Novel approaches to the analysis of the spatial patterns in the bisulfite DNA sequencing data

Theodoulos Rodosthenous

5th September 2017

Imperial College London
MSc in Statistics project
Supervised by: Dr Vahid Shahrezaei
External Partner: Dr Petra Hajkova (MRC LMS)

The work contained in this thesis is my own work unless otherwise stated.

Signature:…….

Date:……

Abstract

DNA methylation (the presence of the chemical modification of deoxycytidine to 5methyl-deoxycitidine in DNA) has been shown to play an important role in regulation of gene expression in both physiological (cell differentiation, development) and pathological (cancer) conditions.

The presence and abundance of DNA methylation is typically assessed by bisulfite sequencing (a variant of DNA sequencing), which involves: 1) bisulfite conversion - chemical reaction converting all unmodified cytisines (Cs) to uracils (Us) while the modified 5methylcytidines (5mC) remain unconverted; 2) amplification of the converted DNA and 3) sequencing. The presence of cytosines in the obtained sequence indicates the presence of 5mC in the original analysed DNA strand. Importantly, each obtained sequence represents information about the modification status of an individual original DNA molecule present in the sample and thus provide information not only about the total abundance of this cytosine modification, but also about the exact position of this modified base, the relative distribution along the DNA strand and finally, about the existence and the abundance of the various modification patterns in the original population of cells.

Despite this richness of the information provided by the bisulfite sequencing approach, the existing data analysis pipelines are usually limited to the quantification of the relative presence of 5mC vs unmodified cytosines in the "window" of analysed sequence. Other statistical approaches such as LASSO Regression, PCA and various clustering algorithms will be implemented in the DNA Methylation data.

Some promising results have been produced, which represent some specific segments with patterns that could potentially separate the two groups in comparison. The segments of which the data were found to be "identical" between the two groups were analysed further. Even so, further investigation and work must take place in order to achieve more efficient and significant results.

Table of Contents

1 Introduction

Nowadays, statisticians are involved with projects in almost every scientific field and even in most industrial sectors. From investing, to optimising marketing schemes, more and more people are using statistical approaches and their applications to harness their data.

As technology advances, analysing huge volumes of data becomes more and more viable. The ability to analyse Big Data has opened new doors and opportunities in obtaining potentially interesting and significant outcomes. An important application of such analysis lies in Biology, a scientific field that cannot stop growing and that deals with large data-sets (e.g. DNA data).

Many of the known achievements in Biology and Genetics depend on the most recently developed statistical methods and models that could only efficiently work with the ever-growing technology. Deep Learning and other statistical algorithms are used to make sense of Big Data. Even though, Deep learning is not considered in this project, it could be implemented in the future(See section 6).

Besides Genetics, some researchers have moved on to work on Epigenetics, another sub-field of Biology. Little is still known about it and that produces an additional reason behind its attractiveness and its recent interest [1, 2, 5, 14, 15, 16]. In this project, DNA Methylation data will be analysed with the purpose of achieving a better and clearer knowledge on this matter.

A big challenge would arise while trying to fully comprehend the biological and statistical meaning of the data. After-all, statistical significance doesn't necessarily mean biological significance[1]. Cleaning, slicing and taking only the necessary variables is a time-consuming task. Difficulty also lies in implementing and applying various relevant statistical methods in large data-sets such as the ones used in this project.

A collection of data to be analysed will be given to us from Dr Petra Hajkova from the Medical Research Council (MRC) and the London Institute of Medical Sciences (LMS) [2]. The data are taken from mice of two different types (KO and Wild) on three separate Embryonic days (E10, E12, E14).

The primary goal is to identify approaches to separate the mice of KO type from those with Wild type. However, finding notable differences in the patterns between females and males, or between Embryonic days would also be beneficial and could open doors for further investigation. Even so, we will focus on analysing the KO and Wild type data-sets acting as the two groups in comparison.

In short, the plan for the analysis in this project is as follows: First the data would need to be transformed into the necessary format. Wilcoxon Signed-Rank test will be adapted in order to separate the instances (segments) for which the data seem identical from the rest. We are interested in the data that Wilcoxon test fails to identify a significant difference between the two data-sets. Further investigation and analysis will take place on the segments of DNA which have shown that the two groups have identical data. Hence, we are expecting limited (if any) significant results to be found.

After the reduction of data of interest, LASSO regression will be implemented. Other shrinkage regression models will be considered, but we will focus on the $\boldsymbol{l_1}$ penalisation(LASSO) [18]. Care must be taken when dealing with the sample sizes in LASSO regression. Over-sampling will be used, due to the imbalanced sample sizes between the data-sets since otherwise, the prediction rates would be misleading.

Next step, would be to focus on the segments that LASSO regression has shown potential in obtaining interesting results. These segments are expected to be be very limited. Unsupervised clustering approaches will be used in order to visualise the data and identify clusters that separate the two data-sets. Being able to find a separation through PCA [8] or K-means and K-medoids [12] approaches would be beneficial with

potential for deeper investigation.

Finally, a type of network will be introduced with the sole purpose to improve the visualisation of the data. Through the graph, one would be able to identify differences between patterns in a specific segment of DNA Methylated data, i.e. to identify segments with the potential of investigating further.

Some of the results will be disussed in the section 5. Our hopes in obtaining significant results were limited and that is confirmed by the very small number of promising segments found. Out of more than 150 segments taken into consideration, only 5 have shown interesting results which would be worth investigating further and closer.

Some suggestions on improving this analysis performed will be made in the last section of this report.

2 Basics of Biology

It is essential to have some basic knowledge of Biology in order to fully comprehend what follows. As previously mentioned, a statistical analysis will take place on bisulfite DNA sequencing data. In order to understand what the data exactly involve and more importantly what the purpose of the project is, some basic definitions of biological terms will be introduced.

2.1 Useful terms

This project lies in the sub-field of Biology, called Epigenetics. Before moving on with jargon regarding the Epigenetics field, let us take a step back and introduce some basic, yet important biological terms [3].

DNA: (Deoxyribonucleic Acid) A molecule in all living organisms, that carries the necessary genetic information. DNA is unique and most DNA consists of two strands, coiled around each other to form a double helix. It is also worth mentioning that, it is made up of an immense number of nucleotide bases (human DNA is made up of approximately 3 billion bases).

The four fundamental types of nucleotide bases are the Adenine(A), Thymine(T), Guanine(G) and Cytosine(C). The two DNA strands are complementary to one another. In particular, Adenine compliments Thymine and Cytosine complements Guanine as shown in the Figure 1.

Chromosome: A DNA strand encoded with genes. Humans have 22 pairs of chromosomes plus the two sex chromosomes (XX for females or XY for males).

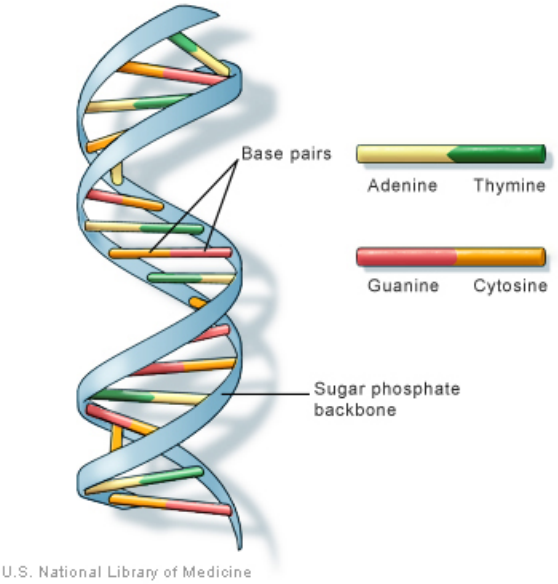


Figure 1: The structure of the DNA. Image provided by the Genetics Home reference of the U.S. National Library of Medicine.

Genome: An organism's complete set of DNA, including all of its genes.

Reference Genome: A digital succession of letters that indicate the order of nucleotides within a DNA molecule, assembled as a representative example of a species' set of genes. They do not accurately represent the set of genes of any particular single organism.

DNA Sequence: Process of determining the precise order of nucleotides within a DNA molecule.

Single-end: Sequencing DNA from only one end. This solution delivers unprecedented volumes of high-quality data and it is often used when we have a reference genome.

Paired-end: Sequencing that allows users to sequence both ends of a fragment and generate high-quality, alignable sequence data.

Note: That data in this project were single-end sequenced.

Sequence Assembly: Refers to aligning and merging fragments from a longer DNA sequence in order to reconstruct the original sequence. This is needed as DNA sequencing technology cannot read whole genomes in one go. A good analogy for this would be to consider putting shredded parts of many pages back together.

Sequence Alignment: A way of arranging the sequences of DNA to identify regions of similarity that may be a consequence of functional, structural or evolutionary relationships between the sequences.

Gene Expression: Process by which information from a gene is used in the synthesis of a functional gene product (e.g. proteins)

Base pair (bp): A unit consisting of 2 nucleotide bases bound to each other by hydrogen bonds. A base pair in DNA could either be Adenine-Thymine or Cytosine-Guanine.

2.2 DNA Methylation

As mentioned before, the data (and in an extension, the project) come from the **Epigenetics** field of study. Epigenetics [3] is the study of potentially heritable changes in gene expression without the involvement of the underlying DNA sequence, i.e. studying various biological mechanisms that influence which genes are switched on or off. Epigenetic changes could result in diseases like cancer, in which case various genes are switched away from the healthy state. In addition, these Epigenetic modifications can be influenced by external/environmental factors, including the lifestyle and disease state of each individual.

Nowadays, various epigenetic mechanisms are considered to initiate and sustain epigenetic change. One of which will be considered in this project and it is called **DNA methylation**. In particular, DNA Methylation [1, 3, 4] is a natural process by which methyl groups are added to the DNA molecule. In other words, DNA methylation refers to the addition of a methyl group to the DNA strand. More details on what a methyl group is and what its purpose is, is beyond the scope of this project.

This type of mechanism, occurs mostly on Cytosines (Cs). In this particular project, we will consider the conversion of Cs to 5-methylcytosines. Modifications such as these, usually occur on Cs which lie next to Guanines(Gs), called CpG methylation and the result is two methylated Cs lying diagonally to each other (on opposite strands). Cases of CHH, and CHG, where H can be represented by A, T or C, will also be considered throughout the project.

It is worth noting that DNA methylation has also been observed in Adenines(As) [1]. However, methylation

in As, has attracted less attention due to its less frequent observations.

2.3 Bisulfite Sequencing

A number of different approaches are used to detect DNA methylation in scientific research, many of those being introduced in the recent years. The most common and highly used method by scientists is called **Bisulfite Sequencing**[1]. It involves the use of bisulfite in DNA, as well as resequencing the DNA.

In particular, the usage of bisulfite converts the Cs in the DNA strand to Uracils(Us), but the 5-methylcytosines are left unaffected. Through the amplification of bisulfite-treated DNA, the 5-methylcytosines are amplified as Cs, and the Us as Ts. An approach (often called three-letter alignment) is used in order to re-align the DNA sequence with the induction of the methylation state of each Cytosine.

2.4 Summary

Now that some definitions in Biology have been introduced, let us recall what this project is about as well as include the proposed solution and novel approach to be used.

So far, scientists are considering the methylation percentage of a DNA strand, as well as the probability that a certain position in the DNA should be methylated or not [1, 2]. These percentages are helpful in order to determine various epigenetic changes or factors that could link the DNA methylation and human diseases(e.g. cancer, lupus etc).

However, patterns of the methylated states in various DNA fragments could potentially provide better and more accurate results. In fact, our interest lies with the identification of new links that are being overlooked through the methods being considered nowadays. In this project, we will investigate this possibility through an analysis of DNA methylation data. Obviously there is an immense amount of fragments to be analysed as well as various approaches to consider. Hence, due to the time limit of the project, we will limit the methods and areas(fragments) of DNA. This analysis though, will be the cornerstone of developing and implementing novel approaches to deal with DNA Methylation data.

3 Pre-processing

Before moving on to the statistical analysis and the various approaches considered in analysing the complexity of DNA Methylation data, let us discuss the software used to do so. In addition, it would be wise to briefly mention the data structure, as well as how the observations and variables will be used in the analysis.

3.1 Software

The analysis was performed mostly in R. However, due to the fact that we are considering "Big Data", we would have to reduce the size of the data before uploading it to the R workspace. To do so, High Performance Computing (HPC) was used.

In general, in clusters of HPC, it is wise to implement various "jobs" to be performed. These jobs are set in a queue of all the jobs entered by the users with access in the specific cluster. The job is then performed

and the results, output of it is given as specified. The structure of the job is similar to the following. Assuming the code we need to run is saved in a file called "bismark",

```
qsub  bismark
```

puts the job in the queue, while qstat shows the status of the job at that particular time.

An example of the code written in the jobs that have been implemented in this project can be found in the Appendix C.

In addition, in the HPC many pre-existing modules will be helpful in the overall analysis of the data. These modules would be used in the sequencing and alignment of the data, as well as in converting the data to a readable (.txt) format. First and foremost, **Bismark Bisulfite Mapper** [9] is the module responsible for making sense of the .fastq format, aligning the data based on the reference genome, extracting the methylation states and essentially anything related to bisulfite sequencing and aligning of the data.

Samtools is another module necessary in the analysis and in particular in reading the output of Bismark and converting it to a readable format. Bismark produces its results in a .bam file. This file could be read and uploaded to an R environment, but it would be easier to convert it to a .txt format and then reduce the size of the data through some additional approaches (e.g. Map Reduce) before loading the data in R. Samtools module is used as an extension of the Bismark module in order to obtain a sensible read out of the raw data given.

A Map Reduce approach has also been applied to a certain number of jobs in order to reduce the dimensionality, but also to reduce the number of observations with similarities (e.g. same segment position and methylation state). Having a counter variable, replacing a number of observations would reduce the size of the data-set and hence making it easier and simpler for us to upload the file to our local device for further analysis. The exploration and investigation of the data could also be performed in the HPC, but it would be more convenient to do so in the local device without the need of a connection to the cluster.

The scripts for a Map Reduce purpose were written in Python and can be found in the Appendix C.

The rest of the analysis, including exploration, further investigation and statistical performance was done in R. Different various statistical approaches and methods have been considered throughout this project in order to harness and analyse the complexity of the DNA methylation data.

The scripts used in R for all the statistical methods considered can be found in the Appendix C.

3.2 Data Structure

As previously mentioned, the data considered in this project are DNA methylation data from mice [2]. Mice are the most commonly used mammalian research model in genetics and epigenetics before moving on to other mammalian species (involving humans). Mice belong to the Euarchontoglires clade, in which humans belong to as well.

Due to the unique format of the DNA Methylation data and to the limited knowledge on sequencing (aligning) DNA, a lot of time was spent in obtaining a readable and efficient output of the data given. It is vital to understand what the data represent fully, prior to analysing them.

