University of Edinburgh
School of Informatics

Clustering and Visualisation of DNA Sequencing Data

Undergraduate Dissertation
BSc Artificial Intelligence

Saulius Lukauskas

April 4, 2013

Abstract: Next Generation Sequencing methods, such as ChIP-Seq, have already become vitally important for studying epigenetic processes. Previous research has shown that visually observable patterns in the data produced by these methods can yield insights to the underlying biological mechanism, yet the enormous scale and specifics of this data makes the task of exploration challenging. This project proposes Dynamic Genome Warping (DGW), a novel method of exploring this data that combines Dynamic Time Warping, a well-established method coming from speech recognition community, with hierarchical clustering. The method is shown to be able to recognise well-documented features in ChIP-Seq datasets automatically, as well as to provide means to discover previously unseen ones by exploring data.

Acknowledgements

I would like to thank Gabriele Schweikert and my supervisor, Guido Sanguinetti, for providing constructive discussions and support required to keep pushing this project past all the limits.

Contents

1. Introduction

The DNA sequence of a living organism encodes the complete blueprint required to create the said organism. An emergent picture is that the mechanisms controlling how this blueprint is used are as important as the DNA sequence is.

Next-generation sequencing methods, such as ChIP-Seq have already become vitally important for studying these epigenetic processes. These new methods usually produce large datasets of complex patterns, such as a series of peaks that extend over a long region of the DNA, complicating the data exploration task.

This project proposes a novel method of analysing this data that combines well-established methods from speech recognition community with hierarchical clustering. This method aids the data exploration by grouping similar patterns together and by providing visualisations that simplify the complicated structures described earlier.

This report is structured as follows: chapter 2 provides the necessary biological background required to be able to understand the specifics of this method. The formal motivation and goals of the project are provided in the subsequent chapter 3. The report then proceeds to describe the method (chapter 4) as well as the details of the implementation (chapter 5) in depth. This method is then evaluated in chapter 6 and the report is concluded with the directions for further work in the final chapter 7.

2. Biological Background

This chapter summarises the biological background required to get to grips on what this honours project is about. A more in depth review of the topics discussed in this chapter could be found in molecular biology textbooks, for instance in [AJL+02].

2.1 DNA

The complete set of information required to create and maintain cells of a living organism is contained in the genome of the said organism. In humans, this information is encoded in a long chain of nucleotides that form the DNA molecules. These chains of nucleotides can be sequenced by determining the order of appearance of the four bases within the genome: adenine (often abbreviated as simply "A"), cytosine (abbreviated as "C"), guanine ("G") or thymine ("T"). This sequence of nucleobases is what is commonly referred to as the DNA Sequence.

Generally, DNA molecules are observed in pairs of two tightly connected molecules known as strands. These strands are held together by hydrogen bonds between the nucleobases: guanine is known to form a bond with cytosine, whereas adenine forms a bond with thymine, as is illustrated in Figure 2.1.

Both strands have opposing chemical polarities, determined by the position of phosphate group (marked as yellow P in the figure) relative to the position of sugar group (deoxyribose in DNA, marked as orange pentagon). The end of the strand with a phosphate group is generally referred to as $5'$ end (the "five prime end"), whereas the other end is referred to as $3'$ end, as labeled in the figure.

Since each kind of nucleobase can form a bond with only one other kind of nucleobase, both of the DNA strands contain enough information to recreate the other on its own, a crucial property that allows DNA to replicate.

Because of this redundancy, the sequences on both strands can be sequenced simultaneously, therefore sequences of nucleotides are often referred to in terms of base pairs (abbreviated as bp.), rather than individual nucleotides. Human sex cells are known to contain around three billion such base pairs, other cells containing twice the amount [Ann08].

Since each base pair in the DNA is around 0.34 nanometers long [Ann08], DNA sequence of a human diploid cell will take up around 2 meters in length, if stretched, yet it is packaged into a volume as small as the nucleus of a cell. This packaging

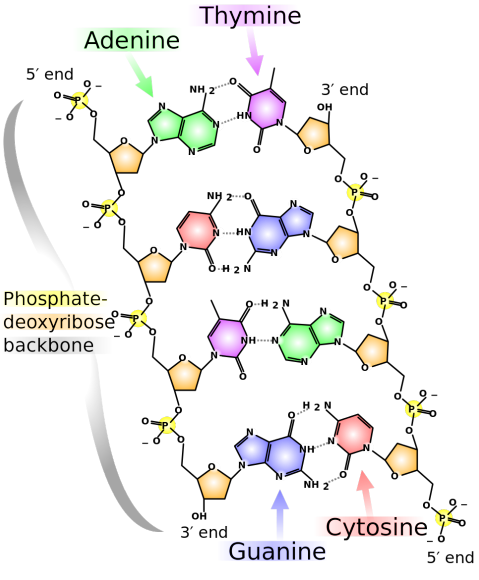


Figure 2.1: Chemical structure of the DNA molecule. The two strands of DNA are being held together by the hydrogen bonds between adenine and thymine or guanine and cytosine, as pictured. Note that the strands have opposing chemical polarities, the end of a strand with a phosphate group (yellow) is often referred to as the 5′ end, whereas the end with a deoxyribose group (orange) is commonly referred to as the 3′ end.

