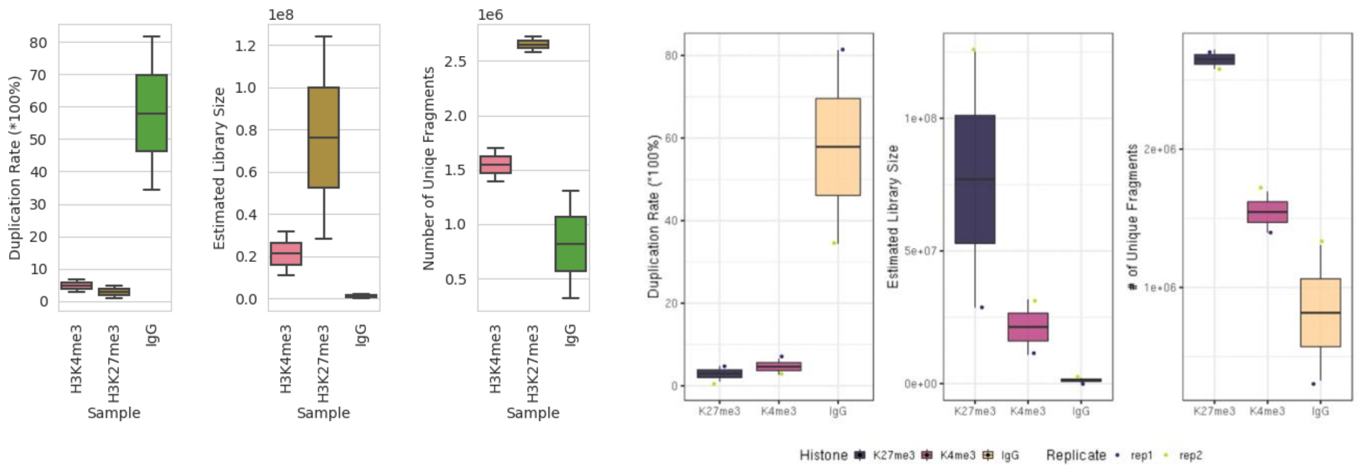


Figure 4.5 *- Boxplots of duplication rate, estimated library size, and numbers of unique fragments from duplication removal, from the pipeline created in this thesis (left) and the inspiration pipeline Zheng Y et al. (2020) protocol. (right)*

4.4.4 Mapped Fragment Lengths Plots

Figure 4.6 shows fragment length plots generated by the proposed pipeline function `fragment_length` (upper panel of the figure) and by the Zheng Y et al. (2020) protocol (lower panel of the figure). As described in Chapter 3.4.5, this function only collects the fragment length information from aligned SAM files, which is the same procedure as in the Zheng Y et al. (2020) protocol. Therefore, both works should give the same results. Indeed, there is a strong resemblance between the current work and the previous study in Figure 4.6. It confirms that `fragment_length` function works as the original design in the protocol, for the visualization of mapped fragment length distribution. Moreover, these plots show gatherings of fragment lengths around 200 bp and 400 bp. It indicates a successful CUT&Tag histone modification experiment. That is because pA-Tn5 usually tags the chromatin at either side of a chromatin particle (e.g., in this case, the nucleosome with ~180 bp), which results in slightly longer fragment lengths than the actual DNA bound by one or two nucleosomes. Nevertheless, smaller fragment sizes (50-100bp) might not indicate background noise because pA-Tn5 may tag DNA on the outside of nucleosomes and linker DNA that connects the nucleosomes. [4] Figure 4.6 shows a small peak around 50 base pairs, likely caused by tagmentation of nucleosomal DNA. Overall, the `fragment_length` function provides useful plots for exploring the fragment-length information which are related to data quality.

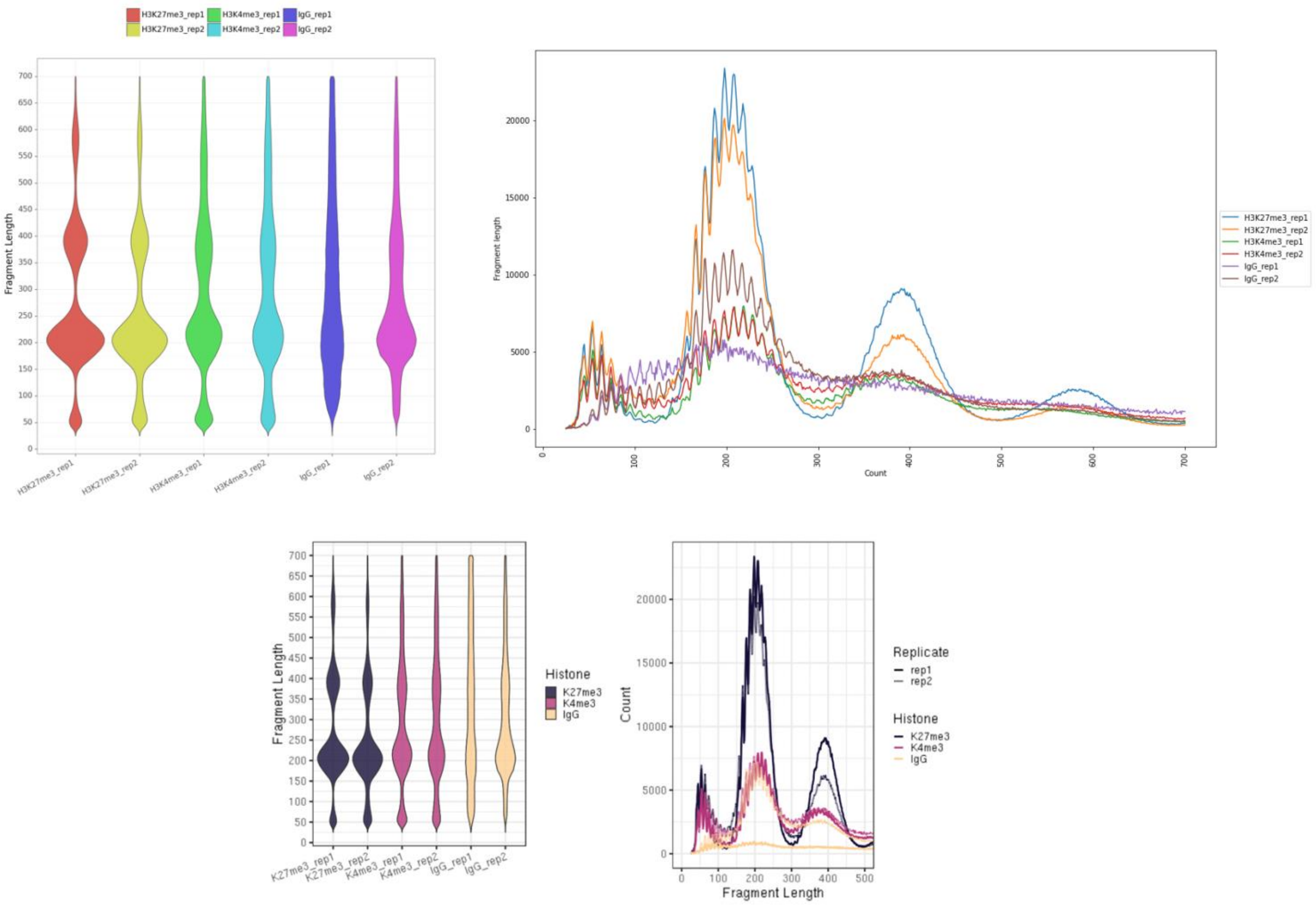


Figure 4.6 – *Violon plot and line plot of fragment lengths found in samples created by this thesis´s pipeline (top) and the inspiration pipeline Zheng Y et al. (2020) protocol (bottom)*

4.4.5 Reproducibility Plots

As described in Chapter 3.3.6, the pipeline´s function, `filtering`, evaluates the reproducibility of biological replicates, by creating heatmaps for correlation coefficients. For example, in Figure 4.7, the left side and right side are correlation coefficients plots for raw data and filtered date (e.g., reads counts <8 are removed), respectively. As expected, the correlation coefficient matrix shows better matches between the biological replicates based on the filtered data than that based on the raw data. This is because some of the background noise is eliminated by filtering the weak or close to noise signals. It is especially notable in the replicates of IgG control samples, where there is not a significant correlation before the filtering but the correlation coefficient reaches 0.56 after the filtering. This indicates that the control samples have high background noise, which is expected from a negative control. For H3K27me3 and H3K4me3 samples, their correlation coefficients are also improved after the filtering but with moderate changes. In summary, there are background noise in the biological replicates, but their overall quality are good with reproducibility please refer to Figure 4.7.

In Figure 4.8, the heatmap of correlation matrix created in the Zheng Y et al. (2020) protocol is shown. It shows better correlations between all samples, biological replicates, and across conditions when compared to the matrixes created in the current pipeline, Figure 4.7. The proposed pipeline removes blacklisted genome regions and background noise, while the Zheng Y et al. (2020) protocol does not. The expected result would be the opposite, better correlation in the pipeline than that in the original study. However it is not the case because the Zheng Y et al. (2020) protocol does not consider the overlapping of bins when calculating their correlation matrix. This is violating a statistical assumption of that assumes that all data points are independent to each other when computing the correlation coefficient. Thus, the calculation by Zheng et al (2020) amplified the statical significance, when the same/overlapping bins are repeatedly used in the correlation coefficient calculation that resulted in decreased effective sample size and false positive results. This becomes clear when considering some earlier plots: for example, Figure 4.4 shows that IgG has the highest variance of alignment rate, yet in Figure 4.8, the samples have the highest correlation. In an experiment like CUT&Tag experiment, the control samples would not be expected to have the highest correlation as they are the result of non-specific tagmentation by pA-Tn5. Thus, the proposed pipeline fixes a potential pitfall in the original analysis protocol, avoiding data redundancy and inflated significance by filtering data before calculating correlation coefficient matrixes.