3.2.1 Raw Data-Sets

The data were provided by Dr Petra Hajkova from the Medical Research Council (MRC) and the London Institute of Medical Sciences (LMS). The data were split into 10 data-sets, each being of size around 10GB. The data-sets were split based on the Embryonic day, sex and type of the mouse. Let us call these data-sets, **Raw Data-sets** (RDS), which are:

(i) E10K, (ii) E10W, (iii) E12MK, (iv) E12FK, (v) E12MW, (vi) E12FW, (vii) E14MK, (viii) E14FK, (ix) E14MW, (x) E14FW,

where:

- E - Embryonic day, e.g. E10 meaning on the 10^{th} day
- M - Male
- F - Female
- K - KO type
- W - Wild type

where KO and Wild are simply two separate types of mice to be considered in the specific data.

Each of the above data-sets involved two separate files: one for the raw data, and one for the processed data. The former includes segments of the DNA specifying the existence of methylation in each position (more details about the raw data will follow), while the latter includes the methylation percentage at each position of the DNA, as well as the total number of observations with methylation occurring.

In this project we are analysing the complexity of the above data, and hence considering just the processed data would not be sufficient. The simple structure of the processed data is what scientists are using when looking at the DNA methylation data, while trying to come to some meaningful conclusions. Even though, the processed data will not be used in this analysis, they will be considered in obtaining some basic statistics as a first look at the whole DNA methylation data (See section 4.1).

On the other hand, the raw data which involve a more detailed information, will solely be used in the analysis of this project (by considering only the necessary variables). For each observation, these variables would vary from the chromosome it lies in, to the methylation state of each position it describes. To obtain these variables though, a certain analysis (sequencing and alignment), is required to be performed through the High Performance Computing (HPC), due to the large size of the data.

3.2.2 Bam Data-Sets

The raw data were in a .fastq format, from which we are unable to directly read (and understand). Hence, a conversion to a readable format (e.g .txt - text file) is required. To do so, a written program, called Bismark Bisulfite Mapper, which can be found in the module called bismark, will be used. This module already exists in the HPC, and all that is required from us is to simply load it and correctly input the necessary files. However, it should be specified that since the data are single-end, the corresponding reference genome was added in the process.

The output of the bismark [9] function gives out a new data-set, called **Bam Data-Set** with the following information per observation:

1. QNAME (Seq-ID)
2. FLAG (this flag tries to take the strand a bisulfite read originated from into account (this is different from ordinary DNA alignment flags!))
3. RNAME (Chromosome)
4. POS (Position)
5. MAPQ
6. CIGAR
7. RNEXT
8. PNEXT
9. TLEN
10. SEQ
11. QUAL (Phred33 scale)
12. NM-TAG (edit distance to the reference)
13. MD-tag (base-by-base mismatches to the reference)
14. XM-tag (methylation call string)
15. XR-tag (read conversion state for the alignment)
16. XG-tag (genome conversion state for the alignment)

Note that most of the above variables will not be considered any further, as it would be unnecessary for the purpose of this project. It is useful to extract at least the chromosome, the position (start and end) and the DNA sequence.

The methylation state is specified in the following way: There are 3 separate methylation contexts:

(i) CpG context, meaning a Cytosine followed by a Guanine

(ii) CHH context

(iii) CHG context

where H can be replaced by Adenine, Cytosine or Thymine. More specifically though, the methylation call for each context is specified as follows:

- z - C in CpG context - unmethylated

- Z - C in CpG context - methylated
- x - C in CHG context - unmethylated
- X - C in CHG context - methylated
- h - C in CHH context - unmethylated
- H - C in CHH context - methylated
- u - C in Unknown context (CN or CHN) - unmethylated
- U - C in Unknown context (CN or CHN) - methylated
- . - not a C or irrelevant position

An example of the above data outcome is the following:
HWI-D00467:104:C6V4YANXX:7:1101:2470:2189_2:N:0:CGATGT, 0, chr16, 94932665, 42, 100M, *, 0, 0, CGGAGTGA TTAGGTAGA TTAGGAGTGAA GTGTGTTT TGTGGAGGT TTTGTTT ATGATATGGG TGTTTTTGG AGTAGGGTT TGATGGGAG TAGATGAGGAA, «<BBBBB FFFFFFFF FFF FFFFFF FFFFFFFF FFFFFFFFFFFF FFFFFFFFF FFFFFFFF FFFFFFF FFFFFFF FFFFFF FFFF FFFFFFF, NM:i:16, MD:Z:8C0 C23C0C0C8 C4C0C5C7C0C1 C5C4C0C9C10, XM:Z:Z.....hx... ....hhx..... ...h....hh..... h.....hh.x... ..x....hx..... ...x....., XR:Z:CT, XG:Z:CT

Based on the above specifications, one could analyse the data as a whole by specifying Methylated (Upper case letters) or Unmethylated (Lower case letters), or more concentrated for each context separately. For simplicity, in this project we are taking the former case, for which we will replace the letters with a binary variable. The variable would take the value of 1, if methylated (Upper case), or 0, if umethylated (Lower case). Based on the results obtained through this approach, we would be able to investigate more closely on the context of the methylation call as well.

As mentioned before, since this project is of an exploitative nature in analysing the specific type of data, various approaches will be used. Some of these methods would require different measures and variables as other. Most of these measures would be calculated in R, either through HPC or through our local device. Examples of the calculations can be found in the Appendix C

3.2.3 Confined Data-Sets

After slicing the raw DNA methylation data, we end up with two main separate data-sets for each of the Raw data-sets. To avoid confusions, let us call these newly created data-sets as, **Confined Data-sets**. These two data-sets were created after compressing all observations that lie in the same segment position and having the same pattern, into one row with an additional counter variable as a column. In addition to this reduction, one data-set can be seen as a "summary" of the other. Even though, such an additional data-set might seem redundant, this simple structure will make the future main analysis much faster and simpler.

The first data-set, called **Binary Data-set**, consists of the following variables:

- Chromosome

- Start Position
- End Position
- Binary read
- Strength Measure
- Number of Observations

where Strength measure is computed as the Euclidean distance with the previous pattern. To avoid large unnecessary dissimilarities the observations have been sorted prior of the computation.

An example of the above data-set is given by:
chr1, 3014974, 3015073, 0++++++0+++++++++0++++++++++00+00++++++++++0++++0
00+++++0+00++++++++++0+0+++0+0++00++++++++00+++++, 0, 15
,where "0" represents Unmethylated Cytosines, "1" represents Methylated Cytosines and "+" represents Non-Cytosines.

In the above data-set, observations with the same Start/End positions but with different Binary Read can be found. Hence, the second data-set, called **Patterns Data-set**, comes into play. It consists of:

- Chromosome
- Start Position
- End Position
- Number of Patterns
- Number of Observations (Total)

where the specific binary reads are not necessary. Although, the number of patterns would prove to be helpful more than once. Note that in this data-set, there are no two separate observations with the same Start/End position.

An example of the above data-set is given by:
chr1, 3014974, 3015073, 1, 15

By having these data-sets to be used in R through our local devices, we are ready to start the exploration and investigation of the plethora of information found in the data.

3.2.4 Segment Data-Sets

For a big part of the analysis, we will only consider the Binary read of the segments as the only variable. For every observation, the segment consists of 100bp and hence every binary read also consists of 100 characters. Each of these 100 characters would become variables so, we can work with a new data-set made of all the observations as rows against 100 columns. We will call this data-set **100bp Segment Data-Sets**.

However, due to duplicated data points. it is necessary to reduce the base pairs per segment from 100bp to 50bp. That is done solely to obtain a larger number of observations. Even though, in the data a segment

might appear more than once, it is not always a "valid" observation. Due to amplification of the same original DNA molecule duplicate reads are expected to exist in the data. Hence, it is safe to assume that if a read (observation) has the same chromosome, same start/end position and same binary read as some other read, then it is a duplicate. Duplicate reads must be removed from the data, before moving on with the analysis, as it will produce misleading results.

By setting up the overlapping reads(See figure 4), one can obtain additional observations when reducing the base pairs. In this situation, reads with the same pattern are welcome. In fact, one can identify reads that occur often and reads that could be listed as outliers. We will call the newly sliced data-set **50bp Segment Data-Set**. This data-set was created after certain positions in the data (which could produce 50bp segments) were identified.

It is also worth mentioning the fact that even though we have 10 Raw data-sets, we will mostly work on six of them (Embryonic days 10 and 12). That is due to the time permit we have during the project. Not enough time was available to implement all of the following methods to the data-sets and compare the results between all possible combinations of these data-sets. In this report, results from the E10 data-sets will be mentioned, since similar conclusions were obtained from the other data-sets as well.

Lastly a mapping was performed on the data to separate the Methylated, Unmethylated and Non-Cytosine elements in the data. Suppose the matrix $X \in \mathbb{R}^{N\times 50}$ represents our data, where N is the number of observations. For each position $j = 1, \ldots, 50$ and observation $i = 1, \ldots N$ the following mapping was used:

$$X_{ij}= \begin{cases} 1, & \text{if Methylated Cytosine} \\ 0, & \text{if Non-Cytosine} \\ -1, & \text{if Unmethylated Cytosine} \end{cases}$$

Since categorical data are being considered, the above simple mapping will be used in the Analysis.

4 Analysis

In this section, we will discuss the potential statistical approaches considered and implemented in analysing the complexity of the DNA methylation data given as explained in the previous section.

For each method implemented in the analysis, a brief theoretical discussion will take place, as well as a more practical application tailored to the data-sets we have, will be explained. In addition, reasons for considering these specific approaches will be given. The conclusions reached after the implementation of these methods in the data-sets can be found in section 5.

4.1 Basic Statistics

It is reasonable to take a look at the basic statistics of the data, before moving on to more specific and deeper analysis. Some of the statistics about to be discussed were found automatically through the Bismark Bisulfite Mapper (See Appendix B), while others were computed directly from the processed data in HPC. We haven't mentioned the processed data much, since they will not be used in this project besides computing these starter statistics. Through the processed data though, we will be able to have a brief overview of the data.

The first step in every exploitative analysis should be to obtain a very well understanding of the purpose of such analysis. However, quantitatively, the most important step is having a deep knowledge of the data. That is the goal, of this section.

Basic statistics

Data-set	Number of Observations	Average number of patterns in each segment	Number of segments in 50bp
E10K	41958641	1.52	173
E10W	39167164	1.50	190
E12MK	40565773	1.86	173
E12MW	41344500	2.09	176
E12FK	40917239	1.79	138
E12FW	32373018	1.87	142
E14MK	33778380	1.88	156
E14MW	34657170	1.82	162
E14FK	34067204	1.78	134
E14FW	47437615	1.67	143

Table 1: Some statistics of the data to start us off.

The summary created from the Bismark Bisulfite Mapper is essentially what scientists are mostly considering when analysing DNA methylation data. Before implementing hypothesis tests, t-tests, Wilcoxon tests etc, one takes a close look at these statistics to obtain a better understanding of the specific data-set.

In Table 1, one can observe the number of observations (including duplicated reads) and the average number of patterns of each segment found in the data-sets considered throughout the project. In addition, the number of different segments found in the finalised 50bp Segment Data-set are also presented. That is the number of separate segments we are considering and in extension the number of tests performing for each data-set.

Based on a sample from the data and in particular the E10K data-set, consisting around 300000 different positions in the DNA, the following were observed and shown by the Figure 2: About 7% of the position have a higher than 50% methylation percentage, of which 2% have 100%. In addition, 82% have 0%. These values will help us in short-listing the fragments of the DNA with high methylation percentage, as well as in identifying potential patterns in a sequence.

4.2 Entropic measures

In this section we will discuss about a few measures which would be beneficial in identifying how similar the binary reads lying in the same segment are. As mentioned, different binary reads correspond to different patterns. On top of that, we want to see how similar the patterns are with each other.

Our goal is to analyse the data and find statistically significant results based on the segments as a whole and not just by looking at the methylation states of certain positions in the segment. This statement is equivalent in finding statistically significant results by looking at the patterns (different binary reads) directly, i.e. to specific combination of methylation states.

Shannon entropy, or often simply entropy, is widely referred to as the expected information gain. Only

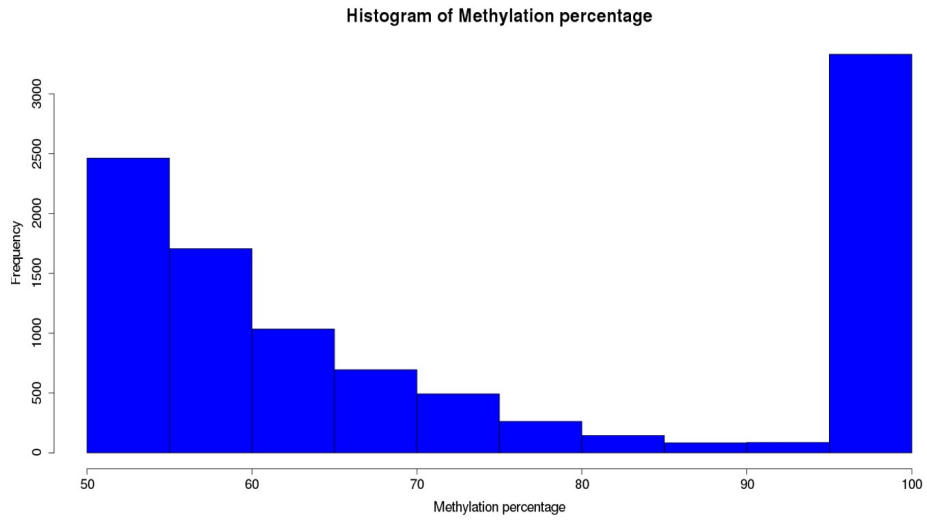


Figure 2: Histogram showing the Methylation percentage of above 50%.

the discrete case will be considered, since the events occurring reflect to the various possible patterns. Suppose M possibilities, with given probabilities of p_i, for $i = 1 \ldots, M$. The Shannon entropy is given by:

$$H = -\sum_{i=1}^{M} p_i \ln p_i$$

,where ln referes to as the natural logarithm. Information is usually measured in bits, and hence the logarithm of base 2 is often considered instead.

Entropic measures in DNA methylation data have previously been introduced, such as the Methylation Entropy by (Xie et al, 2011) [6], given by:

$$ME = \frac{e}{b} \sum(-\frac{n_i}{N} \log \frac{n_i}{N})$$

,where:

- e: Entropy for code bit
- b: Number of CpG sites
- n_i: Observed occurrence of methylation pattern i
- N: Total number of sequence reads generated

The above entropic measure has been implemented in C#, by introducing the DMEAS (DNA Methylation Entropy Analysis Software), a simple software for this exact purpose(He et al, 2013). [7] Code bit is referring to the segments in consideration.

As useful as the entropic measures has proven to be [6], the patterns in certain loci are not clearly considered. Instead of focusing on the entropic measures, through this project, we will try to explore other statistical methods that might prove to be significant by looking at the patterns(binary reads) and not just the percentage methylation state.