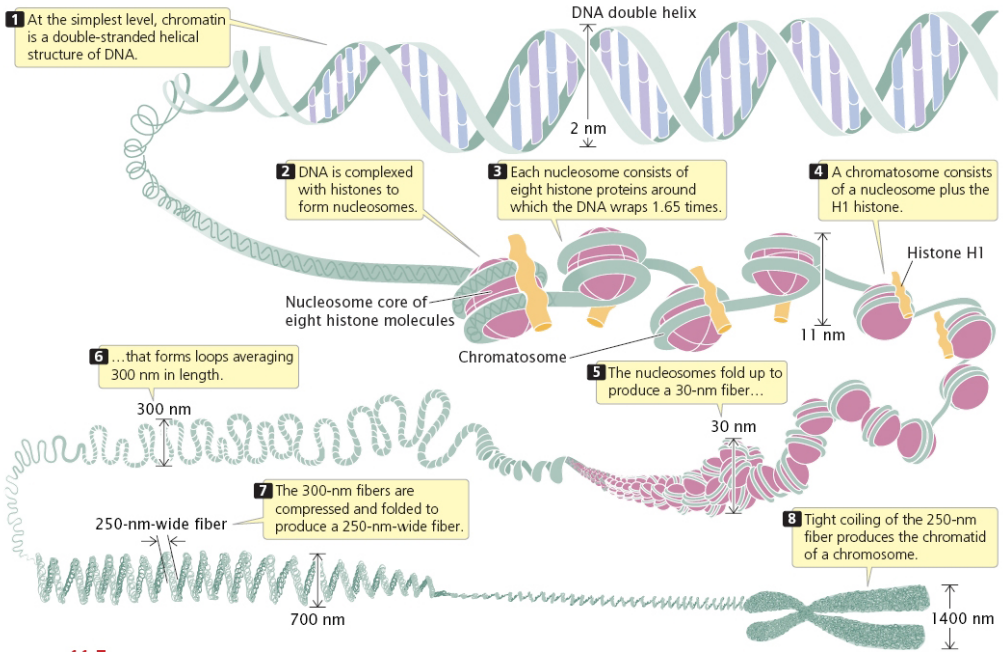


Figure 2.2: Infographics of DNA Packaging mechanism.

is done with the help of proteins called histones that fold the DNA around them, eventually wrapping it into three dimensional structures.

More specifically, DNA wraps around an octomer of two copies of each of the four core histones, creating a structure resembling beads on a string that is pictured in panels 2 and 3 of Figure 2.2. These core histones are known as histones H2A, H2B, H3, and H4. The fifth histone, H1, attaches itself to the DNA from the opposite side to the other histones, as pictured in yellow in the panel 4 of figure 2.2. The segment of DNA and the histone it is wound around, is known as nucleosome. These nucleosomes are then further wrapped into a structure known as the 30-nanometer fibre, which is then wrapped to even higher order structures, as seen in Figure 2.2. The warped combination of DNA and the proteins inside the nucleus of a cell is commonly referred to as the chromatin.

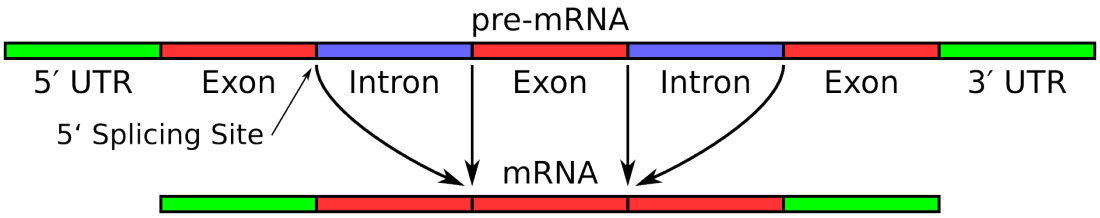
These DNA-wrapping histones are known to be mostly modular in structure, with a number of "tails" that are susceptible to chemical modifications such as methylation, acetylation and phosphorylation [FWA03; Kou07]. These chemical modifications on histone tails have been shown to be able to alter the structure of chromatin. For instance, acetylation of histone H4 on lysine 16 (shortened as H4K16Ac, where *H4* is the histone number, *K16* stands for lysine 16 and *Ac* stands for Acetylation) has been shown to inhibit formation of the 30-nanometer fibre [SK06]. These changes of the chromatin structure can affect the accessibility of the DNA. Intuitively, the tighter DNA is packaged, the less accessible it is to various chemical interactions.

2.2 Gene Expression

Gene expression process can be summarised using the central dogma of molecular biology: *DNA makes RNA makes proteins.* Technically speaking, gene expression is a combination of two processes, transcription (DNA to RNA) and translation (RNA to protein) which produce a functional product of a gene, e.g. a protein[1].

During transcription, RNA polymerase molecule binds to a region of DNA that is indicated by a complex ensemble of proteins bound to promoter regions of DNA. This molecule then proceeds to read information encoded in one of the strands, synthesising RNA molecule complimentary to the nucleotides read. The strand that was read by RNA polymerase molecule is commonly referred to as the template or antisense strand, and is read in direction from $3'$ end to $5'$ end, meaning that the RNA molecule is created from $5'$ end to $3'$ end. Since the generated RNA is complimentary to the sequence on the template strand, it will

[1]Not all genes produce proteins as their functional product. For the sake of simplicity, these so called non-coding genes are not discussed in this chapter.



Modified version of https://en.wikipedia.org/wiki/File:Pre-mRNA_to_mRNA.svg. The original image was released to the public domain.

Figure 2.3: RNA splicing. The intron regions of the transcribed pre-mRNA (blue) are removed by splicing, leaving only the exon regions (red) that are then joined together. The regions between exons and introns are called splicing sites. The 5′ splicing site, also referred to as the first splicing site is marked in the figure. The green regions at the edges of both pre-mRNA and mRNA are called untranslated regions (UTR) and are included here just for completeness. These regions are not discussed any further in this project.