In summary, the `filtering` function improves the reproducibility of biological samples, by filtering blacklisted genomic regions and background noise before computing correlation coefficient matrixes for creating heatmaps, which provides a better solution than the original protocol by Zheng Y et al. (2020).

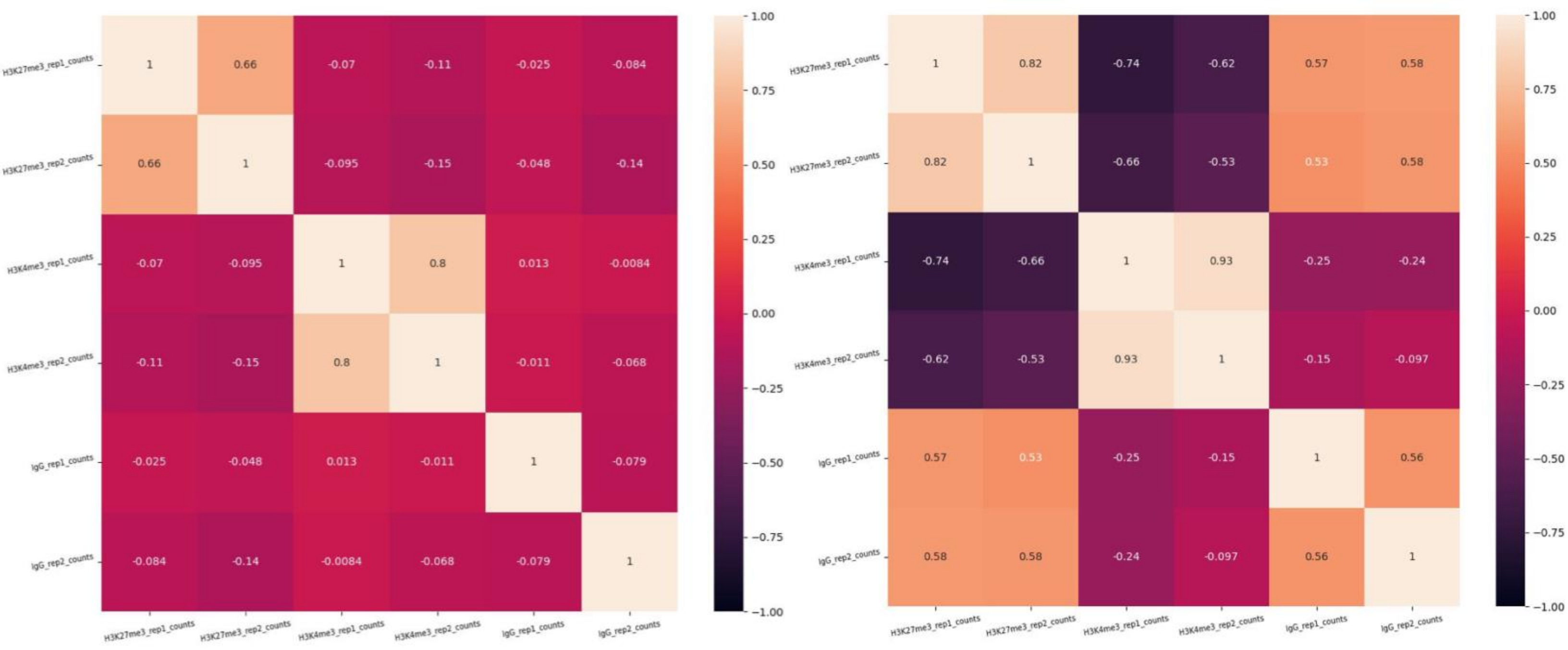


Figure 4.7 – *Correlation heatmap plots of score matrixes filtered (right) and unfiltered (left) for reproducibility assessment created by this thesis pipeline´s* `filtering` *function.*

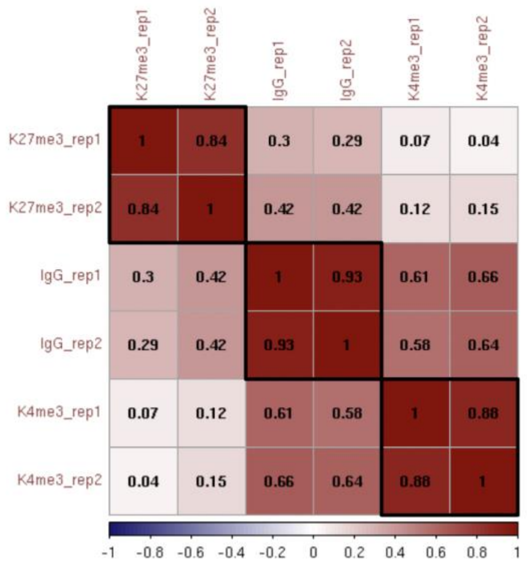


Figure 4.8 – *Correlation heatmap plot for reproducibility assessment from pipeline Zheng Y et al. (2020) protocol.*

4.4.6 Data Normalization - Scaling Factor and Normalized Fragment Count Plots

Figure 4.9 shows boxplots of scaling factors and the normalized fragment counts created by the proposed pipeline function `spike_in_calibration` and the Zheng Y et al. (2020) protocol in the left and right panels, respectively. The two plots are identical because the same spike-in calibration method is implemented in both studies, based on the alignment of reads to the spike-in genome, please refer to. Chapter 3.3.7 for more information. The likeness between the plots validates `spike_in_calibration` as an alternative to the online Zheng Y et al. (2020) protocol. The IgG samples are expected to have the lowest scaling factor, as they had the highest alignment rate to the spike-in genome. Hence, the normalized fragment count for the controls is close to zero, which indicate that they primarily consist of background noise from the spike-in genome.

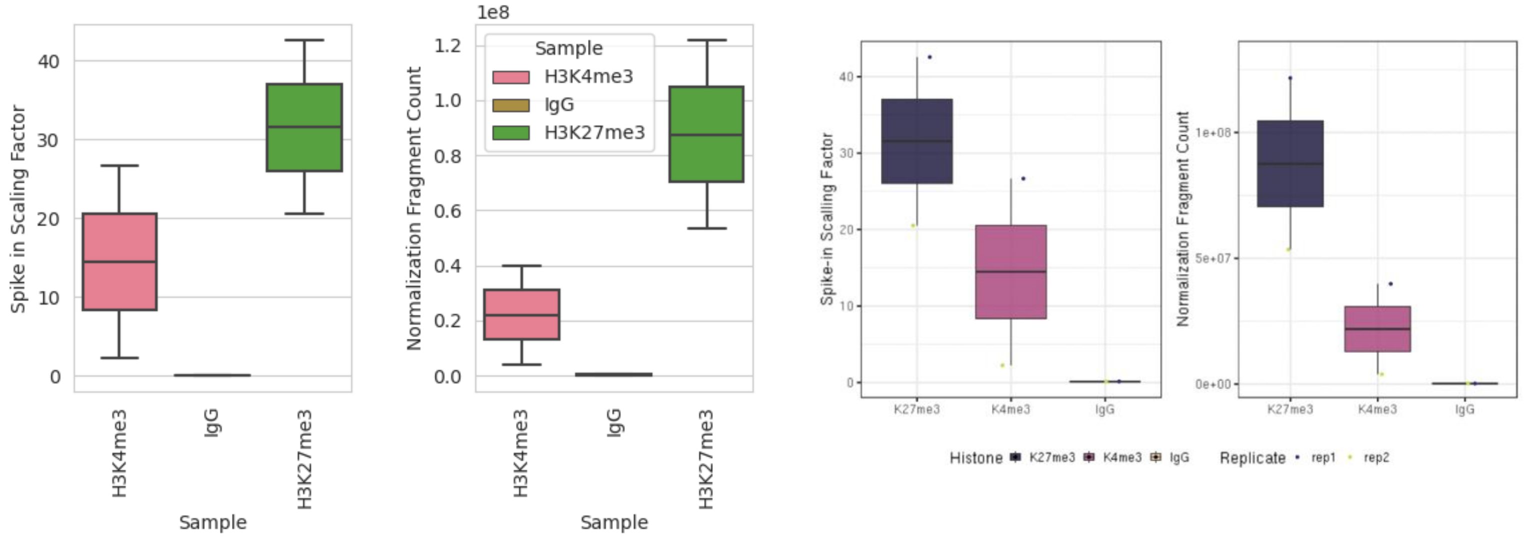


Figure 4.9 *- Boxplots of scaling factors and normalized fragment counts created by this pipeline´s function* `spike_in_calibration` *(left) and the Zheng Y et al. (2020) protocol (right).*

4.4.7 Peak Reproducibility Plots

Figure 4.10 shows the peak reproducibility plots based on the current pipeline `peak_calling` function, while Figure 4.11 displays the same plots generated by the Zheng Y et al. (2020) protocol after SEACR peak calling. The plots in both figures are the number of peak boxplots, peak width violin plots, fragment proportions in peaks (FRiPs) boxplots and peak reproduction barplots. The main peak calling with SEACR was executed similarly for both pipelines, however, the reproducibility and FRiPs were calculated differently. Here, the proposed pipeline used BEDTools [55] to find overlaps between peaks and fragments, while the Zheng Y et al. (2020) protocol used R code. Nonetheless, the two pipelines created similar plots, thus validating the calculation method used in this thesis pipeline; see Chapter 3.3.8 for details about calculations. From a biology point of view, H3K27me3 shows a higher number of peaks than H3K4me3, which is expected as it showed a higher number of alignable fragments and sequencing depth, Figure 4.4. Further, high FRiPs scores indicate low background noise as FRiPs is a measure of signal-to-noise.