Throughout the analysis, a need of similarity measures will arise(See section 4.10). For these scenarios a good starting point would be using distance measures. In certain cases, it would be beneficial to look at the Entropic measures. However, due to the fact that a similarity of two observations is required, a distance measure seems more appropriate. Hence, Euclidean distance measure will be used as a similarity measure when needed.

4.3 Segment plots

A certain type of plots will be developed in this section. These plots are a good visualisation of the different patterns in each segment. In addition, through these plots, one can see the number of times each pattern is observed. Some patterns might be outliers, while some other might correspond to a big majority of the observations. In some cases, a couple of patterns seem to cover almost the whole set of observations for the specific segment. Examples of these cases along with their plots can be seen below.

Let us briefly mention the simple computation of such plots. As mentioned, all the segments in the data before slicing consist of 100bp. The plot will place one observation of the same segment on top of the other. Different colours on the figure will represent the different methylation state, with three possibilities:(i) Methylated Cytosine, (ii) Unmethylated Cytosine, or (iii) Non-Cytosine, i.e. Thymine, Guanine or Adenine. One can clearly see the different distinctions in patterns through the above very simple plot.

Based on the number of patterns and these plots, we will decide which segments to analyse and dive into first. Same plot has been developed for the 50bp Segment Data-set as well.

4.3.1 Combine segments

The observations consist of segments of 100bp each. In this subsection, we will try to see whether we can decrease the number of base pairs to consider with the goal to increase the number of observations. Through this plot, the 50bp Segment Data-set will be created.

Before implementing such an additional consideration in the data, it is better to take a look visually at the data and see whether that would help us or hurt us. It would hurt us, if the data can be found along a large range equivalently and combining them would be pointless, as it would produce data with less observations . It would help us, if there the observations lie in specific regions and hence combining them would be beneficial.

Implementing a simple function in R, considering the Start and End position of each observation, the results seen in figure 4 have been observed.

We can clearly see some regions in which we could indeed combine the reads for a smaller than 100bp range of read. However, for some regions the number of observations are significantly larger than other. Since the interesting regions for the combination are only a few and the range would not increase by much, we will stick with the structure of 50bp in each observation. It would keep things simple for now, but it would be interesting to investigate a bit deeper and more careful the regions with the potential of extending their base pair. It has not been investigated in this project, but suggestions on how to do so can be found in the section 6.

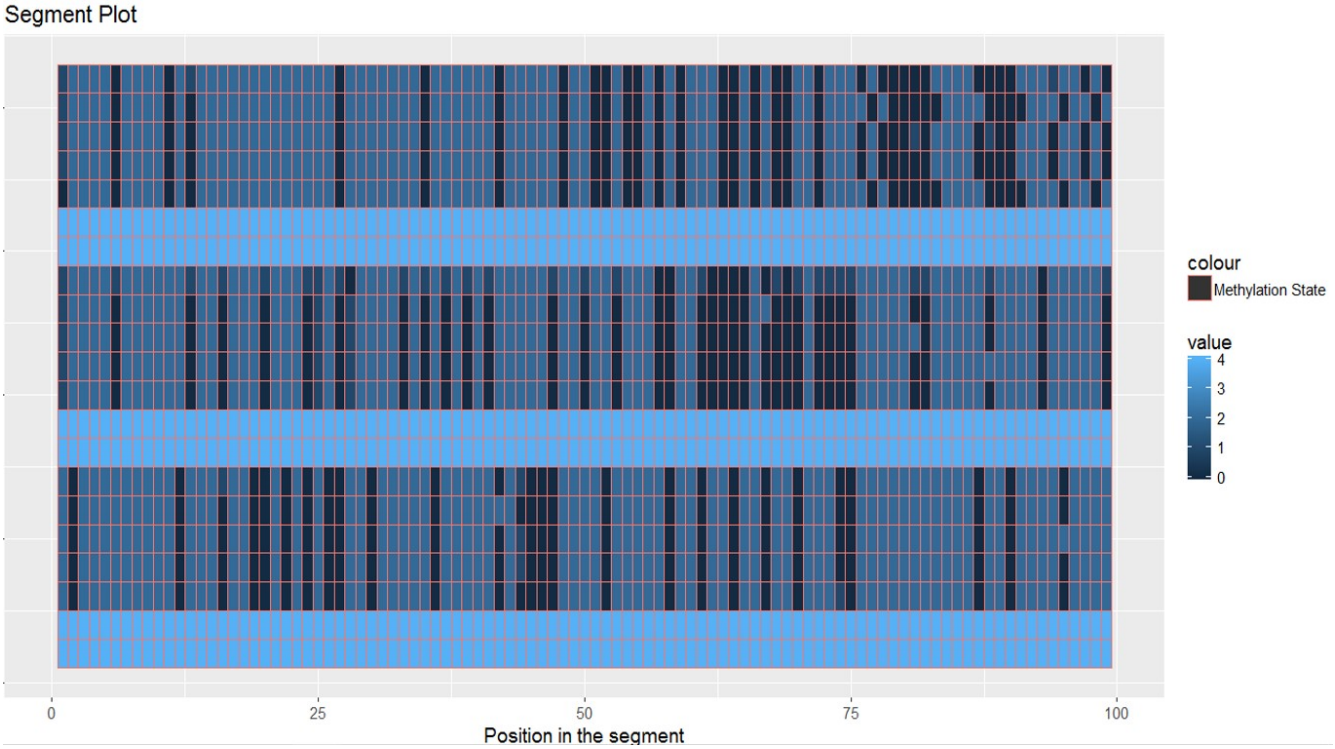


Figure 3: Three different segments represented by their respective (different) patterns. The x-axis represents the position in the segment, not in the DNA. Different colour represents different methylation state. Refer to the numerical values in the legend. 0 - Unmethylated, 1 - Methylated, 2 - Neither methylated nor unmethylated (non-Cytosine), 4 - Used to separate the segments. 3 has been neglected to make the separation clearer.

4.4 Wilcoxon Signed-rank test

Researchers are using hypothesis testing to look at the differences on methylation states in order to come to various conclusions [1, 2, 5, 14, 15, 16]. The usual case is that the distribution of the data is not specified, and hence a non-parametric hypothesis test method is implemented. The most common such test is the Wilcoxon Signed-rank test [10].

It is a non-parametric test used when comparing two related samples to assess whether their population mean ranks differ, i.e. it is a paired difference test. Wilcoxon Signed-rank test is usually considered as an alternative of the paired Student's t-test when the assumption of normally distributed data doesn't hold.

The following assumptions must hold before implementing the test [10]:

- Data comes from two matched, or dependent populations
- Non-parametric test, hence no requirement for special distribution of the dependent variable in analysis.

One can refer to Wilcoxon Signed-Rank test as a repeated measures test of dependency, i.e. a test in the difference of observations when the observations are matched. It pools all differences, ranks them and applies a negative sign to all the ranks where the difference between the two observations is negative. This is why it is called Signed Rank.

Notes:

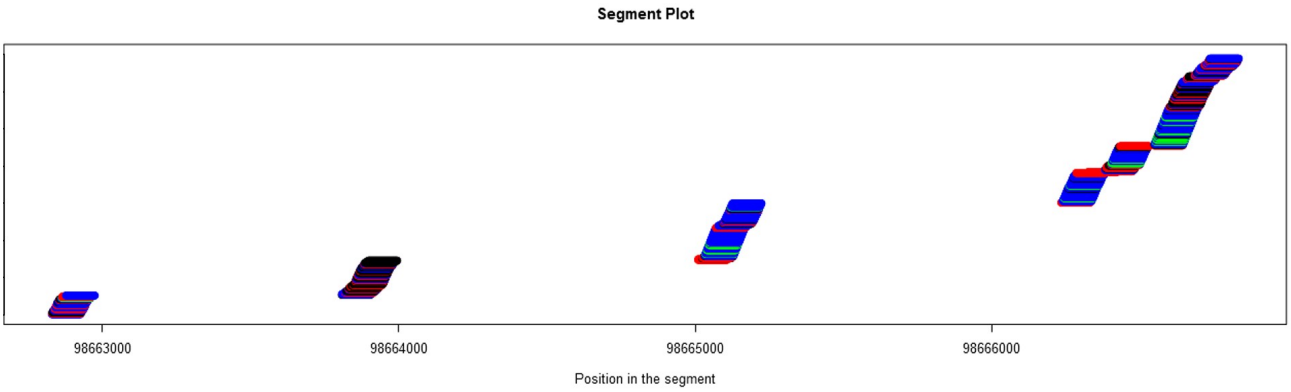


Figure 4: Segment plot starting from position 98662795 up to position 98666795. Colours on the plot refer to the number of observations were seen for each segment. Red if it was observed once, black for less than 10 observations, blue for less than 100 and green for greater than 100.

A. It tests whether the difference between two observations has a mean signed rank of 0

B. Thus it is much more robust against outliers and heavy tail distributions

Suppose $\mathbf{a}, \mathbf{b} \in \mathbb{R}^p$ are two observations. The observations could either be data points (vectors) or the means of a number of observations in two separate classes (clusters). A typical Wilcoxon Signed Rank test, would take as Null Hypothesis that the two observations are identical, and the Alternative Hypothesis being that they are not. In other words:

$$H_0 : \text{Observations } \mathbf{a} \text{ and } \mathbf{b} \text{ are identical}$$
$$H_1 : \text{Observations } \mathbf{a} \text{ and } \mathbf{b} \text{ are not identical}$$

The p-value of the test would determine whether or not there exist enough evidence to reject or not the Null hypothesis.

This test is the first step of the analysis of the DNA methylation data (that is of course after they have been sliced and written in the required format). This test will determine whether or not the data from one group (e.g. KO type) is identical with the other group (e.g. Wild type). The input (observations) would be the mean of the 50bp Segment Data-set

Assume that the significance level of the test is at 0.05. If the test shows that the two observations are not identical, i.e. rejecting the Null hypothesis by obtaining a p-value less than 0.05, then no steps further are really required. The percentage methylation state of the segments would provide enough evidence for any conclusion.

In other words, we are interested in the segments, for which the Wilcoxon Signed Rank test cannot classify the two observations, i.e. being unable to identify that the observations are different (or come from a different group). Cases which lead us in not rejecting the Null hypothesis (obtaining a p-value greater than 0.05), are the cases we are interested in and we will try to explore, follow, and investigate further.

4.5 K-Means

A big part of this project in various ways is classification. Even though, the complexity of the DNA methylation data lies much beyond simple classification, it is a good point to start the analysis and work on such data. A better understanding of the data-sets can be obtained by taking a closer look at the different classes that lie in the data. We can compare the data-sets regarding the type of the mice (KO or Wild type), the sex or even between the embryonic days 10, 12, 14. Classifying patterns could lead into interesting results, worth of a closer investigation.

K-means is a clustering algorithm [11]. In other words, an unsupervised classification method. By not considering any response variable, the algorithm tries to split the data into a specified k number of classes, called clusters.

In particular, consider N observations that lie in a p-dimensional space, i.e. data points $\mathbf{x}_i \in \mathbb{R}^p$ for $i = 1, \ldots, N$. It is necessary to initialise the cluster means, $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k \in \mathbb{R}^p$, where k is the number of clusters. The initialisation can be done either randomly or selectively given some prior knowledge of the data. The difference between the two initialisation approaches would be how fast the K-means algorithm would converge. Then we repeat the following steps, until convergence:

(i) $\forall i, c_i = \arg\min_j ||\mathbf{x}_i - \boldsymbol{\mu}_j||^2$

(ii) $\forall j, \boldsymbol{\mu}_j = \frac{1}{N_j} \sum_{\mathbf{x}_n \in \mathcal{C}_j} \mathbf{x}_n$

,where $\mathcal{C}_j$ represents the j^{th} cluster and N_j the number of data points that lie in it.

In other words, the algorithm is split into two parts: (i) Assign each observation to a cluster based on its Euclidean ($\boldsymbol{l_2}$) distance(smallest) with cluster's mean, and (ii) Re-calculate each cluster's new mean.

K-means will also be considered after the Principal Components Analysis [12] has been implemented (see section 5).

4.6 K-Medoids

K-medoids is another unsupervised clustering algorithm very similar to k-means. Besides computationally, the main difference between the two algorithms, is that the centers of the clusters are no longer the means of each cluster as in k-means, but they are data points such that they minimize the average dissimilarity (distance) to all the points that lie in the cluster. The distance measure usually considered in this algorithm is the sum of pairwise dissimilarities, i.e. the $\boldsymbol{l_1}$ norm.

There exist different variations of the algorithm with the most common realisation being the PAM (Partitioning Around Medoids) algorithm. Suppose we have N data points, $\mathbf{x}_i \in \mathbb{R}^p$ for $i = 1, \ldots, N$, and p being the dimensionality of the observations. The following steps take place in PAM:

- Select k of the N data points as an initialisation
- Assign each of the N data points to the closest medoid
- Repeat the following until optimality, for each medoid m and non-medoid o:
    1. Swap m and o

2. Re-calculate the sum of distances of points to their medoid, i.e. the cost
3. If the cost increased, then undo the swap

In our case, each variable (dimension) is a binary variable from the 50bp Segment Data-set, since it indicates whether or not a certain position in the segment is methylated or not. Hence, considering k-medoids instead of k-means might be a better approach, as the means of the cluster would no longer be binary. We are expecting very little difference between the results of the two algorithms, but k-medoids has the advantage of showing the pattern that lies in the center of each cluster.

4.7 Principal Component Analysis

A lot of the analysis performed in the data was based on the Principal Component Analysis (PCA). As a dimensionality reduction algorithm, PCA [8] could help us reduce the 50 (or 100 depending on the specific data-set considered) variables considering out of the binary reads in each observation.

PCA is the most common and one of the simplest dimensionality reduction algorithm which also lies in the unsupervised category. In short, it finds orthogonal projections of the data onto a lower dimension with the objectives to: (i) Minimise the average projection cost, i.e. the mean squared distance between data points and their projections, and to (ii) Maximise the variance of the projected data.

Suppose we have N p-dimensional observations (as input vectors), i.e. $\mathbf{x}_i \in \mathbb{R}^p$ for $i = 1, \ldots, N$. Our goal is to find a q-dimensional representation of the data, where $q \leq p$. In general, we want to optimise a the following q-dimensional representation:

$$\hat{\mathbf{x}}_i = \mathbf{m} + \sum_{j=1}^{q} a_{ij}\mathbf{u}_j,\ q \leq p$$

,where $\mathbf{m} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i$, is the sample mean. In fact, it is shown that the optimal zero-dimensional representation of the data is $\mathbf{m}$.

In other words the objective function is:

$$F_q(\mathbf{u}_1, \ldots, \mathbf{u}_q) = \sum_{i=1}^{N} ||(\mathbf{m} + \sum_{j=1}^{q} a_{ij}\mathbf{u}_j) - \mathbf{x}_i||^2$$

It has been proved [8] that $F_q(\mathbf{u}_1, \ldots, \mathbf{u}_q)$ is minimised when the vectors $\mathbf{u}_1, \ldots, \mathbf{u}_q$ are the q eigenvalues of S, the covariance matrix, taken in decreasing order of their eigenvalues. Note that the covariance matrix is given by: $S = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^T$.