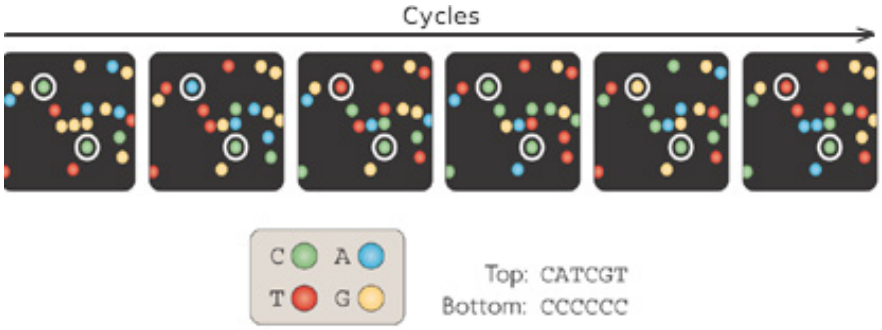
have exactly the same arrangement of nucleotides, as in the remaining strand[2]. Due to this reason the remaining strand is often referred to as the coding or sense strand. Also, the location of the first nucleotide that was transcribed is often referred to as the transcription start site of a gene or TSS for short.

Even though the complete length of the gene is transcribed into RNA, only a fraction of it is translated to the protein. Fragments of genes, known as introns, are removed post transcription by the process known as RNA splicing. The remaining regions, exons, are then joined and form the final messenger RNA (mRNA). This process is schematically illustrated in Figure 2.3. The regions between exons and introns are known as splicing sites. Please note the location of the first splicing site, otherwise known as the 5′ splicing site, that is marked in the figure as it will become important in the next chapter.

After splicing, the resulting messenger RNA is transferred to the ribosome, that will create the proteins from the blueprint encoded inside the mRNA. A more in depth view of these processes can be found in a molecular biology textbook, e.g. in [AJL+02].

Gene expression is particularly interesting for epigeneticists, who are concerned about the factors that may alter it without altering the underlying DNA sequence. Chemical histone modifications have been shown to correlate with the transcriptional activity of the genes near them [PHLC05; Kou07], making them an interesting topic of research for epigeneticists. The next section summarises the methods designed to study them.

[2]The only exception being that in RNA molecule thymine is replaced with uracyl.



Adapted by permission from Macmillan Publishers Ltd: Nature Reviews Genetics, Michael L. Metzker, *Sequencing technologies the next generation*, copyright 2010 ([Met10])

Figure 2.4: Illustration of parallel sequencing of nucleotides, as in Illumina/Solexa Sequencing platform. Each coloured dot in the figure corresponds to a different fragment of DNA being sequenced. At each cycle, colour-coded nucleotide is incorporated to DNA by a mechanism similar to DNA replication. Then imaging is performed to get the sequences of all the pieces of DNA all at once, resulting in an image similar to the one in this figure. For each DNA fragment, the colour is read and mapped back to the nucleotide. The dyes are then washed and the cycle continues [Met10].

2.3 Next-Generation Sequencing

It took 13 years and around $2.7 billion to sequence the whole human genome for the first time, by the scientists of the Human Genome Project[3]. Since then, the cost and duration of DNA sequencing has dropped sharply with the invention of the Next-Generation Sequencing methods (NGS). Instead of sequencing a long segment of DNA sequentially, the next-generation sequencing methods sequence a lot of short DNA segments in a massively-parallel way, much like as depicted in Figure 2.4.

The short-sequenced reads can then be put together via the overlapping fragments between the reads. Anyone who ever tried solving a puzzle without looking at the cover, knows how hard this process could be, even for computers. That is why NGS methods that produce very short reads are usually used not for tasks of sequencing the genome for the first time (*de-novo* sequencing), but rather to study the DNA itself.