In Figure 4.10, the peak width of the samples seen in the violin plot of shows a relatively wide range of peak widths. It suggests that some regions of the genome have more specific and precise binding patterns for the histone modifications. In contrast, other areas may have broader peaks and a diffuse binding pattern, where the histone modifications bind to multiple sites in a larger genomic region In the current work, both histone modifications are the most common ones in the gene regulation with broad peaks.in the gene annotation, these broad peaks may indicate specific regions activated or repressed by H3K4me3 and H3K27me3, respectively [10]. Lastly, the reproduction levels of the peaks are sensitive to the total number of peaks called in each replicate; see Chapter 3.3.8 for details. Consequently, biological replicates may have divergent reproduction percentages even though they have a similar number of overlapping peaks between them. Therefore, it is essential to consider other aspects like FRiPs when assessing the peak calling quality. In conclusion, the `peak_calling` function succeeds in SEACR peak calling and provides plots to assess the quality of peak calling results based on replicated experiments.

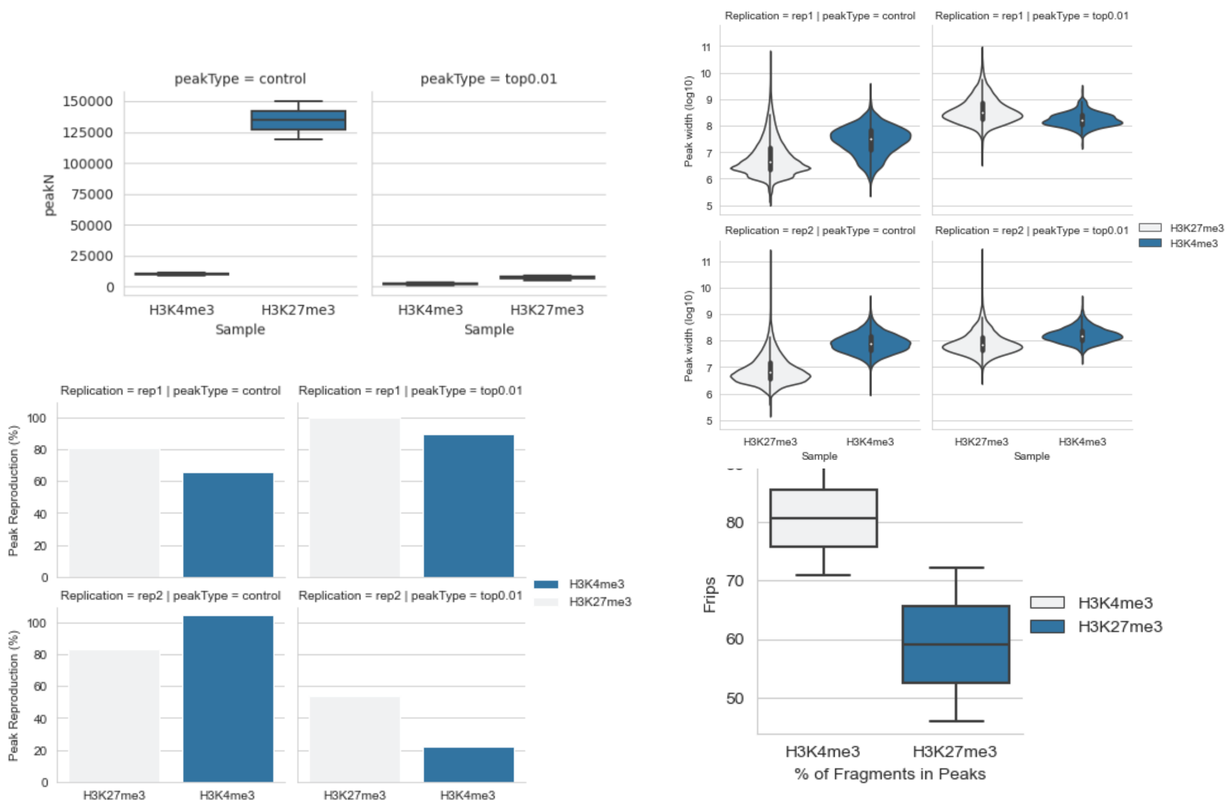


Figure 4.10 *– Plots created by this thesis´s function* `peak_calling` *during the test run: boxplot of number of peaks (top left), violin plot of log10 transformed peak width (top right), bar plot of peak reproducibility percentages (bottom left) and boxplot of FRiPs percentages (bottom right).*

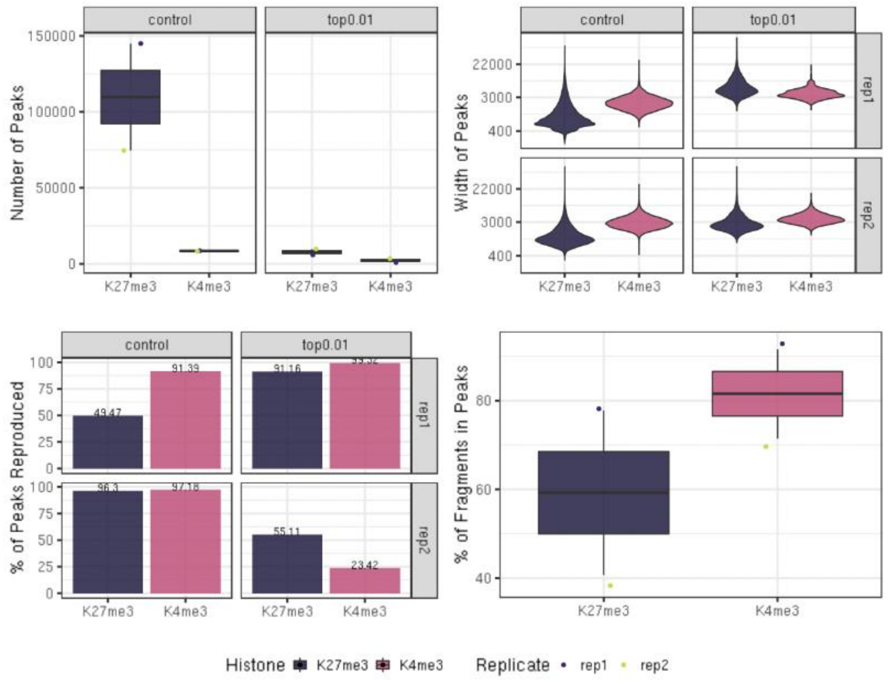


Figure 4.11 *– Plots from the Zheng Y et al. (2020) protocol: boxplot of number of peaks (top left), violin plot of log10 transformed peak width (top right), bar plot of peak reproducibility percentages (bottom left) and FRiPs percentages (bottom right).*

4.4.8 Heatmap Visualization on Specific Regions

In Figure 4.12 heatmaps of histone modification enrichment around annotated genes (e.g., from -3000 bp of TSS (Transcription start site) to +3000 bp of TES (Transcription end site)) generated by the pipeline´s `heatmap` function and the Zheng Y et al. (2020) protocol are shown in the left and right panels, respectively .Since both works used the same analysis procedures and the same data sets, their heatmaps are undisguisable. It supports that the function created in this thesis may work as a substitute for the Zheng Y et al. (2020) protocol. From a biological interpretation of the plots, it is apparent that compared to H3K27me3, the H3K4me3 modification is enriched in gene areas, which supports the known function of H3K4me3 at transcription start sites and promotors to enhance/activate gene regulation [59]. Additionally, there is a small peak of H3K27me3 at TSS , which is a common repressive mark for gene regulation [10].