The following assumptions need to be considered prior to implementing the Principal Component Analysis algorithm:

- Observations are real-valued
- No missing values
- No missing observations

Notes:

A. The corresponding eigenvectors are orthogonal and are the basis vectors for representing any input vector $\mathbf{x}$

B. Coefficients a_{ij} are different for each data point, as they depend on different $\mathbf{x}_i$

C. Centering the data ($\mathbf{m} = 0$) before implementing the algorithm helps in many ways. The q-dimensional representation would correspond to a rotation of the coordinate system to a new system defined by $\mathbf{u}_1, \ldots, \mathbf{u}_q$. The original data would be replaced by a new set of coordinates.

Practically, the following steps are taken into account when performing PCA:

(i) Collect data to a $N \times p$ matrix, $X = (\mathbf{x}_1, \ldots \mathbf{x}_N)^T$

(ii) Normalise the data, i.e. transform $\mathbf{x}_i$ such that they have zero mean

(iii) Find the p pairs of eigenvalues and eigenvectors of the covariance matrix S

(iv) Order the eigenvalues in a decreasing order, i.e. $\lambda_1 \geq \cdots \geq \lambda_p$

(v) Take the first q corresponding eigenvectors $\mathbf{u}_1, \ldots, \mathbf{u}_q$

(vi) Set $Y = XZ$, where:

- $Y = (\mathbf{y}_1, \ldots, \mathbf{y}_N)^T$ is the new $N \times q$ matrix containing the reconstruction, i.e. the "new" data
- $Z = (\mathbf{z}_1, \ldots, \mathbf{z}_N)$ is the $p \times q$ matrix created by placing the q eigenvectors alongside one another (as columns)

It is also worth noting that usually the first few Principal Components (PCs) provide enough variability and a good reconstruction of the data.

In this project, it could make sense to consider a dimensionality reduction as reducing the dimensionality of a 50-dimensional vectors would be extremely beneficial. Even though the data we have are categorical, i.e. discrete (and in particular binary), considering PCA as a dimensionality reduction algorithm can still be useful.

4.8 Hierarchical Clustering

Hierarchical clustering (HC) is another unsupervised learning approach for clustering but different than k-means or k-medoids.Unlike those two other algorithms, one does not need to clarify the number of clusters (what k was representing) when using the HC algorithm. However, just like k-means and k-medoids, HC requires the specification of a distance measure.

Consider the set-up: At the bottom of the structure lie all of the observations. Each observation is its own cluster, while at the top, all observations belong to one big cluster. Based on this set-up, two basic paradigms of HC arise:

1. Agglomerative (Bottom-Up): Start at the bottom. At each level merge a selected pair of clusters into a single cluster. The two groups merged into one has the smallest intergroup dissimilarity

2. Divisive (Top-Down): Start at the top. At each level split one of the existing clusters into two new clusters. The two new groups have the largest between-group dissimilarity

Dissimilarity can be measured by various distance measures, such as (i) Euclidean - $\sqrt{\sum_i (a_i - b_i)^2}$, (ii) Manhattan - $\sqrt{\sum_i |a_i - b_i|}$, (iii) Mahalanobis - $\sqrt{(\mathbf{a}-\mathbf{b})^T S^{-1}(\mathbf{a}-\mathbf{b})}$, where $\mathbf{a}$ and $\mathbf{b}$ are vectors and S the covariance matrix.

A visualisation of Hierarchical Clustering can be achieved by a dendrogram, a tree-diagram that is highly interpretable (See figure 8b in section 5). The visualisation component of HC plays a big role in how attractive this algorithm is in clustering, as well as when trying to confirm various results.

In our case, one can observe whether patterns seem to belong in the same clusters as they should, depending on their group, i.e. type(KO, Wild), sex(M or F), Embryonic day and combination of those. The way the clusters are split (or merged) would also be beneficial to observe, as it could potentially lead into interesting results, pointing to a further, deeper investigation for certain loci.

4.9 Regression

As a final part of the project we will consider a regression approach which would produce a statistical model on our data. Unlike the clustering approaches we have previously discussed, regression methods require a response variable. In this project, a binary response variable will be considered which would indicate whether or not the input is a control or a case (depending on the groups, control could be Wild type and case KO type. Similarly, it could also be Male or Female respectively, as well as more possible responses).

Since a binary response variable is involved, it is common to consider Logistic Regression. It is a Generalised Linear Regression with a binary response. Logistic regression makes the following major assumptions:

- The outcome must be discrete taking one of two possible values
- No outliers in the data
- No high inter-correlations among the predictors (multicollinearity)
- Control takes value 0 and case takes the value 1. If the likelihood of the input being a control is > 0.5, then it is assumed to be a control, otherwise it is assumed to be a case.

Due to some of the above assumptions (particularly multicollinearity)[24], implementing logistic regression is not the best possible regression approach to consider. Hence, Shrinkage regression models have been considered instead.

Multicollinearity refers to the existence of (near-) linear relationship among the independent variables. LASSO, as well as Ridge regression (and in fact Elastic-Net) can deal well with data in which multicollinearity might exist among the predictors [24]. Since multicollinearity could create inaccurate estimates of the regression coefficients, shrinkage methods will be implemented instead of the logistic regression.

After comparing the three Shrinkage regression models [20], we have concluded to focus on LASSO regression [18], and we will briefly discuss this approach. The main difference between LASSO and Ridge

Accuracy measure on Shrinkage regrssion on E10 data-set

End Position of Segment	LASSO	Ridge	Elastic-Net ($\alpha = 0.05$)
5860612	0.639	0.639	0.650
146261282	0.766	0.756	0.766
39845187	0.634	0.634	0.645
39847676	0.769	0.785	0.769
39848575	0.782	0.782	0.782

Table 2: Comparison between three Shrinkage regression models on some specific segments from the Embryonic day 10 data-set on 50bp.

is penalisation component [20]. LASSO uses an $\boldsymbol{l_1}$ norm, which refers to absolute penalisation, instead of the $\boldsymbol{l_2}$ norm which is used by Ridge regression. Elastic-net is thought of as a combination of these two regression models. Due to the usage of the $\boldsymbol{l_1}$ penalisation, LASSO has the advantage of shrinking some regression coefficients exactly to zero.

By comparing the three Shrinkage regression models,we have observed very similar (almost identical) results. The results were based on some performance measures which will be discussed further in Section 5. In the table the Accuracy (or Prediction rate) values of some examples can be seen. Since LASSO has the advantage mentioned above, we will focus on just LASSO regression throughout our analysis.

4.9.1 LASSO Regression

[18]Consider the response vector $\mathbf{y} \in \mathbb{R}^N$ and $N \times p$ input matrix $X \in \mathbb{R}^{N \times p}$. As in the usual regression set-up, we assume either that the observations are independent, or that the $y_i s$ are conditionally independent given the $x_{ij} s$. We also assume that the x_{ij} are standardized so that $\sum_i \frac{x_{ij}}{N} = 0$ and $\sum_i \frac{x_{ij}^2}{N} = 1$. The LASSO estimate $\hat{\boldsymbol{\beta}}_{LASSO}$ (which involves the coefficients) is defined as:

$$\hat{\boldsymbol{\beta}}_{LASSO} = \arg\min_{\boldsymbol{\beta}} ||\mathbf{y} - X\boldsymbol{\beta}||^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

, where λ is the tuning parameter. Note that as λ increases, more coefficients are set to zero.

There is no standard choice for λ, and hence cross-validation will be considered in order to determine the optimal value of the tuning parameter.

Some of the properties [18] of the LASSO method include the following:

- Prior using the method, we standardize the input variables by centring its mean at 0 and variance at 1. This is done so every variable would contribute equally to the analysis.
- The greater the tuning parameter is, the greater the amount of shrinkage.
- If $\lambda \longrightarrow 0$, then no penalty is applied.

4.10 Networks

In this section, we will shortly discuss our aim in improving the visualisation of the data beyond the plots discussed in the section 4.3. Having a good visualisation of the data comes a long way and helps in many ways. In addition, these networks could also be used in Correlation clustering, a different clustering method that has not been implemented in this project, but could show promising results (See Section 6).

It involves two separate simple networks that when looked closely could be beneficial.

Even though the restricted time did not permit us in grasping in full the possible potential of these networks, a short discussion will made on how and why these would be created.

4.10.1 Local graph

The "local" network is a graph unique for each locus, i.e. for each segment of the DNA. To create a network, we need a set of nodes and edges. Let us define the $G_{local} = (V_l, E_l)$, where the set V_l corresponds to the vertices and E_l to the edges. In this local network, the different patterns would be represented as nodes. This graph would be a path graph, i.e. a node will be connected with two other nodes, its predecessor and its successor. Hence, the order of the vertices would play a big role in this graph.

Starting from a random pattern in the specified segment, we will order the rest of the patterns in a decreasing order of their similarity (or entropic) measure with the previous pattern. In other words, the closest two nodes are the most similar ones. In addition, the size (thickness) of the edge would depend on the value of the similarity measure. Hence, by glancing in the network, one can see which two nodes have the biggest dissimilarity (by looking at the size of the edge).

The thicker the edge, the larger the dissimilarity. If none of the edges seem large enough, then it would be safe to assume that all patterns are similar to each other (or at least to two of the other patterns).

However, if some edges seem larger than others, then it would be possible that the data in the specified segment could be classified, e.g. if 2 edges seem thicker than the rest, then we could conclude that we could split the data into 3 main classes, and investigate those directly.

The similarity measure is tricky and further progress should take place in its regards. More details in the section 6.

4.10.2 Global graph

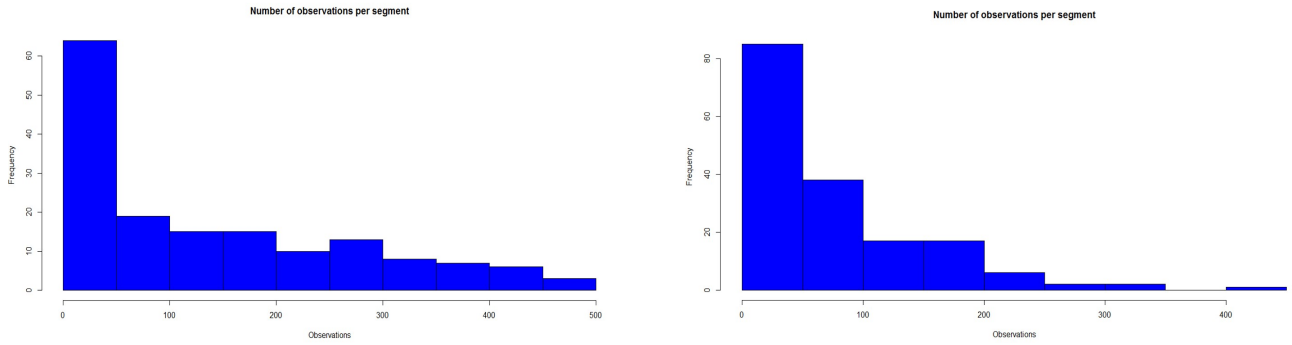
Let us define the graph, $G_{global} = (V_g, E_g)$. The vertices would correspond to the different segments and the edges would correspond to the similarity between the segments. In a way, each local network discussed in the previous subsection, would represent one node in the global network.

The main question in this approach is how we would measure the similarity of the segments. It could vary from the number of different patterns, to a distance measure between the closest (or furthest) patterns in the two nodes in question. This approach has not been implemented nor tested and hence some ideas will be discussed in the section 6.

	True	
	Class A	Class B
Estimated Class A	Tp	Fp
Estimated Class B	Fn	Tn
	P=(Tp+Fn)	N=(Fp+Tn)

Precision rate	Prec=*Tp*/(*Tp*+*Fp*).
True Positive rate (recall)	Tpr=*Tp*/*P*.
Accuracy	Acc=(*Tp*+*Tn*)/(*P*+*N*).
Error rate	Err=(*Fp*+*Fn*)/(*P*+*N*).
False Positive rate	Fpr=*Fp*/*N*.
False Negative rate	Fnr=*Fn*/*P*.
True Negative rate	Tnr=*Tn*/*N*.

Figure 5: Confusion Matrix and measures[25]



(a) Wild Type (b) KO Type

Figure 6: Histograms showing the number of observations per segment found in both KO and Wild type of the 50bp Segment Data-set.

5 Results and Conclusions

In this section, we will discuss the results and outcomes the analysis has produced. The results presented are based on the $E10$ data-sets and in particular the 50bp Segment data-set. However, same methods have been implemented in the other data-sets as well, for which a brief discussion will take place.

In Figure 6, the histogram of the number of observations per segment of both data-sets can be seen. For the development of the histograms, the 50 bp Segment Data-sets were used, i.e. without any duplicated reads. Even though most segments have less than 100 observations, some have as much as 500. It is worth noting that more observations does not necessarily mean more patterns or the other way around.

Let us focus on the 50bp Segment data-set. We start off by executing a Wilcoxon Signed-Rank test on the averaged values of the two types, i.e. we end up comparing two 50-dimensional vectors on whether they are identical or not. In the E10 data-set there are 165 separate segments to compare. For each of these segments a test will be performed independently.

Wilcoxon Signed Rank test results on E10 and E12 data-set

E10	E12	
165	171	Total number of tests
58	121	Tests with p-value $>$ 0.05
107	50	Tests with p-value $<$ 0.05

Table 3: The results from the Wilcoxon Signed Rank test on Embryonic day 10 and 12 50bp Segment Data-sets.

By taking a look at the results of the tests (Table 3), one can observe that the number of segments to consider in the E10 data-set have been reduced drastically, around one third remaining. The 58 remaining segments have shown to be identical between the two types and hence our interest lies with these observations.

The next step of the analysis would be to implement a LASSO regression on the remaining data. It is worth noting that LASSO regression should produce better results in the segments left behind. A confirmation of this will also take place. Different measures will be used in assessing the performance of the regression. Considering only the prediction rate (Accuracy) can lead to misleading results. Hence, a confusion matrix will be created (See Figure 5).

The measures that will be considered are the following:

- Precision: $P = \frac{Tp}{(Tp+Fp)}$
- Recall: $R = \frac{Tp}{(Tp+Fn)}$
- Accuracy: $A = \frac{(Tp+Tn)}{(Tp+Tn+Fp+Fn)}$
- F-measure: $F = 2\frac{P \cdot R}{(P+R)}$

In other words, F-measure is the harmonic mean of the Precision and Recall.

Before implementing the LASSO regression, care must be taken in regards of the sample sizes. It has been observed that in all segments, there is an imbalance between the two sample sizes, with Wild type having more observations than KO type. Incorrect and misleading results can occur when working with imbalanced data.

As a solution we have considered over-sampling the KO type data, i.e. some of the observations found in the KO type data-set will be repeated until the sample size is close to the Wild type's sample. After doing so, we can move on with the implementation of LASSO regression.