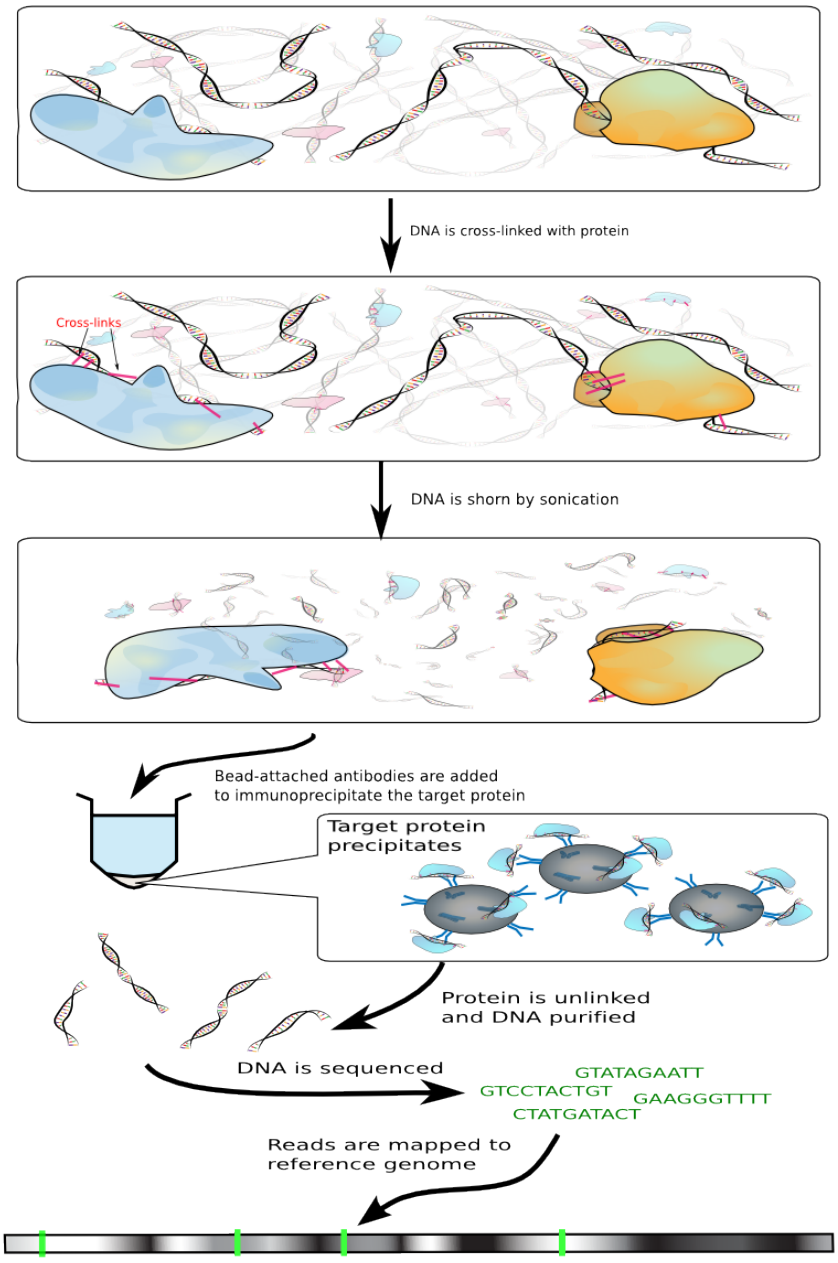


Figure 2.5: ChIP Sequencing workflow. First, the target proteins are cross-linked with DNA. This DNA-Protein complex is then shorn into smaller pieces by sonication. Antibodies for the target proteins are added to mixture in order to immunoprecipitate it. The precipitate containing the target protein is collected and purified to extract DNA. This DNA is then sequenced and mapped to reference genome.

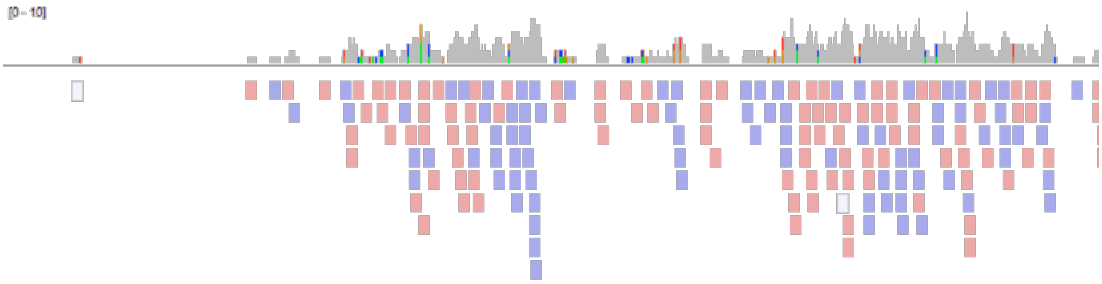


Figure 2.6: A typical result of ChIP-Seq experiment. Pictured here is the region around the transcription start site for gene *WDTC1* (chr1:27,559,006-27,563,006). The experiment targeted histone modification H3K4Me3 in cell type K562 from ENCODE dataset (see subsection 2.4.1). The aligned reads are pictured as rectangles with colour indicating the directionality. The histogram, often referred to as the, in this case H3K4Me3, mark is computed by calculating the number of reads that overlap particular base pair. The image was generated by taking a screenshot of Integrative Genomics Viewer (IGV) [TRM12].

2.4 ChIP-Sequencing

Chromatin Immunoprecipitation, followed by Sequencing (ChIP-Seq) is a NGS method that studies protein-DNA interactions within a cell. Instead of sequencing the whole genome, ChIP-Seq aims to sequence only parts of DNA that interact with some target protein. These target proteins can either be internal to DNA (histones) or arbitrary external proteins that interact with DNA.

Basic ChIP-Seq workflow is summarised in Figure 2.5. In short, proteins are first "glued" together with the DNA by chemical cross-linking. DNA with proteins attached to it is then shorn into smaller pieces by sonication. Antibodies are added to single the target proteins out using immunoprecipitation. Precipitate is then collected and the DNA that is attached to the target proteins is sequenced. This sequenced DNA is then matched to the reference genome to provide hints of which parts of genome interact with the protein [Mar07].

Figure 2.6 shows a typical result of ChIP-Sequencing experiment. The aligned reads are pictured as rectangles below the histogram. Usually only a short fragment of DNA attached to protein is sequenced. In most cases, the sequencing is done from $5'$ end to $3'$ only [Par09]. This means that the DNA that was read from the reverse strand will have opposite direction than the DNA that was read from the forward strand. This directionality is then recorded and can be used to correct some of the sequencing artefacts, as described in section 4.3.

The aligned reads are then stacked together into a histogram, creating a landscape of target protein enrichment, as seen in Figure 2.6. These enrichment landscapes,

[3]For more information, see `http://www.genome.gov/11006943` and `http://www.ornl.gov/sci/techresources/Human_Genome/project/about.shtml`

often referred to as just "marks", are then analysed computationally in order to gain insights on the role the target proteins play in the living organisms. Even though these marks are technically histograms of the number of reads falling within the region, they are commonly visualised as continuous functions, for the sake of simplicity.

2.4.1 Publicly available Datasets

Before jumping to the motivation for the project, it is important to note the publicly available datasets of ChIP-Seq experiment results.

The Encyclopedia of DNA elements, ENCODE, [The04] has compiled a large collection of histone modification data from the Broad Institute (`https://www.broadinstitute.org/`). This data is publicly available to download and use via the ENCODE website at
`http://genome.ucsc.edu/cgi-bin/hgFileUi?db=hg19&g=wgEncodeBroadHistone`.

This project uses these publicly available datasets in all of it's experiments and some of the figures. Unless explicitly stated otherwise, the ENCODE ChIP-Seq datasets for K562 leukaemia cell line are used in this project.

Whenever the annotated locations of known genes, transcription start sites, and first-splicing sites are used, they are used from the UCSC Known Genes collection [HKC+06] (hg19 assembly), and are free to download via the ENCODE table browser.

This track assigns a unique 10-character id, e.g. `uc003vhn.1`, for each of the genes and their variants. These identifiers are referenced in the text of the report whenever possible, and can be used to query the UCSC Genome browser [KSF+02] directly, if it is desired.

3. Motivation

In one of the early reviews of ChIP-Seq, [Par09], P J Park already predicted data processing to be the major challenge of this technology, and therefore listed the most common approaches to data processing at that time. Almost four years later today, most of these approaches still remain valid.