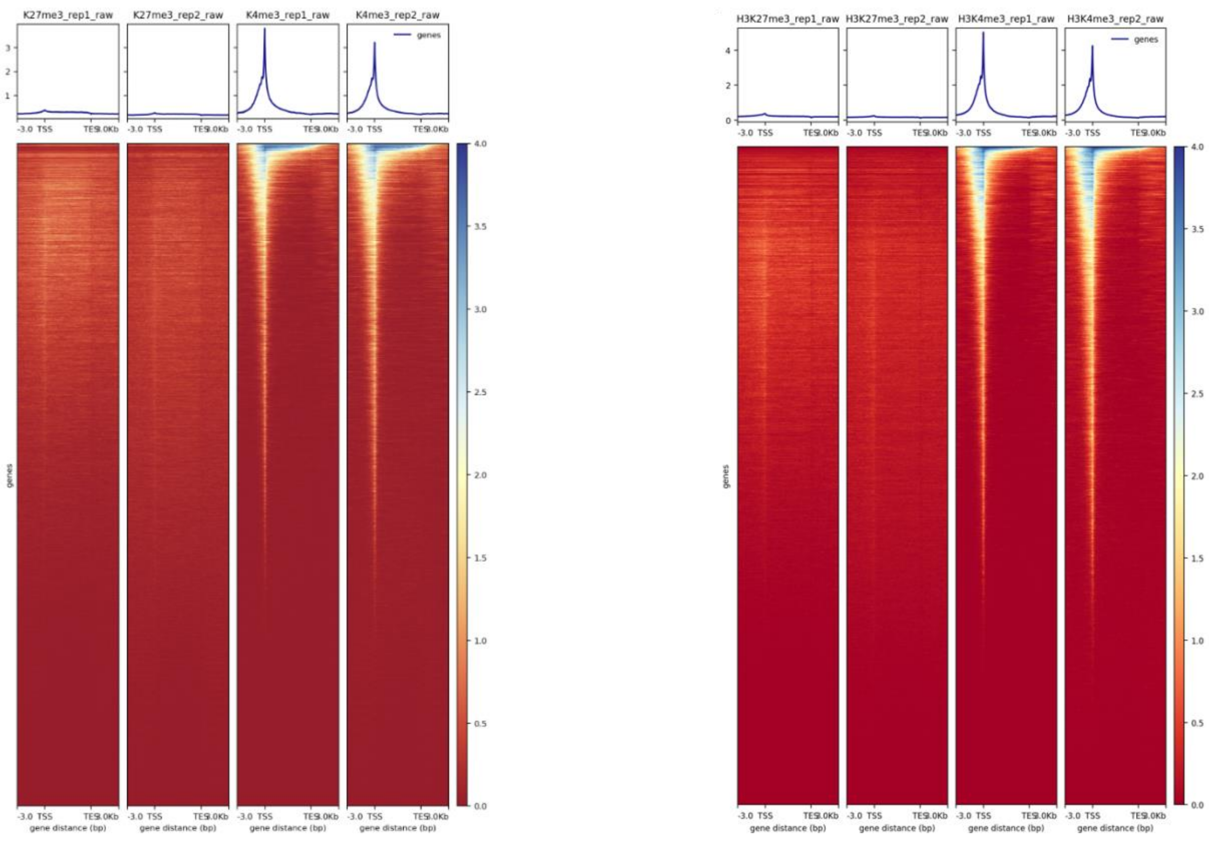


Figure 4.12 *- Heatmaps of histone enrichment around genes, created during the test run of the pipeline´s function* `heatmap` *(left) and the Zheng Y et al. (2020) protocol (right)*

Furthermore, Figure 4.13 shows the average histone enrichment around peaks generated by the current `heatmap` and the Zheng Y et al. (2020) protocol in the left and right panel, respectively. The plots from the two pipelines are also identical, thus, corroborating the success of the current pipeline function, `heatmap`. In addition, the heatmaps reveal strong enrichment of histone modifications around the called peaks found which indicates a successful CUT&Tag experiment.

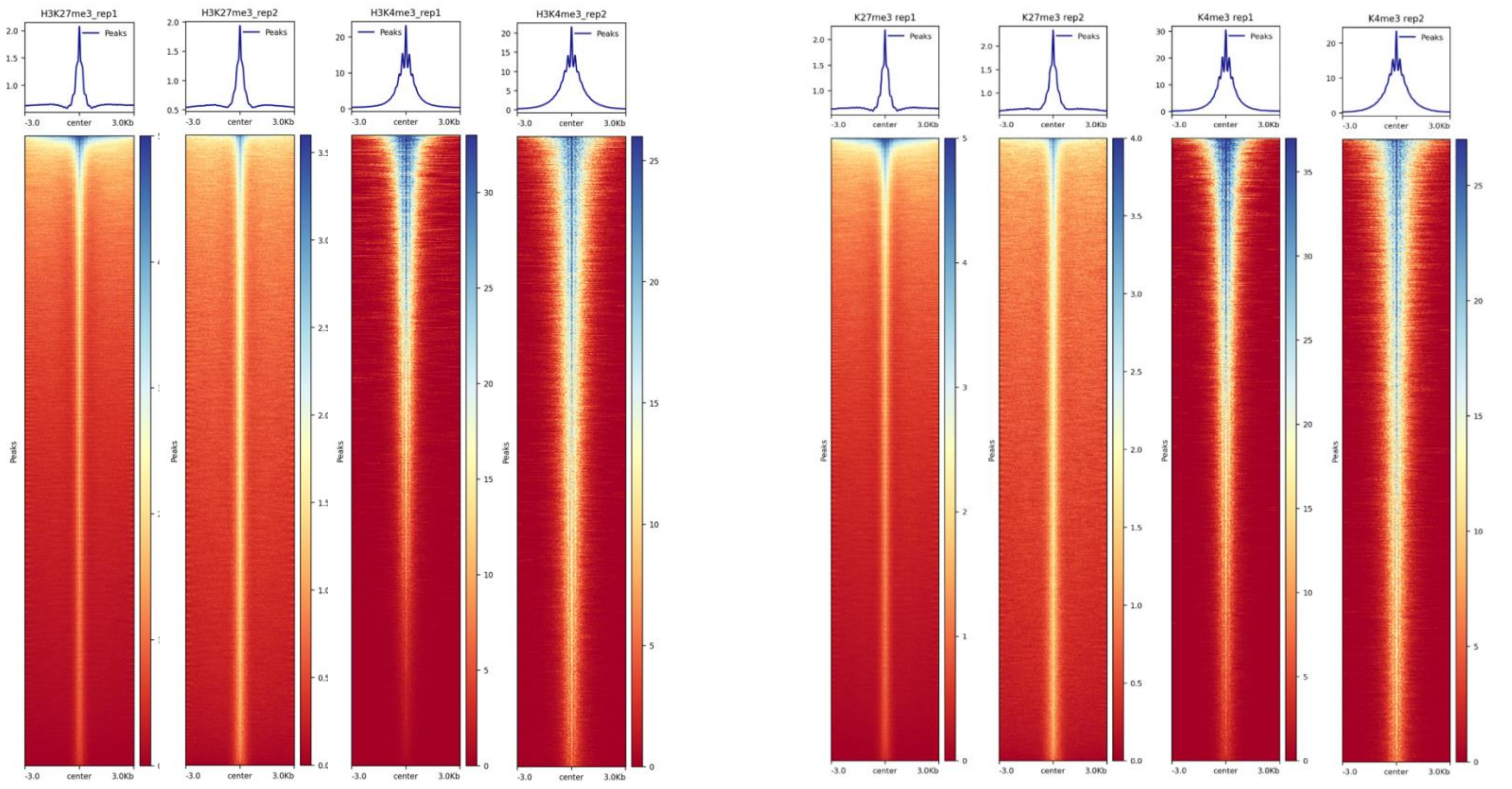


Figure 4.13 - *Heatmaps of histone enrichment around peaks, created during the test run of the pipeline´s function* `heatmap` *(left) and the Zheng Y et al. (2020) protocol (right)*

4.4.9 Differential Analysis of Called Peaks

In Figure 4.14 there are three plots generated by the pipeline function `differential_analysis`: a 3D PCA plot showing the similarity among samples based on their called peaks in predefined chromosome window binds, a pie plot for illustrating the distribution of differential peaks in five genomic regions (TSS, TES, Gene, 5'distance up regions, and enhancer regions), and a BAR plot for displaying the number of up/down differential peaks in seven genomic regions (TSS, TES, Gene, 5'distance up/down region, enhancer, intergenetic regions). These statistics summary reports and plots for analyzing differential peaks are not available in the original Zheng Y et al. (2020) protocol and are a new addition from the current pipeline. In theory, it is ideal to perform differential peak analysis between the different conditions such as healthy versus cancer cells or different tissue samples. Nevertheless, the demo datasets are two histone markers H3K4me3 and H3K27me3, representing active and repressive gene regulation, respectively.

In Figure 4.14, the bar plot suggests the differential peaks between the two histone modifications have clear opposite changing direction, with a high percentage change in various genomic regions. Since both histone modifications are often located in the promoter regions of the genome [10], the biggest difference between the is at the 5' distance Up regions, which includes the promotor area. The PCA plot in Figure 4.14 indicates that the two histone markers are well separated in a 3D-PCA plot (e.g., H3K4me3 and H3K27me3 with $PC1<0$ and $PC1>0$, respectively), though the sample size is small. The pie plot tells that the majority of differential peaks are located in the enhancer, which is an important part of the transcription regulation machinery. Thus, it is expected that the dynamical changes of two opposite histone modifications often affecting the enhancer. The second largest region with differential peaks is gene because both histone modifications are located in gene regions, especially H3K4me3 [61]. In summary, the genomic distribution of the identified differential peaks by `differential_analysis` function, matches to the functional activities of the two histone modifications. It suggests that the newly implemented differential peak analysis function in the current pipeline works well with the demo datasets. Nevertheless, more evaluation on other data sets may be needed in the future.