Performance measures on LASSO regression on E10 data-set

End Position of the segment	Accuracy	Precision	Recall	F-measure
5860612	0.639	0.800	0.444	0.571
146261282	0.766	0.652	0.500	0.566
39845187	0.634	0.583	0.824	0.683
39847676	0.769	0.793	0.719	0.754
39848575	0.782	0.831	0.711	0.766

Table 4: The results from the LASSO regression on Embryonic day 10.

Most of the segments had poor outcomes based on the above performance measures. However, 5 segments have shown promising results. These are the ones with End position: (i) 5860612, (ii) 146261282, (iii) 39845187, (iv) 39847676 and (v) 39848575, with their corresponding measures found in the Table 4. In addition, the confusion matrices can be seen in the Figure 7.

Based on the confusion matrices and the performance measures, the segments ending in 39847676 and 39848575 seem to be the most promising ones. However, visualisation and clustering through PCA, K-

Confusion Matrices

	True class A	True class B
Estimated class A	33	5
Estimated Class B	12	56

5860612

	True class A	True class B
Estimated class A	34	40
Estimated Class B	12	56

39845187

	True class A	True class B
Estimated class A	121	16
Estimated Class B	30	30

146261282

	True class A	True class B
Estimated class A	27	6
Estimated Class B	9	23

39847676

	True class A	True class B
Estimated class A	70	12
Estimated Class B	24	59

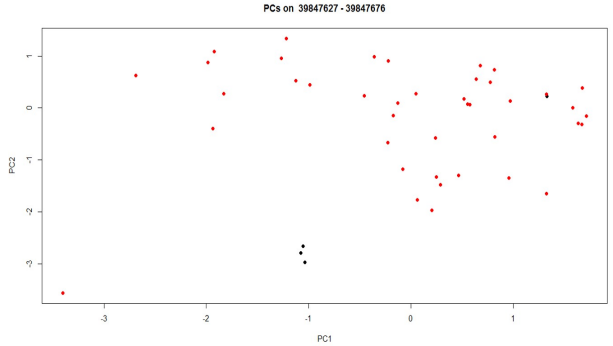
39848575

Figure 7: Confusion Matrices and measures for the 5 remaining segments.

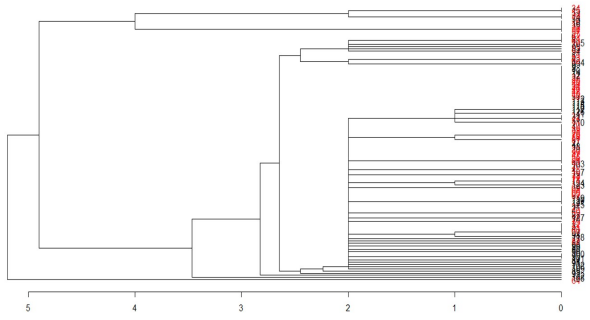
Medoids and Hierarchical clustering will be performed on all of the 5 above examples. The corresponding figures can be found in the Appendix A.

Starting from 165 segments to analyse, we are left with 5 promising segments remaining which we investigate further starting off with PCA. In this section figures of the segment ending in 39847676 will be presented.

The first two principal components cover around 55% of the total variance of the data, while the first three cover as much as 65%. PCA visualisation can be seen in Figure 8a for the first two principal components, while Figure 9 shows a 3D representation of the first three PCs. Black dots represent the KO type segments, while the red dots represent the Wild type. It can be easily determined that the KO type segments are represented by a lesser number of separate observations, but even so they are concentrated in certain coordinates. This behaviour is observed for the rest of the segments as well, but not so distinctly.



(a) Visualisation of the two first principal components, covering the 55% of the variance.



(b) Visualisation of the Hierarchical clustering

Figure 8: Two visualisation plots for clustering, based on PCA and Hierarchical clustering, representing the segment ending in 39847676.

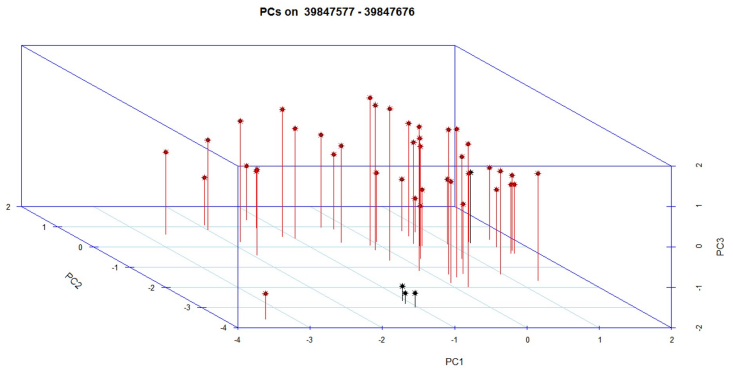


Figure 9: Confusion Matrix and measures[25]

In figure 8b, one can observe the resulting dendrogram of Hierarchical clustering on the same segment. At the bottom of the plot, a collection of some "black dots" or observations of KO type can be seen which could reflect on some particular difference in patterns. In the other part of the dendrogram observations of KO type and Wild type seem to be mixed up, suggesting small dissimilarity.

Furthermore, the implementation of K-Means and K-Medoids in order to cluster the data in two groups and potentially distinguish the two types was not as fruitful as we were hoping. In other words, both algorithms were applied with $k = 2$ and split the data into two clusters. Even though, in one of the clusters, there was a higher probability of having Wild type observations, the testing results of these approaches have not been significant enough. Same results have been observed when the algorithms were applied for the other segments as well.

Unlike the segment ending in 39847676, it was observed that the other segments' observations of KO type blend in with the Wild type's. Hence, a clear deduction cannot be conducted for the time being. The inability to obtain any evidence of clustering could suggest that the performance measure of LASSO are not very meaningful. Further analysis could be applied in order to make a determination.

Same analysis has been performed on the data-sets of $E12M$ and $E12F$, as well as combining the two sex together. None of the segments have shown any promising leads or significant results.

As mentioned, LASSO regression has also been applied in cases for which the Wilcoxon Signed-Rank test has shown that the data-sets are non-identical. Simply as a confirmation and as expected the results of the Shrinkage regression approach were in most cases better than the rest. As a result the averaged Accuracy measure was 0.688 with only two segments having a below 0.5 value.

Finally, a representation of the local network mentioned in the previous section 4.10 is shown in Figure 10. Each path graph represents a certain segment. The nodes within the path graph represent the different patterns found in the data within the same segment. The size (thickness) of each edge is valued by the similarity measure between the corresponding patterns represented by the two nodes in question (the edge connects).

In the figure there can be seen three separate examples. In the nodes 1-6, all the patterns seem to have a low dissimilarity measure, since the size of the edges in between is the same. However, in nodes 28-37, two edges seem to be thicker, one of which lies in the middle of the path graph. The middle thicker edge could reflect in having some difference between those patterns. In this path graph, we can expect two main different patterns. Lastly, segments with only one patterns (e.g. Nodes 7, 27, 45 etc.) are represented

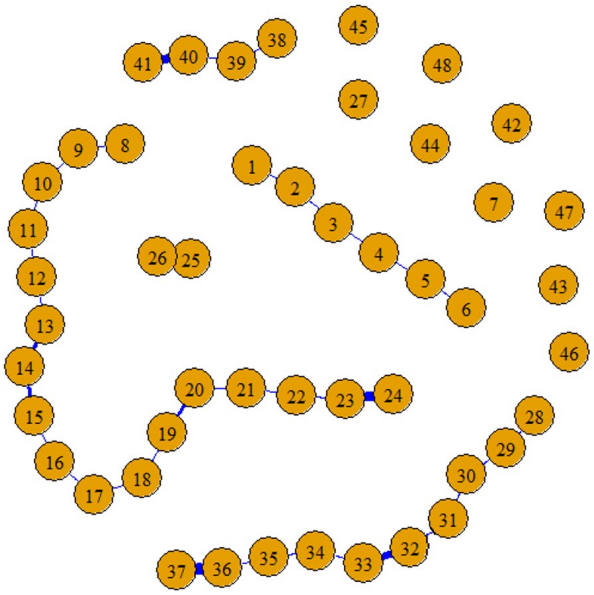


Figure 10: Confusion Matrix and measures[25]

simply as a node (path graph with a single node).

6 Further Work

This project has laid a good foundation for further research with potentially important results by harnessing and analysing DNA methylation data. Plenty of further work can be done, some of which will be mentioned briefly in this section.

First and foremost, due to the lack of experience in working with HPC and analysing big data in general, the implementation of the statistical methods was computationally very expensive. Hence, a good first step would be to make the analysis faster, by implementing a better programming (analysis) code and by adding parallel programming to it.

Furthermore, a lot of time was spent in understanding the data, slicing them up and transforming them to the format necessary for the analysis to be performed. That resulted in having a restricted time-frame available to implement all of the above statistical methods to all possible combinations of the data-sets. A close look and investigation on some of the data-sets was not taken throughout this project. By exploring these data-sets deeper, we could obtain some interesting new results. Such additional analyses could result in beneficial outcomes.

This was an exploitative project, looking at various statistical approaches which could potentially reveal some interesting conclusions about the data we have. Some methods have been implemented, but these are just a few of many more algorithms and statistical methods that could be used. For example, due to the methylation differences in certain positions, one could argue that spatial and temporal autocorrelation could provide a good statistical approach.

In consideration with spatial autocorrelation, one can also consider the more specific approach of Local

Indicators of Spatial Autocorrelation (LISA)[23]. On top of spatial autocorrelation methods, Correlation Clustering [21, 22] could also prove to be beneficial.

In particular, Correlation clustering [26] can be used on the local networks introduced in section 4.10, since this method's intuition is finding a clustering that maximises the number of "similar" edges within (or between) clusters. This method is used on similarity graphs with the purpose to group together similar items (nodes) and distinct dissimilar ones. Hence, it could be helpful in clustering based on similarity measures of the patterns.

Overall, only a hand-full of segments have shown interesting and promising results. However, Neural Networks and Deep Learning could carefully look deeper in the data with a great potential of identifying new significant results. Deep learning methodologies are improving constantly and have been successfully used in harnessing Big Data. Combining neural networks with the statistical approaches considered already in this project could provide more intuitive conclusions regarding the analysis of DNA methylation data.

In addition, Bayesian statistical methods have not been discussed or considered at all. Bayesian data analysis requires some prior knowledge in addition to the likelihood of the data. An analysis of the data in Embryonic day 10, could provide some prior knowledge in analysing the data in Embryonic day 12 and 14. Empirical Bayes methods would be a potentially interesting approach. A lot of care must be taken though, since we are dealing with big data and that could lead to certain difficulties.

A brief discussion took place in section 4.10 regarding a visualisation of the data. Improving the work already conducted on networks would be very beneficial in having a nice visualisation of the data. By looking at the data in that way, one can understand the data easier and be able to analyse the data with a higher efficiency.

It has been mentioned that throughout the process we have assumed Cytosines in the segments to be either Methylated or Unmethylated (as we should). However, we have not took into consideration the different contexts of methylation, i.e. whether it is a case of CpG, CHH, or CHG. Removing this assumption and taking each methylation context separately could lead to new interesting results. However, one can start analysing the loci based on the conclusions we reached through this project.

In conclusion, through this project, some valid results were obtained. Even though, further work needs to be done for more concrete conclusions, a promising and good starting point has been accomplished.

A Figures

In this appendix, the figures produced on the 5 promising segments found as mentioned in section 5 are represented. A PCA visualisation, a barplot showing the averaged difference in Methylation for each position in the segment, a clustering based on K-medoids algorithm with $k = 2$ and a dendrogram will be shown for each segment.

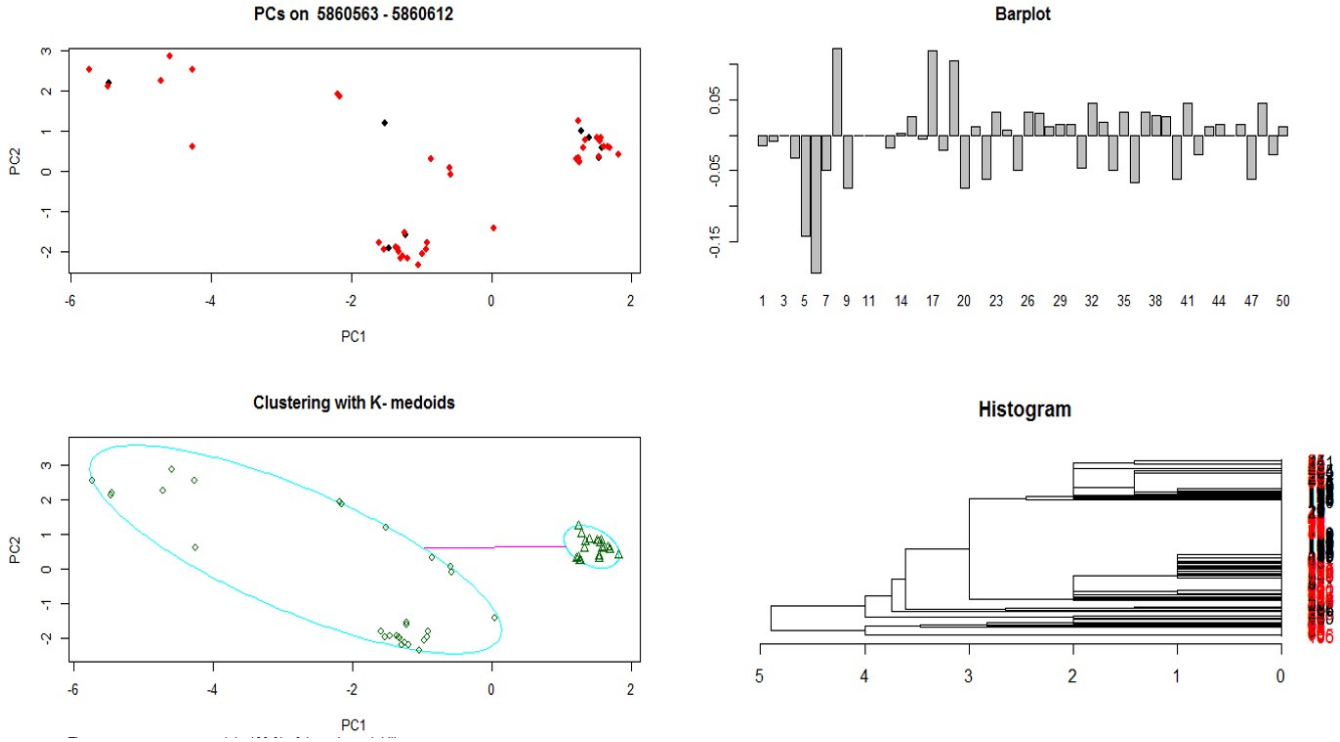


Figure 11: PCA, Barplot, K-medoids and Hierarchical Clustering plots of the segment ending in 5860612