For protein-DNA binding, he writes, the most common analysis is the discovery of the short segments of DNA the protein is most likely to bind to, so called binding sequence motifs. A large variety of methods for this kind of analysis are available today, probably the best-known one being MEME [GBEa97].

In order to find these transcription factor binding sites, significantly enriched regions of the genome (so called "peaks") need to be located first. MACS peak caller [ZLM+08] is the method of choice for this task to date, even though a variety of other methods have been proposed since (see [Fur12] for a recent review).

For histone modification studies, an obvious approach, as claimed by P J Park, is to annotate the locations of these peaks relative to known genomic features. For instance, it has been observed that, on average, histone modifications have very distinct bimodal marks near transcription start sites, as pictured in Figure 3.1. It is also noted that most of the marks, e.g. H3K4me3 mark, correlate with the activity of genes being analysed: highly active genes (red in the figure) have more reads falling within the same region than intermediately active ones (green and blue) [BCC+07].

What is more interesting, these histone marks are so distinctive, that [HSH+07] were able to predict locations of regions that both initiate (promote) and enhance transcription of DNA by training a classifier on a combination of histone marks around known promoter and enhancer regions. Later, [HRW08] developed a method called ChromaSig, that was able to find the promoter and enhancer signatures in a completely unsupervised way by simply looking at the most common histone mark motifs within the genome.

However, the fact that ChIP-Seq signal can tell us more than just the type of of the DNA region is more interesting. In the already mentioned paper, [BCC+07], the authors note: "A series of peaks of H3K4me3 signals at +50, +210, and +360 were detected, suggesting similar nucleosome positioning relative to TSS in active genes"[1], these peaks can be seen in figure Figure 3.1. This claim was later followed-up by [SB07], who were able to locate promoter-specific nucleosomes "with unprecedented precision" from this ChIP-Seq data.

[1] Original text misspells *H3K4me3* as *H4K4Me3*

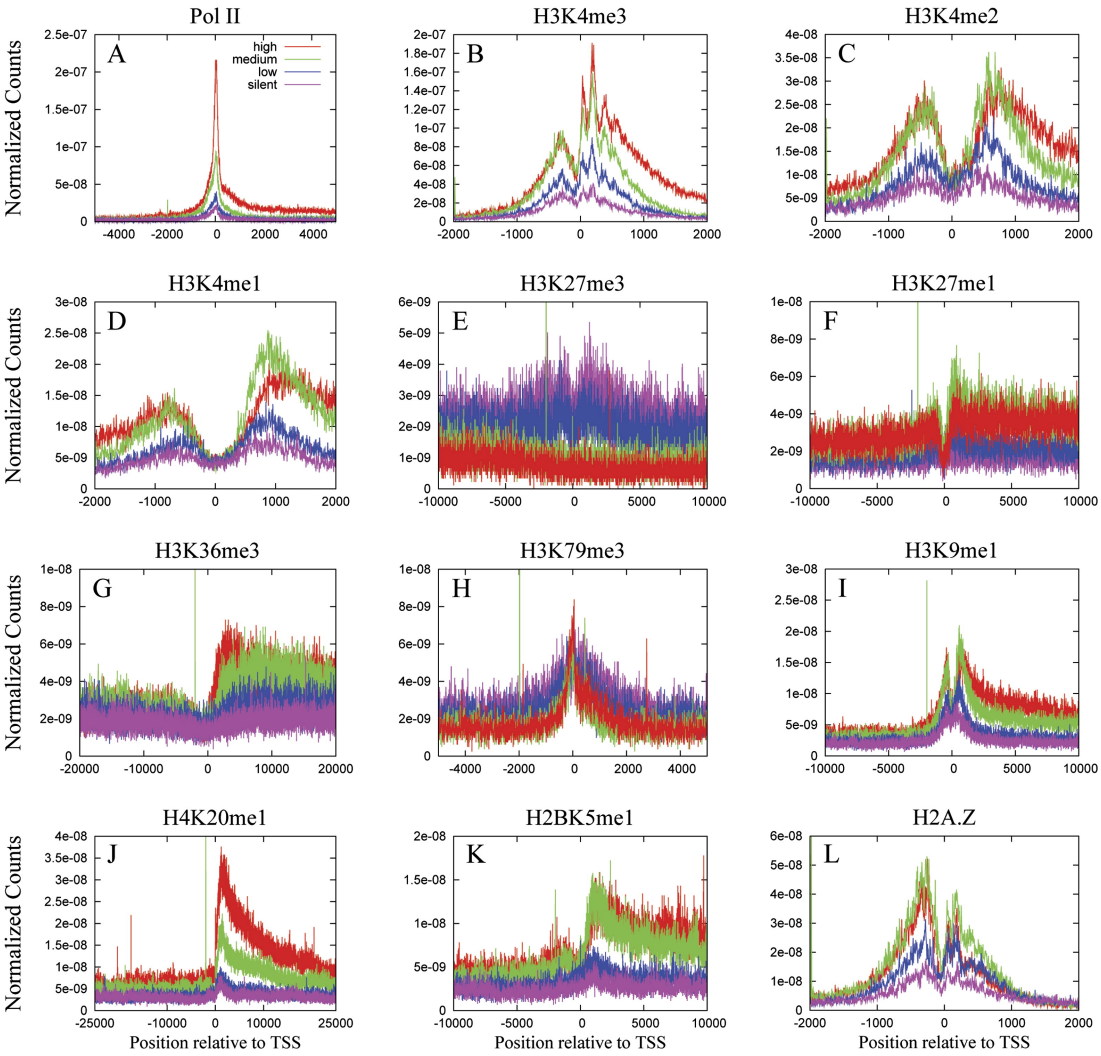


Figure 3.1: Profiles of histone marks around known transcription start sites. The profiles for highly active, two stages of intermediately active and silent genes are colour-coded. The histone modification is shown on top of each plot. "Normalized counts" on the Y axes refers to the normalised number of reads obtained from ChIP-Seq experiment that overlap a particular bin. The results were compiled by [BCC+07]. Please note a series of peaks of H3K4me3 signals at +50, +210, and +360 base pairs offset from transcriptions tart site, that, according to Barski et al., suggests similar nucleosome positioning relative to TSS in active genes.