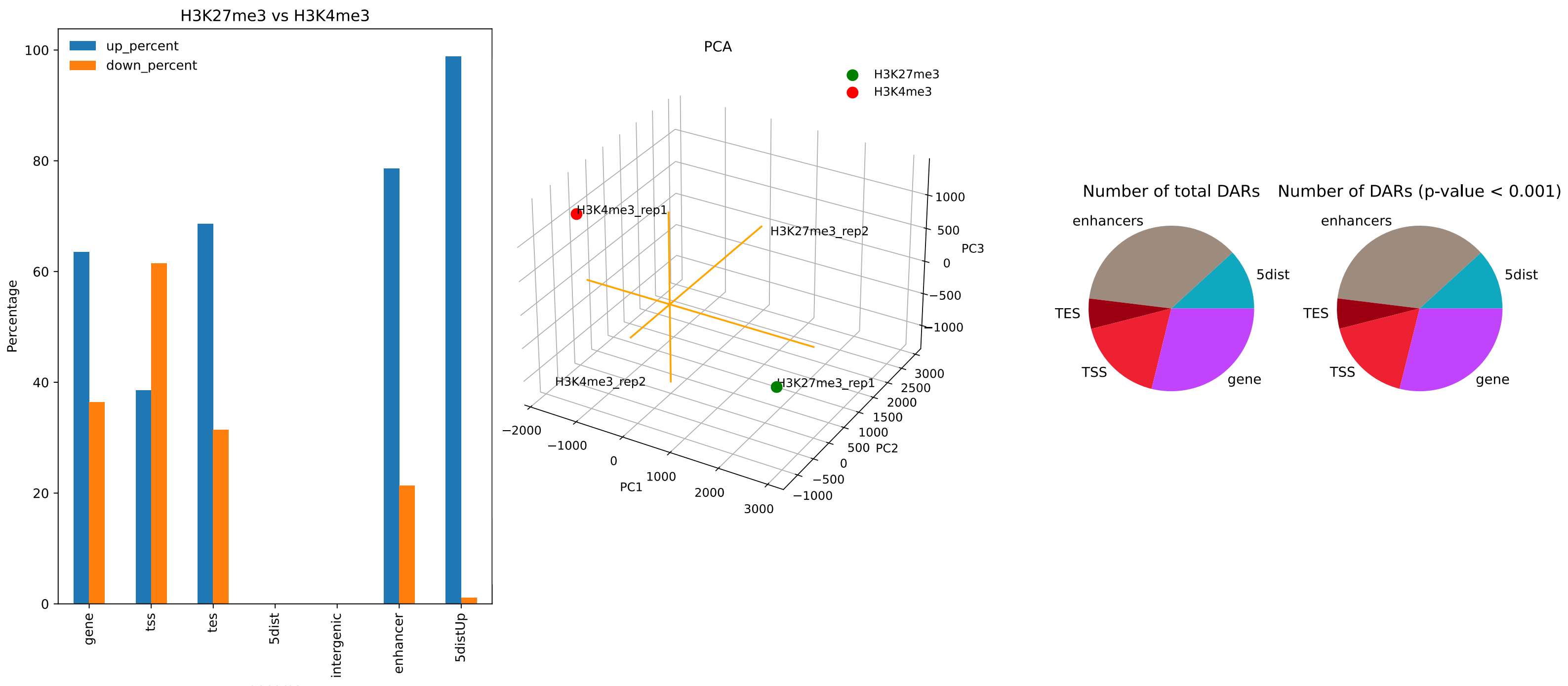


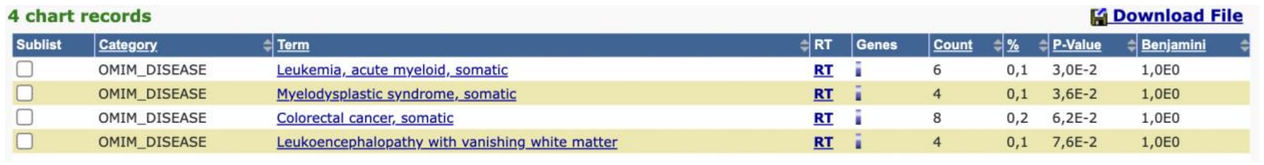
Figure 4.14 – *Plots created in the test run by this thesis differential analysis function: Bar plot of the differently annotated areas and whether they are differently up or down (left), 3D PCA plot (middle), pie plot of the differently annotated regions (right).*

4.5 Functional Evaluation of Identified Differential Peaks

To evaluate the identified differential peaks, a functional gene annotation analysis was performed to investigate the potential biological processes behind them. This is done as an example of how the current pipeline and its results can be used to find differences between biological samples, therefore also identifying differences in their gene regulation. Though an ideal application of function annotation or enrichment analysis on genes should be based on samples with different biological conditions (e.g., sick/healthy or different cell types), the results of the current demo with the two histone marks of opposite regulatory functions in the same cell line can also be used to perform such analysis. However, this demonstration provides little biological insight and should be repeated with another data set for further validation of the current pipeline.

The histone modifications examined in this trial run are H3K27me3 and H3K4me3, which are part of the gene regulation machinery, decreasing and increasing gene activity [10]. As mentioned in Chapter 3.1.4, the CUT&Tag experiments of the aforementioned two histone modifications used K562 cells, which is a cancerous leukemia blood cell line.

As described in Chapter 3.4.10, the identified differential peaks are exported to a BED format file. The transcription start sites (TSS) of genes with differential peaks were extracted, and the corresponding genes were used for functional annotation in DAVID, Database for Annotation Visualization, and Integrated Discovery, a web-accessible program that integrates functional genomic annotations with intuitive graphical summaries [62]. Table 4.5.1 lists the results of the enrichments found in the Online Mendelian Inheritance in Man, OMIM, an online database for human genes and disorders [63], where leukemia is the most significant one, and the other diseases are either a form of cancer or a blood-related illness. This result fits with the cell line (K562, leukemia blood cells) used in the CUT&Tag experiments to generate the demo data.

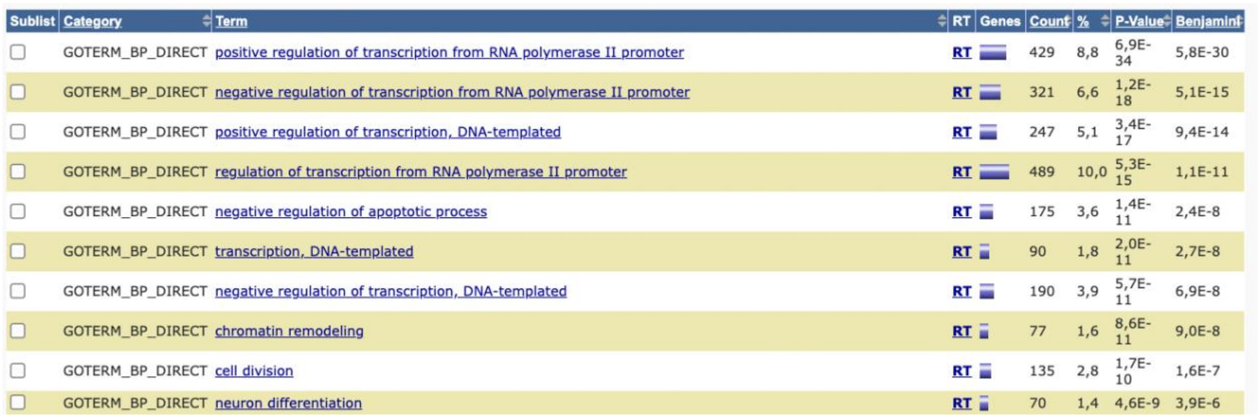


4 chart records Download File

Sublist	Category	Term	RT	Genes	Count	%	P-Value	Benjamini
☐	OMIM_DISEASE	Leukemia, acute myeloid, somatic	RT		6	0,1	3,0E-2	1,0E0
☐	OMIM_DISEASE	Myelodysplastic syndrome, somatic	RT		4	0,1	3,6E-2	1,0E0
☐	OMIM_DISEASE	Colorectal cancer, somatic	RT		8	0,2	6,2E-2	1,0E0
☐	OMIM_DISEASE	Leukoencephalopathy with vanishing white matter	RT		4	0,1	7,6E-2	1,0E0

Table 4.5.1 *– Significant enrichments of genes found in illnesses from OMIM online database [63]*

Table 4.5.2 shows the ten most enriched biological processes (BP) found in the Gene Ontology, GO, knowledgebase, the world's largest source of information on the function of genes [64, 65], based on genes with differential peaks in their promotors. Here, the top enriched GO terms are regularity functions, with the highest two being the positive and negative regulation of transcription from RNA polymerase II promotor. This is expected as the genes selected for the functional enrichment test have differential peaks (e.g., H3K27me3 vs. H3K4me3) at their promotors. Additionally, the two histone marks from the current demo are associated with either repressive or active gene expression [10], which is consistent with highly enriched biological processes (e.g., negative and positive regulation) in Table 4.5.2.