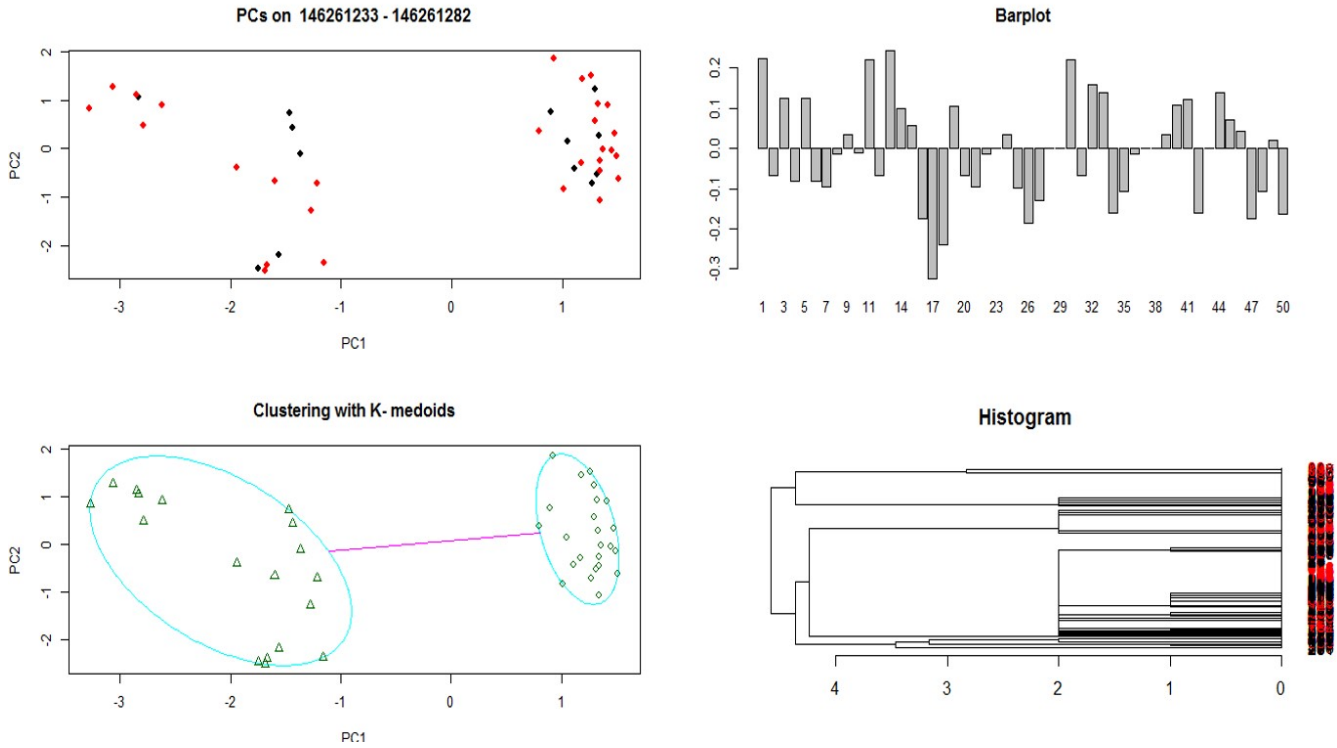


Figure 12: PCA, Barplot, K-medoids and Hierarchical Clustering plots of the segment ending in 146261282

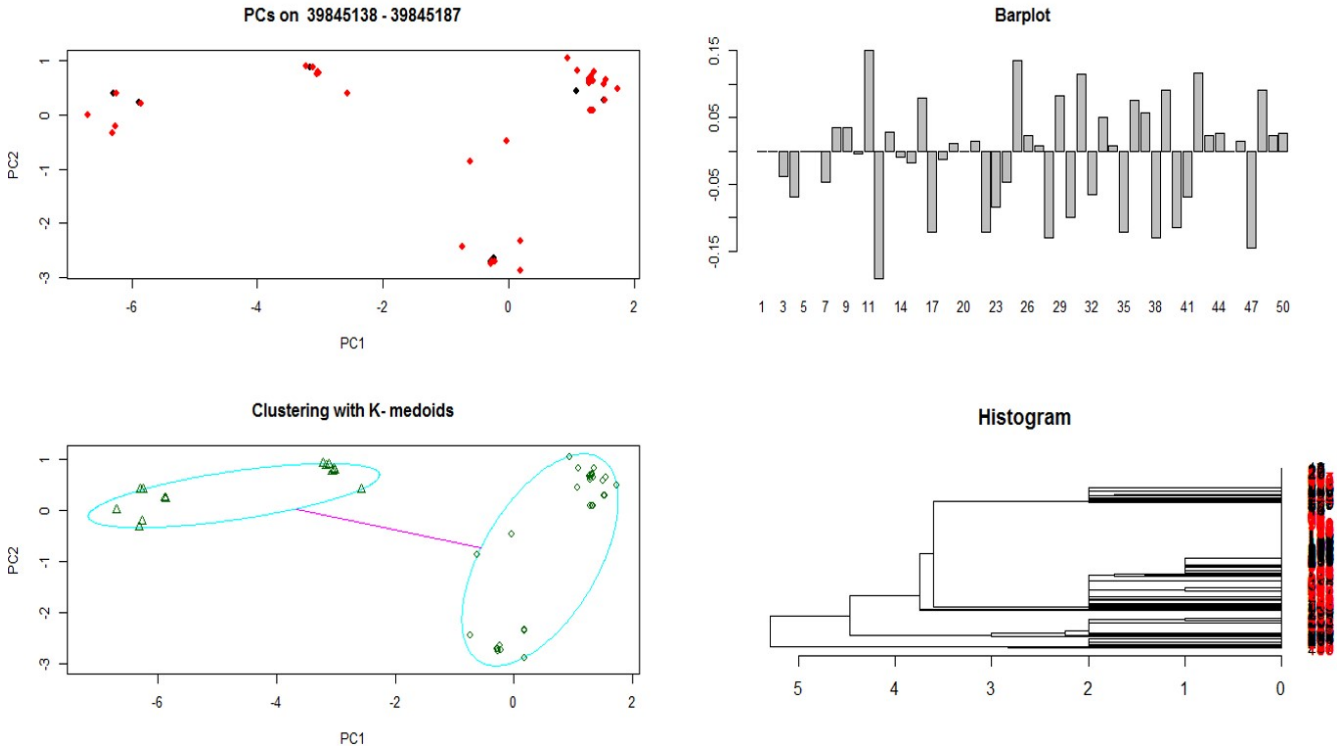


Figure 13: PCA, Barplot, K-medoids and Hierarchical Clustering plots of the segment ending in 39845187

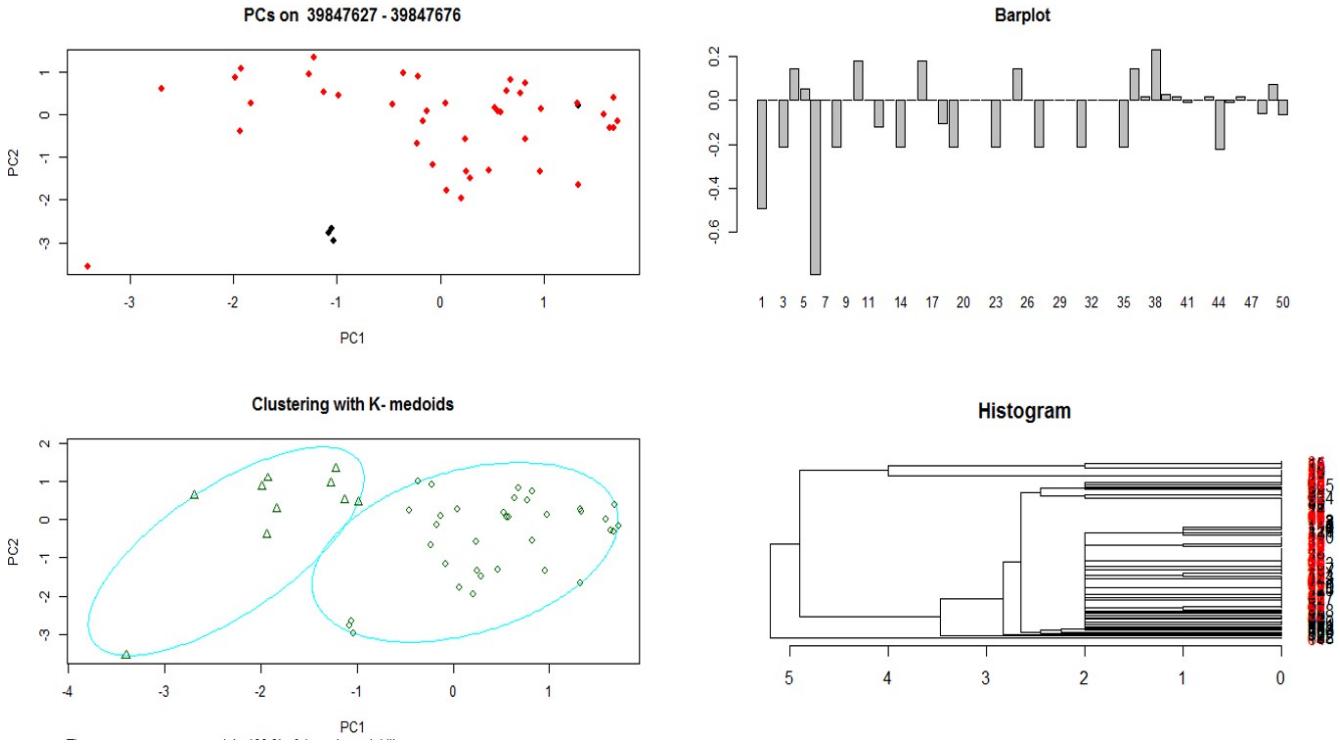


Figure 14: PCA, Barplot, K-medoids and Hierarchical Clustering plots of the segment ending in 39847676

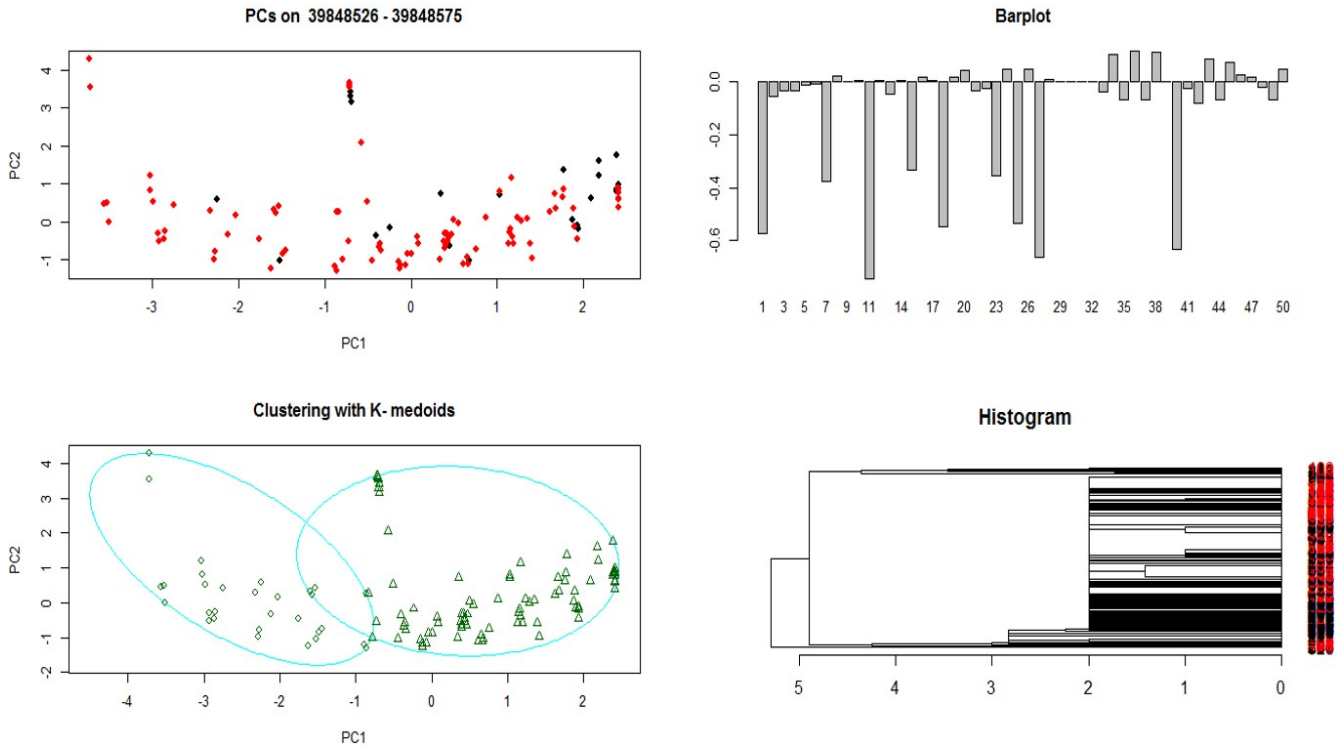


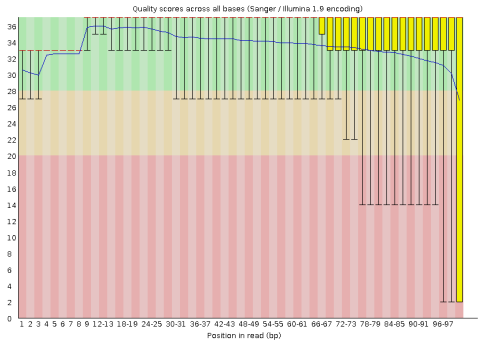
Figure 15: PCA, Barplot, K-medoids and Hierarchical Clustering plots of the segment ending in 39848575

B Basic Statistics Report

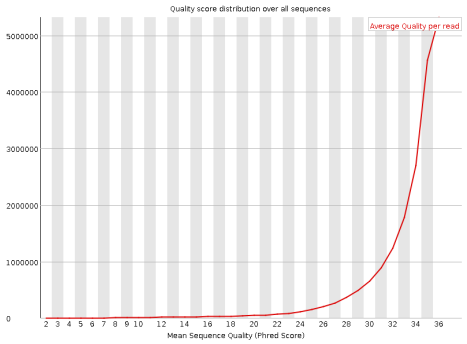
In this Appendix, an automatic report produced by Bismark Bisulfite Mapper will be shown. The Basic Statistics report is prepared right after running the *bismark* function of the module. It varies with information regarding the quality of the data and the methylation percentage, but it also gives a quick and short overview (summary) of the data. Some of the plots are shown below.