Another, very recent paper by Bieberstein and colleagues [BOSN12] has shown that H3K4Me3 and H3K9Ac profiles tend to peak at the locations of $5'$ splicing sites of known genes. The authors also conclude that these peaks are independent of nucleosome positioning, unlike the peaks observed by [BCC$^+$07].

An important point to take away from these two results is that insights to underlying biological processes can be produced from the visual inspection of these histone marks. This observation is the main source of motivation for this project.

Having said that, the ultimate goal of this project is to develop a tool that would allow to group similar patterns into clusters and provide means to visually compare these patterns on a common ground, so hypotheses about their origin could be made.

3.1 Prior Art

There have been a few prior attempts at clustering histone marks. The aforementioned ChromaSig method ([HRW08]) aims to find the patterns that commonly occur in the genome, by aligning various segments of the genome to each other. Another paper, [LB10], uses a similar alignment method to align patterns from user-supplied regions in the genome, so they can then be visually compared on the same ground.

While both of these methods would probably succeed in detecting the patterns that occur at relatively uniform intervals, e.g. patterns that occur due to nucleosome positioning, both of them would struggle to facilitate the splicing site insights documented in [BOSN12]. The vastly varying lengths of first exons call for non-linear scaling of the genomic regions to align them.

Also, in some cases, alignment without any kind of scaling, does not make any sense at all. For instance, if one was about to compare the regions spanning whole lengths of known genes, the best these methods could do is to compare a small fragment of the longer gene to the complete length of the shorter one, which is not necessarily the correct thing to do as usually genes will have distinct patterns that occur, e.g. at either end of the gene. Not comparing the gene regions in full would mean that some of these patterns are simply ignored.

This is the reason why some studies, e.g. [TWY$^+$09], rescale the regions of interest to be of the same length, before doing clustering. This could indeed allow visualisation of similar patterns between the different regions, provided that these patterns scale linearly with the length of the regions, yet there is no theoretical ground why this would be the case.

Having said this, comparing the regions of different lengths on the same scale is the fundamental problem that this project tries to solve.

The proposed solution makes an observation that this problem of comparing genomic patterns of varying lengths is very similar to the problem of comparing waveforms of spoken words in speech recognition. For instance, it is desired to identify two words being the same even though the recording for one of them started, say half a second later. Equivalently, it is desired to detect patterns that are similar, but misaligned. Furthermore, a good speech recognition system would be able to recognise a word correctly even though the speaker prolonged one of the vowels in the word more than usual, what is similar to comparing patterns that are influenced by the length of the first exon.

Fortunately, the speech recognition community has already matured an approach to this problem. This approach, called Dynamic Time Warping (or simply, DTW), compares two sequences by shrinking or expanding both of them non-uniformly to be as similar as possible. This non-linear warping can then both account for misalignment, by shrinking the misaligned part of one of the sequences, and scaled patterns, by rescaling them to be of the same length.

This project proposes a method that combines this Dynamic Time Warping distance with hierarchical clustering with the aim of comparing genomic features. Instead of using Dynamic Time Warping to compare time-series data, it is directly used to compare the genomic data. Due to this reason, this new method is named Dynamic Genome Warping or simply DGW.

This method is similar to the ChAT method, described in [WLJ12]. This method tries to find the best alignment between two patterns before doing any clustering, just as ChromaSig or ArchAlign, yet it allows to ignore a part of some region as if it was not there. The authors of the paper claim that these gaps "are critical because they allow the algorithm to extend beyond regions with variant (or diffuse) chromatin enrichment signatures, and in so doing facilitate the discovery of chromatin signatures that span long genomic regions as well as those with complex multi-modal patterns of histone modification enrichment".

The main difference between the method used in this project and the one used by Wang et al. is that when creating gaps, ChAT algorithm applies a penalty that is proportional to the norm of the vector that is being skipped. Such penalty biases the algorithm to skip unexpressed regions more than expressed ones as per reasoning that depleted regions are not as interesting as the expressed ones. While this penalty allows the discovery of multi-modal patterns separated by depleted regions across genome, it would heavily penalise any attempts to discover patterns of varying lengths in highly-expressed regions.

Contrary to this, the DTW distance measure does not allow skipping of the points at all. It does, however, allow a set of points on one sequence to a single point on

another as if that point was stretched to become a sequence of n points. This way the penalty of warping the genome depends directly on the points that are aligned to each other, so the algorithm is able to do this both for highly expressed and fairly depleted regions, as long as the values mapped to each other are similar.

The exact workings of DGW are explained in detail in the next chapter.

4. Methods

4.1 Dynamic Time Warping

The Dynamic Time Warping (DTW) algorithm is the corner stone of this project. This algorithm was initially developed in the speech recognition community in order to allow accurate comparisons between words spoken with tempo variations.

This comparison is achieved by warping the time axis nonlinearly. That is, the axis is stretched or compressed as to minimise the total distance between the points aligned to each other. The stretching and compressing is done by creating an alignment between the two sequences, also known as the warping path, almost arbitrarily, with a few constraints described below. The distances between aligned points are then computed and added together to obtain what is commonly referred to as the DTW Distance.