Sublist	Category	Term	RT	Genes	Count	%	P-Value	Benjamini
☐	GOTERM_BP_DIRECT	positive regulation of transcription from RNA polymerase II promoter	RT	▬	429	8,8	6,9E-34	5,8E-30
☐	GOTERM_BP_DIRECT	negative regulation of transcription from RNA polymerase II promoter	RT	▬	321	6,6	1,2E-18	5,1E-15
☐	GOTERM_BP_DIRECT	positive regulation of transcription, DNA-templated	RT	▬	247	5,1	3,4E-17	9,4E-14
☐	GOTERM_BP_DIRECT	regulation of transcription from RNA polymerase II promoter	RT	▬	489	10,0	5,3E-15	1,1E-11
☐	GOTERM_BP_DIRECT	negative regulation of apoptotic process	RT	▬	175	3,6	1,4E-11	2,4E-8
☐	GOTERM_BP_DIRECT	transcription, DNA-templated	RT	▬	90	1,8	2,0E-11	2,7E-8
☐	GOTERM_BP_DIRECT	negative regulation of transcription, DNA-templated	RT	▬	190	3,9	5,7E-11	6,9E-8
☐	GOTERM_BP_DIRECT	chromatin remodeling	RT	▬	77	1,6	8,6E-11	9,0E-8
☐	GOTERM_BP_DIRECT	cell division	RT	▬	135	2,8	1,7E-10	1,6E-7
☐	GOTERM_BP_DIRECT	neuron differentiation	RT	▬	70	1,4	4,6E-9	3,9E-6

***Table 4.5.2** - Ten most enriched gene ontologies of biological processes directly found in GO [64, 65].*

Figure 4.5.1 and supplementary Figure 1 in the appendix are the proteoglycan pathway in cancers, with the enriched genes marked with a red star, created by KEGG, Kyoto Encyclopedia of Genes and Genomes [66]. These cancer pathways are the most statistically significant one based on DAVID analysis. Figure 4.5.1 also indicates that most of the genes present in these pathways are have differential peaks at the promotors. Additionally, proteoglycans and their polyhedric nature have been implicated in having multiple functions in cancers, which may interact with ligands and receptors [67].

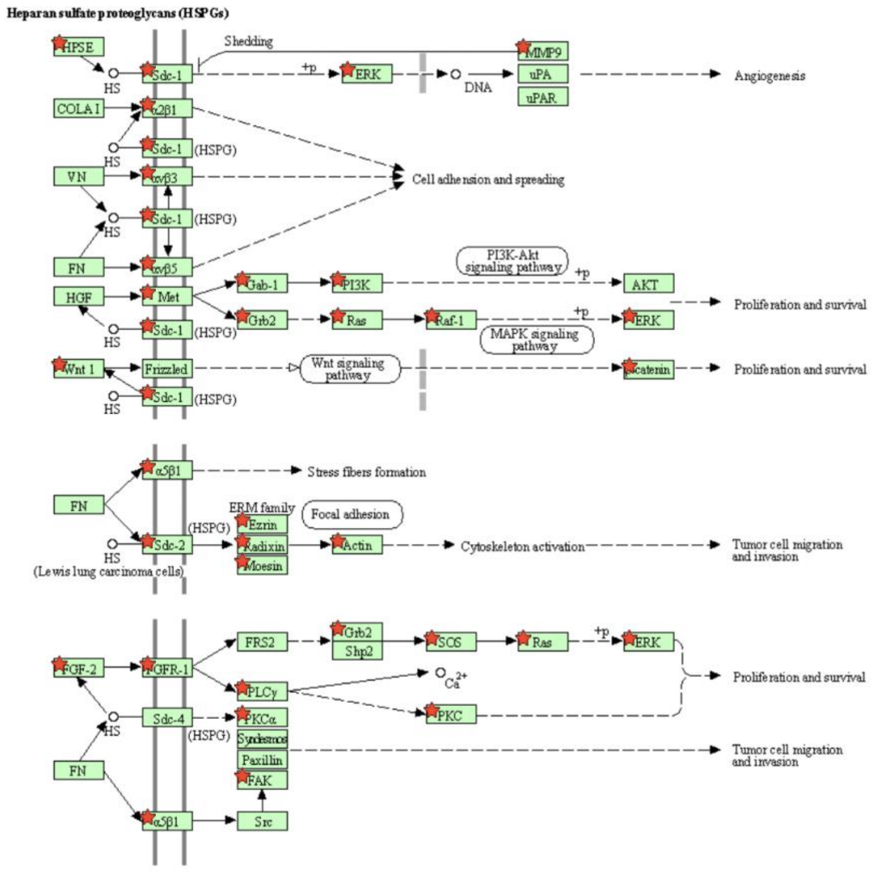


Figure 4.5.1 *– Some proteoglycan pathways found in cancers, with enriched genes marked with red stars, created with KEGG [66]. The rest of the path may be seen in Supplementary Figure 1.*

Figure 4.5.2 show the second significantly enriched pathway, the nucleocytoplasmic transport, generated by KEGG [66]. In this figure, most of the genes with differential peaks are also enriched in the pathway (e.g., marked with red stars).The nucleocytoplasmic transport pathway is essential for cancer cells as it moves proteins and other large molecules across the nuclear membrane. This transport is aberrant in cancer cells, leading to far more macromolecules being transported out of the nucleus than in normal cells. Thus, this pathway has been a target of cancer therapy discovery [68].

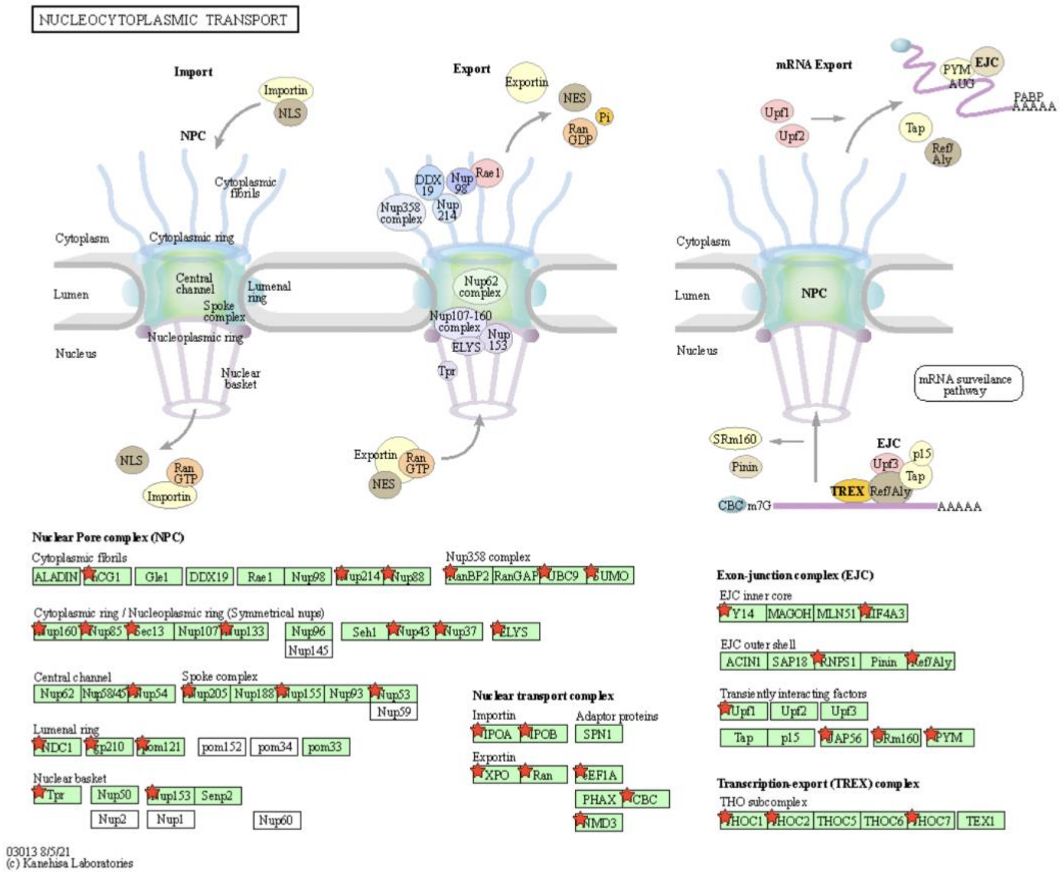


Figure 4.5.2 – *The nucleocytoplasmic transport pathway, with enriched genes marked with red stars, created with KEGG [66].*

To summarize, the functional annotation analysis of genes with differential peaks at promoter regions revealed that they are enriched in cancers, blood diseases, gene regulation ontologies, and cancerous pathways.

5. Conclusion

CUT&Tag is an epigenetic profiling sequencing technique with low input material requirements, high sensitivity, and lower background signals. Though there are pipelines for CUT&Tag data analysis available online, they require some programming skills to perform the analysis. This thesis has developed an easily installable Python package, as a user-friendly tool for analyzing CUT&Tag data for scientists without extensive programming experience. Most of the functions in the proposed Python package are based on the earlier works of Zheng Y et al. (2020) protocol available on GitHub. Additionally, some unique new functions are implemented in the proposed pipeline of this thesis, such as `differential_analysis,` which was newly created. Furthermore, reproducibility plots for assessing the correlation between biological replicates are improved to avoid false positive results caused by overlapping bins, and also a removal of blacklisted genomic regions and weak biased observations are included to improve data quality. The Python package developed in this thesis contains a fully functional pipeline for analyzing CUT&Tag sequencing data, from the low to the high level analysis such as quality control of raw sequencing reads by `fastqc`; reads alignment through `bowtie2_alignment`; data filtering with `remove_duplicates`, `fragment_length`, and `filtering`; data normalization by `spike_in_calibration`; peak prediction by `peak_calling` and `differential_analysis`. Users can run either the full pipeline or use individual functions in the package for a specific need. The package only requires the user to provide the input file names and their path, which is simple for most people.