Measure	Value
Filename	RRBS_E10K1.fastq.gz
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	19413835
Sequences flagged as poor quality	0
Sequence length	100
%GC	28

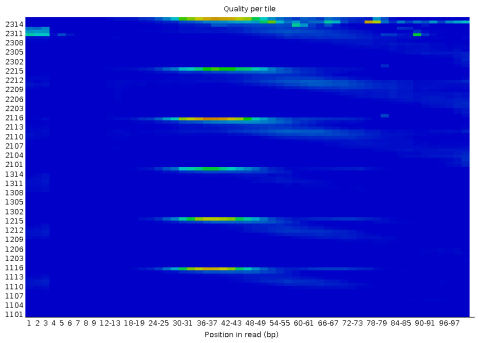
Figure 16: First look of the data. A useful and important overview.



(a) Per Base Quality.



(b) Per Sequence Quality



(c) Per Tile Quality

Figure 17: Quality of the data according to Bismark analysis.

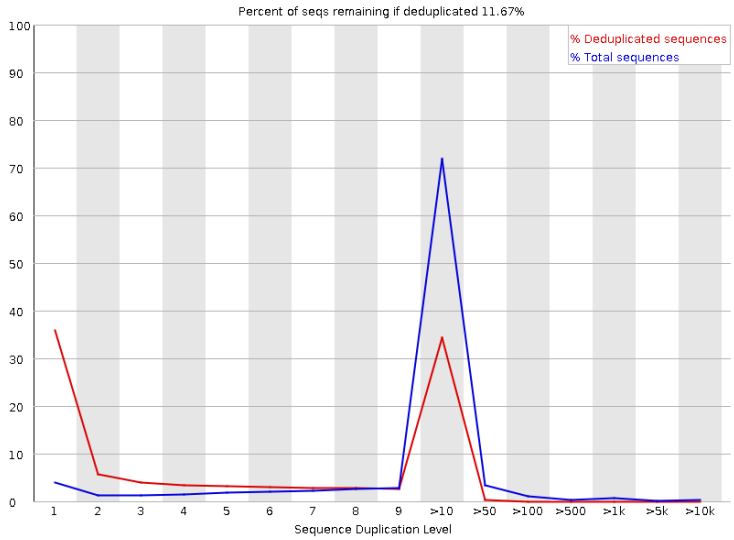
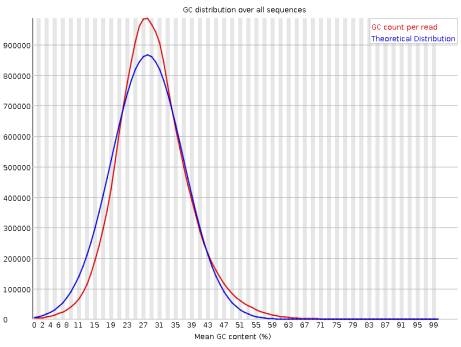
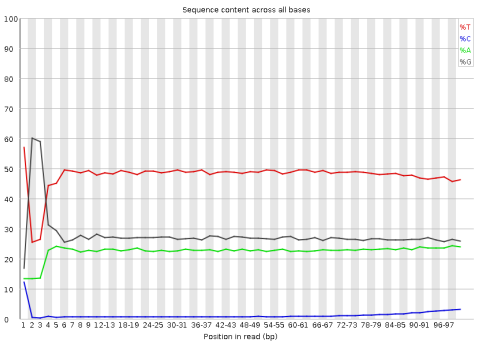


Figure 18: As mentioned in the report, most of the observations had duplicated reads. In this figure we can see the number of duplications.



(a) GC Content per Sequence.



(b) Content per Base Sequence

Figure 19: Content of CpG of the data according to Bismark analysis.

C Coding

In this Appendix, part of the coding used in the project will be presented. The bash commands and jobs in the HPC will be described first, with Python and R code to follow.

```
#!/bin/bash
#PBS -N Bis
#PBS -l walltime=72:00:00
#PBS -l select=1:ncpus=10:mem=64gb
#PBS -M theodoulos.rodosthenous16@imperial.ac.uk
#PBS -m ea

module load bowtie/2.1.0
module load bowtie2/2.2.9
module load samtools/1.3.1
module load bismark/0.16.3
module load fastqc/0.11.5
module load trim-galore/0.4.1
module load cutadapt/1.2.1

cd /scratch2/tr1915/full_data/raw

bismark --genome /scratch2/tr1915/full_data/ref/ *E10K*
```

Listing 1: Job used to implement the Bismark Bisulfite Mapper

C.1 Python

In python, many algorithms used were based on Map Reduce. In particular, Python was used to reduce the dimensionality, remove the duplicated reads and slice the data into the 50bp Segment Data-sets. Since, the python code used is similar, only one particular copy of the code will be provided in this section.

```
#!/usr/bin/env python
from __future__ import division
from operator import itemgetter
import sys

# input comes from the output of binary_1.py -- the file all_binary_sort.txt in HPC
# Introducing some variables needed in the analysis
chrom = ""  # Chromosome
seq = ""  # Sequence
start_pos = 0 # segment Starting position
end_pos = 0   # segment End position
binary = ""   # Binary read
count_patterns = 1    # counter for patterns of the same segment
num_observations = 1  # counter for observations of the same segment
count_specific_pattern = 1  # counter for observations of the same pattern
counter = range(0,105)  # For calculations
```

```
measure = 0   # Loss measure
stren_measure = 0   # Strength measure, i.e. 1-Loss
loss = 0  # final loss value
strength = 0  # final strength value
rev_count = 0 # last counter

for line in sys.stdin:
  # remove leading and trailing whitespace
  line = line.strip()
  # split the line into words
  words = line.split()
  current_chrom = words[0]
  current_start = words[1]
  current_end = words[2]
  current_bin = words[3]
  current_seq = words[4]

  try:
    current_end = int(current_end)
    current_start = int(current_start)
  except ValueError:
    continue


  # Count the number of patterns for a specific fragment as given in the data
  # Need to check if the start position differs
  # More IMPORTANTly, we need to check is the binary read differs with ANY of the
    previously
  # read binaries for the specified fragment (end position)

  ## ASSUME that the binary are sorted as they should
  # Check if same chromosome
  if (current_chrom == chrom):
    # Check if same end position in sequence
    if (current_end == end_pos):
      # check if the start position is different
      num_observations += 1
      if (current_bin != binary):
        ## NEW pattern is observed
        # Print previous pattern
        print current_chrom, "\t", end_pos, "\t", binary, "\t", loss,"\t", strength, "\t",
    count_specific_pattern
        loss = 0
        strength = 0
        # Add the new pattern to the counter
        count_patterns += 1
        # Compare previous binary with current
        split_current_bin = list(current_bin)
        split_bin = list(binary)
        # All observations are of 100bp
        counter = range(0, len(split_current_bin))
        if (len(split_current_bin) == len(split_bin)):
          for value in counter:
            # if the value is "-", we do nothing
            if (split_current_bin[value] != "-"):
              if (split_bin[value] !=   split_current_bin[value]):
```

```
                measure += 1
              else:
                stren_measure += 1
            else:
              # count the number of non-intersections
              rev_count += 1
          loss = measure/(100-rev_count)  ## finalised "loss" value
          # instead of a loss function, a "strength" value - i.e. the higher the value the more similar
          strength = stren_measure/(100-rev_count)
          measure = 0
          stren_measure = 0
          rev_count = 0
          count_specific_pattern = 1
        binary = current_bin
      else:
        count_specific_pattern += 1
    else:
      print current_chrom, "\t", end_pos, "\t", binary, "\t", loss,"\t", strength, "\t", count_specific_pattern
      print current_chrom, "\t", start_pos, "\t", end_pos, "\t", count_patterns, "\t", num_observations
      # Initialise values
      count_patterns = 1
      num_observations = 1
      loss = 0
      strength = 0
      seq = current_seq
      start_pos = current_start
      end_pos = current_end
      count_specific_pattern = 1
      measure = 0
      stren_measure
      rev_count = 0
      binary = current_bin
      # Reset the chromosome
  else:
    # Initialise values
    chrom = current_chrom
    seq = current_seq
    start_pos = current_start
    end_pos = current_end
    count_patterns = 1
    num_observations = 1
    count_specific_pattern = 1
    measure = 0
    stren_measure = 0
    loss = 0
    strength = 0
    rev_count = 0
    binary = current_bin
```

Listing 2: Example of a python code used in a Map Reduce based algorithm

C.2 R

After the pre-processing was performed, and the data structure was complete, the focus lied in R. The total analysis was performed in R mostly through various existing packages. These packages include the following: (i) data.table [27], (ii) cluster [28], (iii) dendextend [29], (iv) MASS [30], (v) glmnet [31], (vi) scatterplot3d [32], (vii) shiny [33], (viii) igraph [34] and other.

Let us begin with the function used in order to create the two Segment plots.

```
# Input would be the Binary Confined Data-set
binary_plot <- function(x){
  table_end <- table(x[,3])
  y <- x[x$`End Position` %in% as.numeric(names(which(table_end >5))),]
  x <-y
  # Initialise variables:
  sep <- vector()
  bin <- x[,4] # Binary read
  bin <- as.matrix(bin)
  bin <- as.vector(bin)
  new_bin <- vector()
  neww <- matrix(ncol = 99) # 100bp - Before slicing to 50bp
  num_patterns <- vector()
  test <- 0
  # Different number for different methylation state
  # Different number will be represented by different shade of blue
  for (i in 1:nrow(x)){
    if (x[i,3] == test){
      split_bin <- unlist(strsplit(x = bin[i],""))
      for (j in 1:99){
        if (split_bin[j] == "0"){ # Unmethylated
          new_bin <- c(new_bin, 0)
        } else if (split_bin[j] == "1"){ # Methylated
          new_bin <- c(new_bin, 1)
        } else {        # Non-cytosine
          new_bin <- c(new_bin, 2)
        }
      }
      neww <- rbind(neww, new_bin)
      new_bin <- vector()
    }else{
      if (i == 1){
        sep <- c(sep, i)
      } else {
        sep <- c(sep, i-1) # The last value in the above fragment
      }
    # Add two rows of lighter blue in order to separate the segment reads
    neww <- rbind(neww, rep(4,99))
    neww <- rbind(neww, rep(4,99))
    test <-x[i,3]
    }
  }
  neww <- neww[-1,]
  nrow(x)
  rownames(neww) <- seq(1,nrow(neww),1)
  library(ggplot2)
  library(reshape2)
```

```
  # Create the plot using package ggplot2
  melted <- melt(neww)
  plott <- ggplot(melted, aes(x = Var2, y = Var1, fill = value, colour = "Methylation State")) + geom_tile() +
     theme(text=element_text(family="Garamond", size=14)) +  ggtitle("Segment Plot") +
     xlab("Position in the segment") + ylab("")
  sep <- c(sep, nrow(x))
  for (j in 2:length(sep)){
    np <- nrow(unique(x[sep[j-1]:sep[j],4]))
    if (length(num_patterns) == 0){
      num_patterns <- c(num_patterns, np)
    } else {
      num_patterns <- c(num_patterns, np-1)
      ## Minus 1 , since the sep includes the last value in the specified fragment. Hence it calculates in the new fragment, one of the previous fragment as well
    }
  }
  # the function returns
  return(list("Plot" = plott, "Positions_of_Separation" = sep, "Number_of_patterns" = num_patterns))
}
```

Listing 3: R code used to create the Segment plot

In the following code in R the code used in creating the Combine Segment plot is provided.

```

library(ggplot2)
library(Cairo)   # For nicer ggplot2 output when deployed on Linux
library(shiny)
data <- patterns_k
# Introduce a new function: Input being the start position of the plot,
# the starting distance and the data from the Pattern Confined Data-set
fragment_representation <- function(start_position, distance, data){
  # Initialise values
  count <- 1
  plot_x <- vector()
  col.pat <- vector()
  plot_y <- vector()
  check <- data[,2]
  check <- as.data.frame(check)
  check <- check[,1]
  check <- as.numeric(as.vector(check))
  end_position <- start_position + distance
  for (i in 1:length(check)){
    if ((check[i] <= end_position) && (check[i] >= start_position)){
      plot_y <- c(plot_y, rep(count, 99))
      count <- count + 1
      # The following decreasing sequence is done due to the fact that for
      # some sequences, the starting is not "correct", but the one which
      # intersects with the previous one. However, the ending position, is
      # always the one provided from the raw data.
      plot_x <- c(plot_x, seq(as.numeric(data[i,3]), as.numeric(data[i,3]-98),-1))
      if (data[i,5] == 1){
        col.pat <- c(col.pat, rep("red", 99))
      } else if (data[i,5] <= 10){
        col.pat <- c(col.pat, rep("black", 99))
```

```
      } else if (data[i,5] <= 100){
        col.pat <- c(col.pat, rep("blue", 99))
      } else {
        col.pat <- c(col.pat, rep("green", 99))
      }
    }
  }
  plot(plot_x, plot_y, col = col.pat, ylab = "", xlab = "Position in the segment",
  main = "Segment Plot")
}
# Preparing the shiny plot
ui <- fluidPage(
      sliderInput(inputId = "dist",
      label = "Choose the distance of the fragment",
      value = 1000, min = 1, max = 5000
),
# Input in shiny plot
numericInput(inputId = "start",
      label = "Starting Position",
      value = 3007000, min = min(as.numeric(as.vector(as.data.frame(data[,3])[,1]))),
      max = max(as.numeric(as.vector(as.data.frame(data[,3])[,1])))
),
# Output of shiny plot
plotOutput("fragment")
)
# Function provides a shiny plot
server <- function(input, output) {

output$fragment <- renderPlot({fragment_representation(start_position = input$start,
      distance = input$dist,
      data = data)})
}
shinyApp(server = server, ui = ui)
```

Listing 4: R code used in the Combine Segment plots (Shiny plot)