More formally, given two sequences $\mathbf{a} = a_1, a_2, ..., a_n$ and $\mathbf{b} = b_1, b_2, ..., b_m$, Dynamic Time Warping aims to construct a warping path
$\mathbf{p} = (p_{1,0}, p_{1,1}), (p_{2,0}, p_{2,1}), ..., (p_{k,0}, p_{k,1})$ that maps each point on the first sequence, specified as $p_{i,0}$ to a point on the second sequence, $p_{i,1}$, such that the following three constraints hold:

1. *Boundary Condition*: The path must start at the first point of sequence $\mathbf{a}$ and this point must be mapped to the first point of the second sequence: $p_1 = (1, 1)$. Similarly, the final points of both sequences should be mapped to each other, and the path must end there: $p_k = (m, n)$.

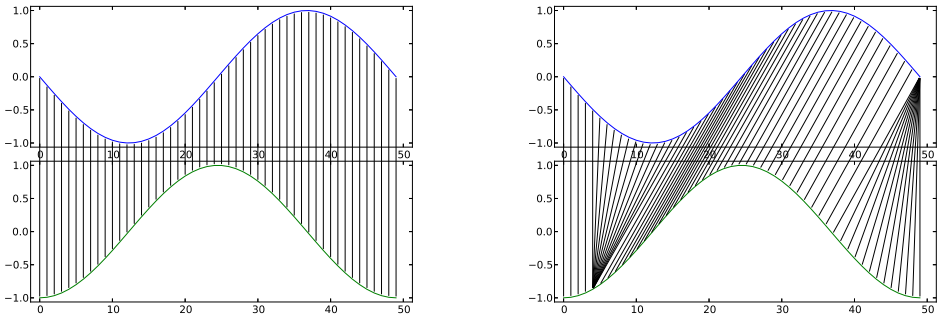
2. *Monotonicity Condition*: The path must not move *back in time*: $p_{1,i} \leq p_{2,i} \leq p_{3,i} \leq ... \leq p_{k,i}$ for both $i = 0$ and $i = 1$.

3. *Step Size Condition*: At each step the path must either map the same point on one sequence to the next point on another sequence, or map next points on both sequences together, mathematically: $p_{i+1} - p_i \in \{(0, 1), (1, 0), (1, 1)\}$ for all $i < k$.

The problem is to find a warping path that is minimal, given some distance measure d that would compute the distances between aligned points (e.g. Euclidean distance):

$$\hat{P} = \arg\min_{P} \sum_{(i,j)\in P} d(a_i, b_j) \tag{4.1}$$

Such minimal warping alignment can be computed in $O(nm)$ time by applying dynamic programming techniques.



(a) Euclidean alignment (b) DTW alignment

Figure 4.1: Two sequences aligned to each other. Figure 4.1a maps each point on sequence A to a point on sequence B directly, resulting in an Euclidean distance measure, whereas figure 4.1b uses Dynamic Time Warping algorithm to find a better mapping between the points.

Specifically, we construct a $n \times m$ matrix $\mathbf{D}$ where each entry $D_{i,j}$ will contain a minimal (warped) distance of matching first i components of the first sequence, $a_1..a_i$, to first j elements of the second sequence, $b_1..b_j$. The first component of the matrix, $D_{1,1}$ is initialised to the distance between the first components of the two sequences, $d(a_1, b_1)$ as specified by the boundary condition.

The matrix is then filled row by row, according to the following recursive equation:

$$D_{i,j} = d(a_i, b_j) + min(D_{i-1,j}, D_{i,j-1}, D_{i,j}) \tag{4.2}$$

In the end, the column $D_{n,m}$ holds the minimal warped distance between two series. If the minimal warping path is needed, it can be traced back by following the recursive relation backwards from $D_{i,j}$. This algorithm is illustrated in Figure 4.2.

More information on the workings of DTW can be found in e.g. [Mül07].

Please note that the elements of the sequences, i.e. $a_1, a_2, ..., a_n$, do not necessarily need to be scalar, as in the example. DTW is able to compute the distance between sequences of vectors, provided that the local distance measure is able to compare these vectors.

To illustrate what DTW does, have a look at the Figure 4.1. The two sequences in the figure are sine and cosine plotted over the range $[-\pi; \pi)$. Given the nature of these functions, we know that the shapes in the two sequences are essentially identical, just shifted by $\frac{\pi}{2}$ radians, therefore we should get a good alignment between them by warping the x axis.

In Figure 4.1a, each point on the sequence A is aligned to corresponding point on the sequence B. If we were to sum the distances between aligned points, we