The Python package was evaluated based on public CUT&Tag sequencing data from two common histone modifications(H3K27me3 and H3K4me3) in human K562 cells, which were also used by Zheng Y et al. (2020) in their online tutorial. The proposed pipeline successfully reproduced the plots and results of the original study [38], authenticating its further usage.

Furthermore, the results of the new function in the pipeline, differential peak analysis, are consistent with the literature information about the two histone modifications, though they cannot be compared to the original from Zheng Y et al. (2020), as that study did not contain this function. Thus, a functional annotation analysis was performed on genes associated with differential peaks identified by the pipeline´s function `differential_analysis`. These predictions are consistent with the literature information about H3K27me3 and H3K4me3 in gene regulation.

5.1 Limitations and further work

Further testing of the pipeline should be performed. Especially, concerning the last task `differential_analysis`. Here, data sets from CUT&Tag experiments of the same histone mark, but different cellular condition (e.g., sick/health or different cell types) should be used for further validation of the biological meaningfulness of the proposed pipeline.

So far the proposed pipeline is only compatible with the human genome, yet adding an option for other model organisms (e.g., mice) will make the pipeline more widely used by scientists. Further, the parameters of several external software and functions that called by the pipeline are hard coded in the package, which shall be easily accessible by users for fine tuning in the future. For example, the number of cells used in the experiment, the sequencing technique, and the target protein etc. are different for different user, which may require specific parameter adjustment in the external software or function that called by pipeline. Especially, making the pipeline compatible with other special features such as clustering of single-cell CUT&Tag experiments, will also increase its popularity. Finally, an optimization of code and a graphical user interference (GUI) may greatly improve its computational efficiency and friendliness toward many researchers.

6. References

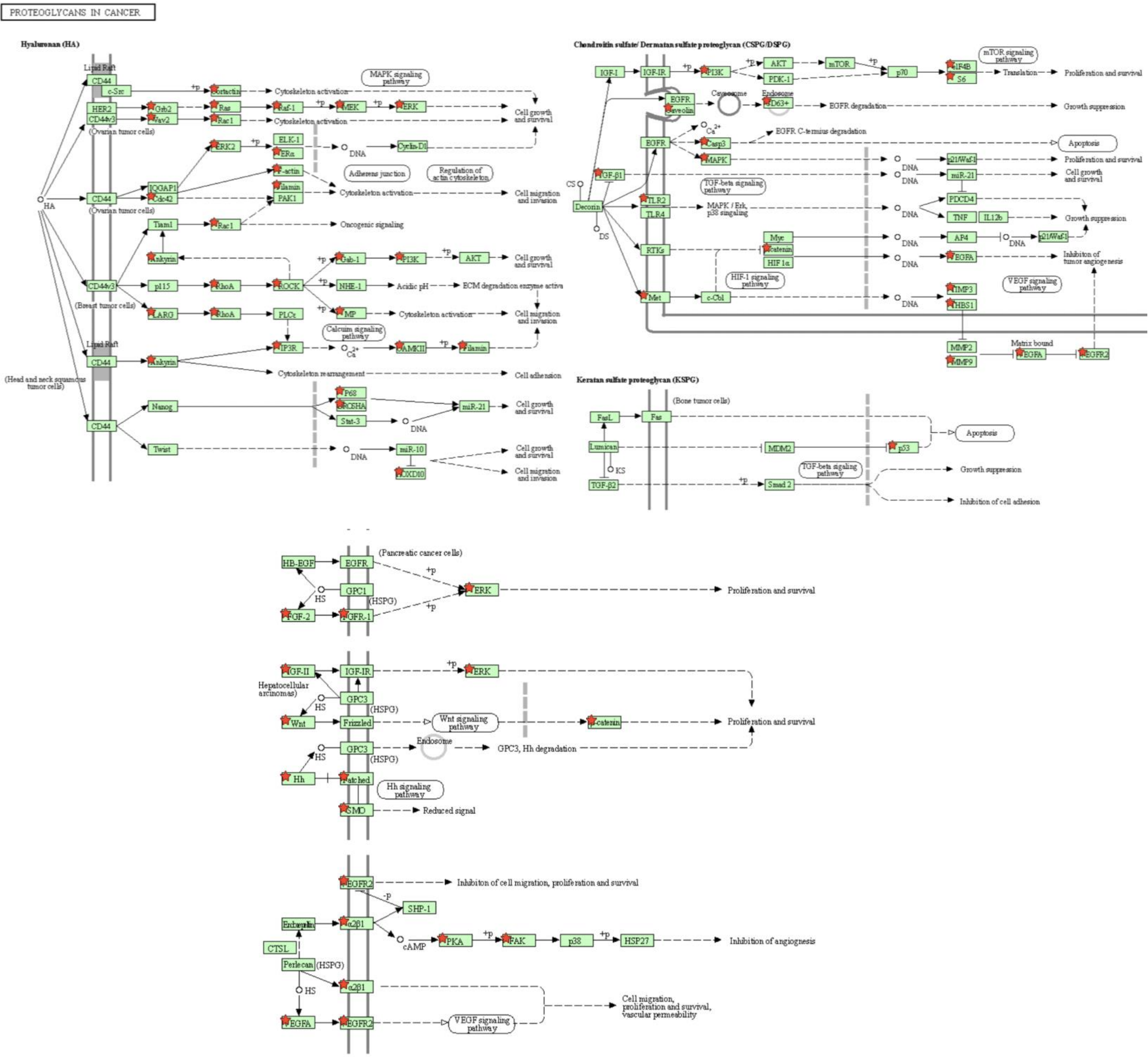
1. Villota-Salazar, N.A., A. Mendoza-Mendoza, and J.M. González-Prieto, *Epigenetics: from the past to the present.* Frontiers in Life Science, 2016. **9**(4): p. 347-370.
2. Zhang, L., Q. Lu, and C. Chang, *Epigenetics in Health and Disease*, in *Epigenetics in Allergy and Autoimmunity*, C. Chang and Q. Lu, Editors. 2020, Springer Singapore: Singapore. p. 3-55.
3. Abascal, F., et al., *Perspectives on ENCODE.* Nature, 2020. **583**(7818): p. 693-698.
4. Kaya-Okur, H.S., et al., *CUT&Tag for efficient epigenomic profiling of small samples and single cells.* Nat Commun, 2019. **10**(1): p. 1930-1930.
5. Kaya-Okur, H.S., et al., *Efficient low-cost chromatin profiling with CUT&Tag.* Nature Protocols, 2020. **15**(10): p. 3264-3283.
6. Jaenisch, R. and A. Bird, *Epigenetic regulation of gene expression: how the genome integrates intrinsic and environmental signals.* Nature Genetics, 2003. **33**(3): p. 245-254.
7. Shuey, D.J., D.E. McCallus, and T. Giordano, *RNAi: gene-silencing in therapeutic intervention.* Drug Discov Today, 2002. **7**(20): p. 1040-6.
8. Lambert, S.A., et al., *The Human Transcription Factors.* Cell, 2018. **172**(4): p. 650-665.
9. Bannister, A.J. and T. Kouzarides, *Regulation of chromatin by histone modifications.* Cell Research, 2011. **21**(3): p. 381-395.
10. Greer, E.L. and Y. Shi, *Histone methylation: a dynamic mark in health, disease and inheritance.* Nat Rev Genet, 2012. **13**(5): p. 343-57.
11. Abcam, *Histone modifications.* 2023.
12. Roth, S.Y., J.M. Denu, and C.D. Allis, *Histone acetyltransferases.* Annu Rev Biochem, 2001. **70**: p. 81-120.
13. Rossetto, D., N. Avvakumov, and J. Côté, *Histone phosphorylation: a chromatin modification involved in diverse nuclear events.* Epigenetics, 2012. **7**(10): p. 1098-108.
14. Kschonsak, M. and C.H. Haering, *Shaping mitotic chromosomes: From classical concepts to molecular mechanisms.* Bioessays, 2015. **37**(7): p. 755-66.
15. Cao, J. and Q. Yan, *Histone ubiquitination and deubiquitination in transcription, DNA damage response, and cancer.* Front Oncol, 2012. **2**: p. 26.
16. Rape, M., *Ubiquitylation at the crossroads of development and disease.* Nature Reviews Molecular Cell Biology, 2018. **19**(1): p. 59-70.
17. Ramazi, S., A. Allahverdi, and J. Zahiri, *Evaluation of post-translational modifications in histone proteins: A review on histone modification defects in developmental and neurological disorders.* Journal of Biosciences, 2020. **45**(1): p. 135.
18. Castro-Mondragon, J.A., et al., *JASPAR 2022: the 9th release of the open-access database of transcription factor binding profiles.* Nucleic Acids Research, 2021. **50**(D1): p. D165-D173.
19. Hainer, S.J. and T.G. Fazzio, *High-Resolution Chromatin Profiling Using CUT&RUN.* Curr Protoc Mol Biol, 2019. **126**(1): p. e85.
20. Park, P.J., *ChIP–seq: advantages and challenges of a maturing technology.* Nature Reviews Genetics, 2009. **10**(10): p. 669-680.