The following function involves every algorithm considered in the analysis of the data. Different parts of the function were implemented at different times, but the final results were given based on the following code.

```
# Introduce a function that produces most of the analysis' results seen
# Input: the two data-sets in comparison from the 50b Segment Data-sets, and the end
    position of the segment to consider
finalclusteringpca <- function(data_1, data_2, position){
  # Take the binary reads
  bin_mat_k <- data_1[which(data_1[,2] == position),3:52]
  bin_mat_w <- data_2[which(data_2[,2] == position),3:52]
  bin_mat <- rbind(bin_mat_k, bin_mat_w)
  # Calculate the Wilcoxon signed-rank test, only those with p-value > 0.05 are being
    considered
  w.p <- wilcox.test(colMeans(bin_mat_k), colMeans(bin_mat_w), paired = TRUE)
  # Do the PCA
  pc1 <- prcomp(bin_mat)
  colour_labels <- c(rep("black", nrow(bin_mat_k)), rep("red", nrow(bin_mat_w)))
  # Plot the visualisation of the 2 first PCs]
  plot <- plot(pc1$x[,1], pc1$x[,2], pch = 19, col = colour_labels,
```

```
    main = paste("PCs on ", (position-49),"-", position),
    xlab = "PC1", ylab = "PC2")
barpl_first <- barplot(colMeans(bin_mat_k) - colMeans(bin_mat_w), main = "Barplot")
obs = c(nrow(bin_mat_k), nrow(bin_mat_w))

# Split to clusters based on K-medoids after the PCA has been implemented:
clarax <- clara(as.data.frame(pc1$x[,1:2]), 2, samples=50)
clarax$data <- -clarax$data
clusplott <- clusplot(clarax, main = "Clustering with K- medoids", xlab = "PC1",
ylab = "PC2")
labels <- c(rep(0,nrow(bin_mat_k)),rep(1,nrow(bin_mat_w)))
distan <- dist(clarax$medoids)
# Initialise values on comparing the clusters produced
barp <- 0
diff_clus <- 0
wilc.p <- 0
if (distan >= 1){
  # Separate the two clusters
  clus1 <- bin_mat[which(clarax$clustering == 1),]
  clus1_labels <- labels[which(clarax$clustering == 1)]
  #calculate number of w over total number in clus1:
  clus1_perc <- sum(clus1_labels)/length(clus1_labels)
  clus2 <- bin_mat[which(clarax$clustering == 2), ]
  clus2_labels <- labels[which(clarax$clustering == 2)]
  clus2_perc <- sum(clus2_labels)/length(clus2_labels)
  if (length(clus1) > 50){
    clus1av <- colMeans(clus1)
  } else{
    clus1av <- clus1
  }
  if (length(clus2) > 50){
    clus2av <- colMeans(clus2)
  } else{
    clus2av <- clus2
  }
  # A barplot showing the difference between the two clusters' results
  #barp <- barplot(clus1av - clus2av)
  diff_clus <- clus1av - clus2av
  wilc <- wilcox.test(clus1av, clus2av, paired = TRUE)
  wilc.p <- wilc$p.value
}

## Implement the LASSO regression
## SPLIT TRAIN AND TEST
smp_size <- floor(0.75 * nrow(bin_mat))
set.seed(123)
# KO type has always less samples
# Over-sampling k data-set:
t <- nrow(bin_mat_w) - nrow(bin_mat_k)
tim <- t/nrow(bin_mat_k)
new_k <- bin_mat_k
for (i in 1:floor(tim)){
  new_k <- rbind(new_k, bin_mat_k)
}
new_k <- rbind(new_k, bin_mat_k[1:floor(nrow(bin_mat_k)*(tim - floor(tim))),])
labels <- c(rep(0,nrow(new_k)),rep(1,nrow(bin_mat_w)))
```

```
bin_mat <- rbind(new_k, bin_mat_w)
train_ind <- sample(seq_len(nrow(bin_mat)), size = smp_size)
train_labels <- labels[train_ind]
test_labels <- labels[-train_ind]
train <- bin_mat[train_ind, ]
test <- bin_mat[-train_ind, ]
## Implement LASSO
if (length(clus1) ==50 | length(clus2) == 50){
  overall <- 0
  prec <- 0
  rec <- 0
  f_meas <- 0
}else{
  system.time(out.cv.lasso <- cv.glmnet(as(train, "dgCMatrix"), train_labels, family="
  binomial",
  type.measure = "mse"))
  # alpha = 1 for ridge and alpha = 0.05 for elastic-net
  out.cv.fit.lasso <- out.cv.lasso$glmnet.fit
  beta.lasso <- out.cv.fit.lasso$beta # Coefficients for all lambdas and all SNPs
  beta.lasso <- as.matrix(beta.lasso)
  nzero.lasso <- out.cv.lasso$nzero #Non-zero coefficients
  lambda <- which(out.cv.lasso$lambda == out.cv.lasso$lambda.min)
  ## Prediction:
  pred.1.lasso <- predict(out.cv.fit.lasso, as(test,"dgCMatrix"), type ="class")
  mode(pred.1.lasso) = "numeric"
  # Initialise values for performance measures
  misClasificError.lasso <- c()
  predictionScore.lasso <- c()
  tp = 0
  tn = 0
  fp = 0
  fn = 0
  # Calculations for confusion matrix
  for (i in 1:nrow(pred.1.lasso)){
    if (as.numeric(pred.1.lasso[i,lambda]) == 1){
      if (test_labels[i] == 1){
        tp = tp + 1
      } else if (test_labels[i] == 0){
        fp = fp +1
      }
    }else if (as.numeric(pred.1.lasso[i,lambda]) == 0){
      if (test_labels[i] == 0){
        tn = tn + 1
      } else if (test_labels[i] == 1){
        fn = fn +1
      }
    }
    misClasificError.lasso[i] <- pred.1.lasso[i,lambda] != test_labels[i]
    predictionScore.lasso[i] <- pred.1.lasso[i,lambda] == test_labels[i]
  }
  # create the confusion matrix
  conf <- matrix(nrow = 2,ncol = 2)
  conf[1,1] <- tn
  conf[1,2] <- fp
  conf[2,1] <- fn
  conf[2,2] <- tp
```

```
    prec <- tp/(tp + fp)
    rec <- tp/(tp + fn)
    f_meas <- 2*((prec*rec)/(prec+rec))
    rownames(conf) <- c("Actual_0", "Actual_1")
    colnames(conf) <- c("Prediction_0", "Prediction_1")
    overall <- sum(predictionScore.lasso)/nrow(pred.1.lasso)
  }
  # Implement Hierarchical Clustering
  bin_mat <- as.data.frame(bin_mat)
  distance_dat=dist(bin_mat,method="euclidean")
  h=hclust(distance_dat,method="single")
  dend <- as.dendrogram(h)
  bin_mat_k <- as.data.frame(bin_mat_k)
  bin_mat_w <- as.data.frame(bin_mat_w)
  groupcodes <- c(rep("k", nrow(bin_mat_k)), rep("w", nrow(bin_mat_w)))
  colorcodes <- c(k = "black", w = "red")
  labels_colors(dend) <- colorcodes[groupcodes][order.dendrogram(dend)]
  par(cex = 1)
  # Plot the HC
  hc_plot <- plot(dend, horiz = TRUE, main = "Histogram")

  # Return all of the above:
  return(list("pc1" = pc1, "plot" = plot,
      "Observations_k_w" = obs, "prediction_rate" = overall,
      "distance" = distan, "difference" = diff_clus,
      "barplot" = barp, "wilcoxon_p" = wilc.p, "clarax" = clarax,
      "wilcoxon_bin" = w.p$p.value, "clus1_pec" = clus1_perc,
      "clus2_perc" = clus2_perc, "confusion_matrix" = conf,
      "precision" = prec, "recall" = rec, "f_measure" = f_meas,
      "hc_plot" = hc_plot))
}
```

Listing 5: R code used to analyse the data

The function developed for the creation of the local Network is shown below.

```

# Implement the Network
# Input: Binary Confined Data-set
network_local <- function(net_read){
  # Using igraph, start off by making an empty graph
  g <- make_empty_graph(directed = FALSE, n = nrow(net_read)) %>%
  set_vertex_attr("name", value = seq(1,nrow(net_read),1))
  check <- as.numeric(as.vector(as.data.frame(net_read[1,2])[,1]))
  wid2 <- as.data.frame(net_read[1,4])
  wid2 <- as.numeric(as.vector(wid2[,1]))*15
  add_width <- wid2
  size_node <- as.data.frame(net_read[1,6])
  size_node <- as.numeric(as.vector(size_node[,1]))
  add_size <- size_node
  node_list = 1
  for (i in 2:nrow(net_read)){
    if (net_read[i,2] == check){
      wid2 <- as.data.frame(net_read[i,4])
      wid2 <- as.numeric(as.vector(wid2[,1]))*15
      add_width <- c(add_width, wid2)
      node_list <- c(node_list, i)
```

```
    }else{
      # Add a path graph
      g = g + path(node_list, width = add_width, color = "blue")
      node_list = i
      wid2 <- as.data.frame(net_read[i,4])
      wid2 <- as.numeric(as.vector(wid2[,1]))*15
      add_width <- wid2
      check = net_read[i,2]
    }
  }
  # Create the plot:
  add_size <- as.data.frame(net_read[,6])
  add_size <- as.numeric(as.vector(add_size[,1]))
  plot(g,vertex.label=V(g)$names,
  edge.arrow.size=0.01,vertex.label.color = "black")#,vertex.size=add_size)
}
```

Listing 6: R code used to create the local network

References

[1] Christoph Bock. *Analysing and interpreting DNA methylation data*. Nature Reviews, Genetics, 2012.

[2] Peter W. S. Hill, Rachel Amouroux, Zhiyi Sun, Jolyon Terragni, Romas Vaisvila, Sarah Linnett, Harry G. Leitch, Gopuraja Dharmalingham, Vanja Haberle, Boris Lenhard, Yu Zheng, Sriharsa Pradhan, Petra Hajkova *Germline epigenetic reprogramming licences the meiotic programme* To be published.

[3] B.Alberts, A.Johnson, J.Lewis, M.Raff, K.Roberts, P.Walter. *Molecular Biology of the cell*. Garland Science, 2008.

[4] L.D.Moore, T.Le, G.Fan. *DNA Methylation and its function*. American College of Neuropsychopharmacology, 2012.

[5] Y.Li, T.O.Tollefsbol. *DNA methylation detection: Bisulfite genomic sequencing analysis*. Methods of Molecular Biology, 2011.

[6] Hehuang Xie, Min Wang, Alexandre de Andrade, Maria de F. Bonaldo, Vasil Galat, Kelly Arndt, Veena Rajaram, Stewart Goldman, Tadanori Tomita and Marcelo B. Soares. *Genome-wide quantitative assessment of variation in DNA methylation patterns*. Nucleic Acids Research, 2011.

[7] Jianlin He, Xinxi Sun, Xiaojian Shao, Liji Liang and Hehuang Xie. *DMEAS: DNA methylation entropy analysis software*. Bioinformatics, Advance Access publication June 8, 2013.

[8] Herve Abdi, and Lynne J. Williams *Principal Component Analysis* John Wiley & Son s, Inc, 2010, pp.433-459.

[9] Felix Krueger *Bismark Bisulfite Mapper* Babraham Bioinformatics Group, May 2017

[10] Bernard Rosner, Robert J. Glynn and Mei-Ling T. Lee *The Wilcoxon Signed Rank Test for Paired Comparisons of Clustered Data*. Biometrics 62, pp. 185?192, Mar 2006

[11] Briti Deb and Satish Narayana Srirama. *Parallel K-Means Clustering for Gene Expression Data on SNOW*. International Journal of Computer Applications, June 2013.

[12] Chris Ding and Xiaofeng He. *K-means Clustering via Principal Component Analysis*. Proceedings of the 21st International Confer- ence on Machine Learning, Banff, Canada, 2004.

[13] Monika Kastner, Julie Makarski, Leigh Hayden, Lisa Durocher, Ananda Chatterjee, Melissa Brouwers and Onil Bhattacharyya. *Making sense of complex data: a mapping process for analyzing findings of a realist review on guideline implementability*. BMC Medical Research Methodology 2013.

[14] Peter W. Laird. *Principles and challenges of genomewide DNA methylation analysis*. Nature Reviews, Genetics, 2010.

[15] Eugene Andres Houseman, John Molitor and Carmen J. Marsit. *Reference-free cell mixture adjustments in analysis of DNA methylation data*. Bioinformatics, Advance Access publication January 21, 2014.

[16] Dongmei Li, Zidian Xie, Marc Le Pape and Timothy Dye. *An evaluation of statistical methods for DNA methylation microarray data analysis*. BMC Bioinformatics (2015).

[17] Kristen H. Taylor, Robin S. Kramer, J. Wade Davis, Juyuan Guo, Deiter J. Duff, Dong Xu, Charles W. Caldwell and Huidong Shi. *Ultradeep Bisulfite Sequencing Analysis of DNA Methylation* Cancer Res 2007.

[18] Robert Tibshirani. *Regression Shrinkage and Selection via the Lasso*. Journal of the Royal Statistical Society. Series B(Methodological), Volume 58, Issue 1 (1996), pp. 267-288.

[19] K. V. Mardia, J. T. Kent, J. M. Bibby. *Multivariate Analysis*. Tenth printing, 1995.

[20] Trevor Hastie, Robert Tibshirani, Jerome Friedman. *The Elements of Statistical Learning*. Second Edition (2008).

[21] Xinghao Pan, Dimitris Papailiopoulos, Samet Oymak, Benjamin Recht, Kannan Ramchandran, Michael I. Jordan. *Parallel Correlation Clustering on Big Graphs*. EECS at UC Berkeley,July 2015.

[22] Shai Bagon, Meirav Galun. *Optimizing Large Scale Correlation Clustering*. The Weizmann Institute of Scince Rehovot 76100, Israel, Dec 2011.

[23] Luc Anselin. *Local Indicators of Spatial Association-LISA*. Geographical Analysis, Vol. 27, No.2 (April 1995).

[24] Gafar Matanmi Oyeyemi, Eyitayo Oluwole Ogunjobi, Adeyinka Idowu Folorunsho *On Performance of Shrinkage Methods ? A Monte Carlo Study*. International Journal of Statistics and Applications 2015.

[25] Vanna Albieri, Vanessa Didelez. *Comparison of statistical methods for finding network motifs*. Statistical applications in genetics and molecular biology (2014), pp. 403?422

[26] Nikhil Bansal, Avrim Blum, Shuchi Chawla. *Correlation Clustering*. Department of Computer Science, Carnegie Mellon University, this research was supported in part by NSF grants CCR-0085982, CCR-0122581, CCR- 0105488, and an IBM Graduate Fellowship.

[27] Matt Dowle, Arun Srinivasan, Jan Gorecki, Tom Short, Steve Lianoglou, Eduard Antonyan. *Package data.table*. CRAN, Feb 2017, https://cran.r-project.org/web/packages/data.table/data.table.pdf

[28] Martin Maechler, Peter Rousseeuw, Anja Struyf, Mia Huber, Kurt Hornik, Matthias Studer, Pierre Roudier, Juan Gonzalez. *Package cluster*. CRAN, Mar 2017, https://cran.r-project.org/web/packages/cluster/cluster.pdf

[29] Tal Galili et al. *Package dendextend*. CRAN, Mar 2017, https://cran.r-project.org/web/packages/dendextend/dendextend.pdf

[30] Brian Ripley, Bill Venables, Douglas M. Bates, Kurt Hornik, Albrecht Gebhardt, David Firth. *Package MASS*. CRAN, April 2017, https://cran.r-project.org/web/packages/MASS/MASS.pdf

[31] Jerome Friedman, Trevor Hastie, Noah Simon, Rob Tibshirani. *Package glmnet*. CRAN, May 2017, https://cran.r-project.org/web/packages/glmnet/glmnet.pdf

[32] Uwe Ligges, Martin Maechler, Sarah Schnackenberg. *Package scatterplot3D*. CRAN, April 2017, https://cran.r-project.org/web/packages/scatterplot3d/scatterplot3d.pdf

[33] Winston Chang et al. *Package shiny*. CRAN, Aug 2017, https://cran.r-project.org/web/packages/shiny/shiny.pdf

[34] *Package* ***igraph***. CRAN, July 2017, https://cran.r-project.org/web/packages/igraph/igraph.pdf