21. Bailey, T., et al., *Practical guidelines for the comprehensive analysis of ChIP-seq data.* PLoS Comput Biol, 2013. **9**(11): p. e1003326.
22. Tiago Faial, *ChIP–seq captures the chromatin landscape.* 2021.
23. Barski, A., et al., *High-Resolution Profiling of Histone Methylations in the Human Genome.* Cell, 2007. **129**(4): p. 823-837.
24. Tiago Faial, *Chip-seq captures the chromatin landscape.* Natrue, milestone, 2021.
25. Ma, S. and Y. Zhang, *Profiling chromatin regulatory landscape: insights into the development of ChIP-seq and ATAC-seq.* Molecular Biomedicine, 2020. **1**(1): p. 9.
26. Landt, S.G., et al., *ChIP-seq guidelines and practices of the ENCODE and modENCODE consortia.* Genome Res, 2012. **22**(9): p. 1813-31.
27. Skene, P.J. and S. Henikoff, *An efficient targeted nuclease strategy for high-resolution mapping of DNA binding sites.* eLife, 2017. **6**: p. e21856.
28. Kaya-Okur, H.S., et al., *Efficient low-cost chromatin profiling with CUT&Tag.* Nat Protoc, 2020. **15**(10): p. 3264-3283.
29. Anne-Sophie Ay-Berthomieu, P.D., *Comprehensive Guide to Understanding and Using CUT&Tag Assays*. 2020: Active Motif.
30. Chen, K., et al., *The Overlooked Fact: Fundamental Need for Spike-In Control for Virtually All Genome-Wide Analyses.* Mol Cell Biol, 2015. **36**(5): p. 662-7.
31. Amemiya, H.M., A. Kundaje, and A.P. Boyle, *The ENCODE Blacklist: Identification of Problematic Regions of the Genome.* Scientific Reports, 2019. **9**(1): p. 9354.
32. Meers, M.P., D. Tenenbaum, and S. Henikoff, *Peak calling by Sparse Enrichment Analysis for CUT&RUN chromatin profiling.* Epigenetics & Chromatin, 2019. **12**(1): p. 42.
33. Chen, Y., S. Chen, and E.P. Lei, *DiffChIPL: a differential peak analysis method for high-throughput sequencing data with biological replicates based on limma.* Bioinformatics, 2022. **38**(17): p. 4062-4069.
34. Nakato, R. and K. Shirahige, *Recent advances in ChIP-seq analysis: from quality management to whole-genome annotation.* Brief Bioinform, 2017. **18**(2): p. 279-290.
35. Andrews, S., *FastQC: a quality control tool for high throughput sequence data*. 2010, Babraham Bioinformatics, Babraham Institute, Cambridge, United Kingdom.
36. Langmead, B. and S.L. Salzberg, *Fast gapped-read alignment with Bowtie 2.* Nat Methods, 2012. **9**(4): p. 357-9.
37. Ramírez, F., et al., *deepTools: a flexible platform for exploring deep-sequencing data.* Nucleic Acids Research, 2014. **42**(W1): p. W187-W191.
38. Zeng, Y., Ahmad, K., Henikoff, S., *CUT&Tag Data Processing and Analysis Tutorial.* Protocol.io, 2020.
39. Yu, F., V.G. Sankaran, and G.C. Yuan, *CUT&RUNTools 2.0: a pipeline for single-cell and bulk-level CUT&RUN and CUT&Tag data analysis.* Bioinformatics, 2021. **38**(1): p. 252-254.
40. Zhao, Y., L. Wong, and W.W.B. Goh, *How to do quantile normalization correctly for gene expression data analyses.* Scientific Reports, 2020. **10**(1): p. 15534.
41. Ma, S. and Y. Dai, *Principal component analysis based methods in bioinformatics studies.* Briefings in Bioinformatics, 2011. **12**(6): p. 714-722.
42. Daniel, W.W.a.C., L. Chad, *BIOSTATISTICS - A Foundation for Analysis in the Health Sciences.* 2013. **10th edtition** p. 215-220, 236-242.
43. (USA), N.L.o.M., *National Center for Biotechnology Information* 1988.

44. Kent, W.J., et al., *The human genome browser at UCSC.* Genome research, 2002. **12**(6): p. 996-1006.
45. Edgar, R., M. Domrachev, and A.E. Lash, *Gene Expression Omnibus: NCBI gene expression and hybridization array data repository.* Nucleic Acids Res, 2002. **30**(1): p. 207-10.
46. Van Rossum, G. and F.L. Drake, *Python 3 Reference Manual:(Python Documentation Manual Part 2).* Scotts Valley, CA: CreateSpace, 2009.
47. Harris, C.R., et al., *Array programming with NumPy.* Nature, 2020. **585**(7825): p. 357-362.
48. Virtanen, P., et al., *SciPy 1.0: fundamental algorithms for scientific computing in Python.* Nature Methods, 2020. **17**(3): p. 261-272.
49. McKinney, W. *Data structures for statistical computing in python*. in *Proceedings of the 9th Python in Science Conference*. 2010. Austin, TX.
50. Farooq, A., et al., *HMST-Seq-Analyzer: A new python tool for differential methylation and hydroxymethylation analysis in various DNA methylation sequencing data.* Comput Struct Biotechnol J, 2020. **18**: p. 2877-2889.
51. Waskom, M., et al., *mwaskom/seaborn: v0. 8.1 (September 2017), Zenodo.* Available at: doi, 2017. **10**.
52. Hunter, J.D., *Matplotlib: A 2D Graphics Environment.* Computing in Science & Engineering, 2007. **9**(3): p. 90-95.
53. Kibrige, H., *plotnine.* 2023 . https://github.com/has2k1/plotnine
54. Danecek, P., et al., *Twelve years of SAMtools and BCFtools.* GigaScience, 2021. **10**(2).
55. Quinlan, A.R. and I.M. Hall, *BEDTools: a flexible suite of utilities for comparing genomic features.* Bioinformatics, 2010. **26**(6): p. 841-842.
56. Meers, M.P., D. Tenenbaum, and S. Henikoff, *Peak calling by Sparse Enrichment Analysis for CUT&RUN chromatin profiling.* Epigenetics & chromatin, 2019. **12**: p. 1-11.
57. Institute, B., *Picard Tools.* Broad Institute, GitHub repository, 2019.
58. Quinlan, A.R. and I.M. Hall, *BEDTools: a flexible suite of utilities for comparing genomic features.* Bioinformatics, 2010. **26**(6): p. 841-2.
59. Howe, F.S., et al., *Is H3K4me3 instructive for transcription activation?* Bioessays, 2017. **39**(1): p. 1-12.
60. Grüning, B., et al., *Bioconda: sustainable and comprehensive software distribution for the life sciences.* Nature Methods, 2018. **15**(7): p. 475-476.
61. Zhao, X.D., et al., *Whole-genome mapping of histone H3 Lys4 and 27 trimethylations reveals distinct genomic compartments in human embryonic stem cells.* Cell stem cell, 2007. **1**(3): p. 286-298.
62. Dennis, G., et al., *DAVID: Database for Annotation, Visualization, and Integrated Discovery.* Genome Biology, 2003. **4**(9): p. R60.
63. Amberger, J.S., et al., *OMIM.org: Online Mendelian Inheritance in Man (OMIM®), an online catalog of human genes and genetic disorders.* Nucleic Acids Research, 2014. **43**(D1): p. D789-D798.
64. Ashburner, M., et al., *Gene ontology: tool for the unification of biology. The Gene Ontology Consortium.* Nat Genet, 2000. **25**(1): p. 25-9.
65. Consortium, T.G.O., et al., *The Gene Ontology knowledgebase in 2023.* Genetics, 2023. **224**(1).
66. Kanehisa, M. and S. Goto, *KEGG: kyoto encyclopedia of genes and genomes.* Nucleic acids research, 2000. **28**(1): p. 27-30.

67. Iozzo, R.V. and R.D. Sanderson, *Proteoglycans in cancer biology, tumour microenvironment and angiogenesis.* Journal of cellular and molecular medicine, 2011. **15**(5): p. 1013-1031.
68. Gravina, G.L., et al., *Nucleo-cytoplasmic transport as a therapeutic target of cancer.* Journal of hematology & oncology, 2014. **7**(1): p. 1-9.

8. Appendix


PROTEOGLYCANS IN CANCER
Hyaluronan (HA)
Chondroitin sulfate/ Dermatan sulfate proteoglycan (CSPG/DSPG)
Keratan sulfate proteoglycan (KSPG)
Lipid Raft
CD44
c-Src
HER2
CD44v3
(Ovarian tumor cells)
(Breast tumor cells)
(Head and neck squamous tumor cells)
HA
MAPK signaling pathway
Cytoskeleton activation
Cell growth and survival
Cell migration and invasion
Oncogenic signaling
Acidic pH
ECM degradation enzyme activa
Cell adhesion
Cytoskeleton rearrangement
Proliferation and survival
Translation
Growth suppression
EGFR degradation
EGFR C-termius degradation
Apoptosis
mTOR signaling pathway
TGF-beta signaling pathway
HIF-1 signaling pathway
VEGF signaling pathway
Inhibiton of tumor angiogenesis
Matrix bound
(Bone tumor cells)
Inhibition of cell adhesion
(Pancreatic cancer cells)
Hepatocellular arcinomas)
Wnt signaling pathway
GPC3, Hh degradation
Hh signaling pathway
Reduced signal
Inhibiton of cell migration, proliferation and survival
Inhibition of angiogenesis
Cell migration, proliferation and survival, vascular permeability

***Supplementary Figure 1 –** Some proteoglycan pathways found in cancer cells, with enriched genes marked with red stars, created with KEGG [66]. The rest of the paths may be viewed in Figure 4.5.1.*