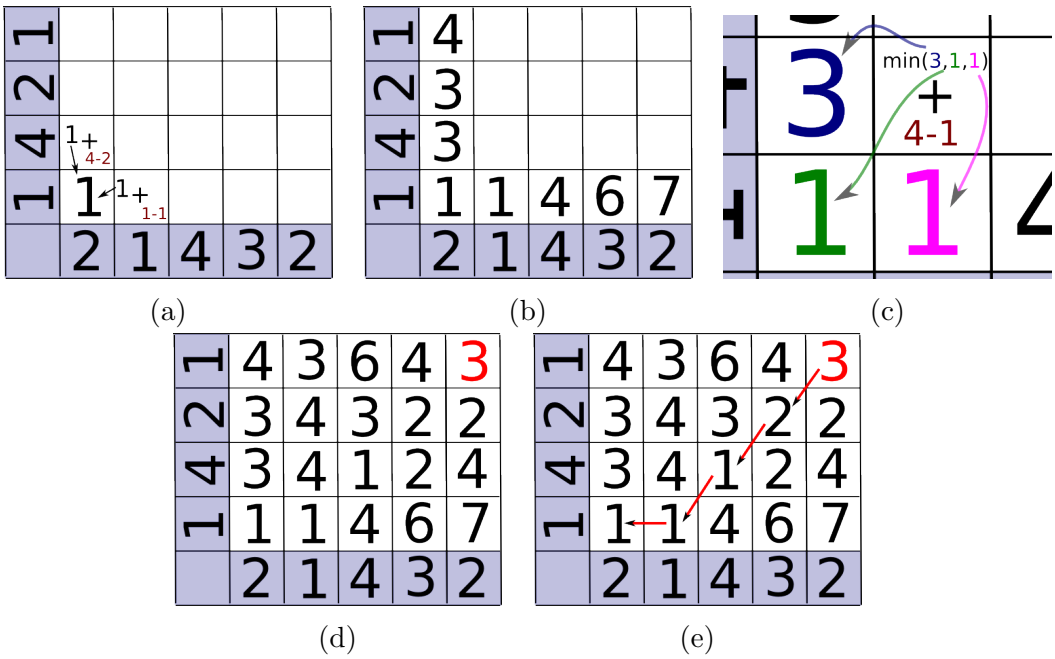
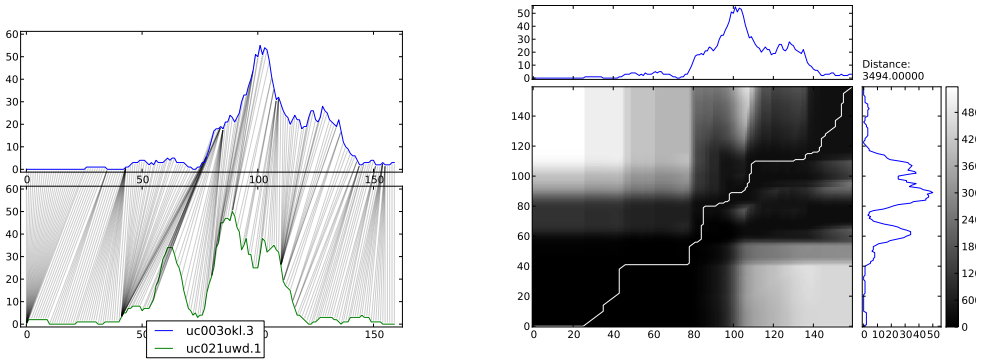


Figure 4.2: Illustration of the DTW algorithm. Two sequences, $(2, 1, 4, 3, 2)$ and $(1, 4, 2, 1)$ are being compared here using Euclidean distance. The algorithm first creates an empty table of size $n \times m$, where n and m are the lengths of respective sequences, in this case $n = 5$, $m = 4$. In this table, each column and row correspond to some value on one or the other sequences, pictured in blue. The distance between the first elements of the sequence is put into the column $(0, 0)$, as pictured in (a). The first row and column are then filled in sequentially, by adding the distance between respective points on each of the sequences, shown in red, to the previously computed value, originating from a column indicated by an arrow, as illustrated in (a). After the first row and column were computed (b), the remainder of the table is filled in according to the recursive equation 4.2 as pictured in (c). Once the table is completed (d) the resulting distance is in the top-right corner (red). The warping path can be traced back from the top right corner, by applying the recursive equation backwards (e).



(a) Alignments (b) Cost matrix and path

Figure 4.3: Result of DTW distance calculation on the H3K4Me3 Activation profiles around transcription start sites for two genes, `uc003okl.3` and `uc021uwd.1`.

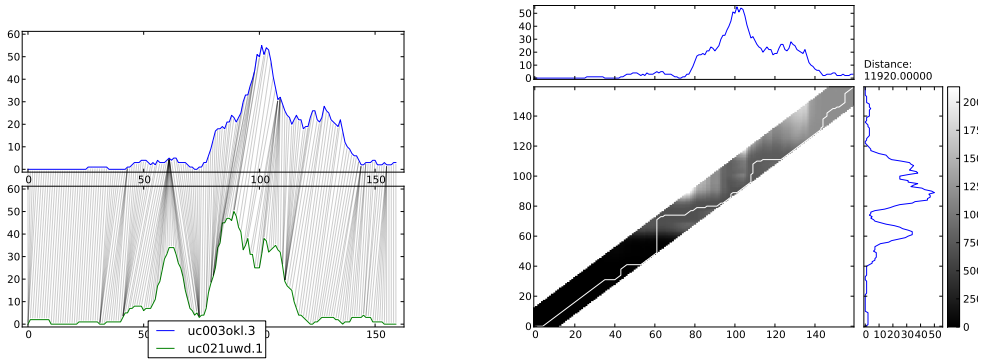
would get the Euclidean distance between these two sequences. We can see that these sequences will likely not be labelled as similar, because the distance between mapped points is always fairly large.

The mapping seen in Figure 4.1b, on the other hand is able to correct for the shift on the x axis. The highest points are now mapped to each other, so are the other points that are in patterns that coexist in both of the sequences. If we were to calculate the differences between the mapped points again, we would see that the resulting distance is much smaller than the Euclidean one and the two sequences will likely be considered close, but not identical, as there still are features in the sequence A that sequence B lacks. For example, sequence A has a valley in the beginning, something what the sequence B does not, what results as a costly penalty.

4.1.1 Constrained DTW

In some cases, warping the time axis arbitrarily might not necessarily be a correct thing to do. For instance, have a look at Figure 4.3. Here DTW is used to compare the H3K4Me3 activation profiles around transcription start sites (TSS) of two genes, `uc003okl.3` and `uc021uwd.1`. We can immediately see that the activation profile in the top panel appears to be unimodal, whereas the activation profile in the bottom panel is clearly bimodal. The minimal warping path maps a relatively inactive region from around bin 40 to around bin 80 in the series for `uc003okl.3` to a single point on the series for `uc021uwd.1`. However, this unexpressed region might have biological significance, e.g. indicate removal of a nucleosome, and we might not want to allow it to be compressed to a single point.

We can manually impose restrictions on warping paths that are allowed by intro-



(a) Alignments (b) Cost matrix and path

Figure 4.4: Result of DTW distance calculation on the H3K4Me3 Activation profiles around transcription start sites for two genes, `uc003okl.3` and `uc021uwd.1`. The DTW distance was constrained with Sakoe Chiba constraint with $k = 12$.

ducing global constraints. Sakoe-Chiba Band is one of the most commonly used global constraints on DTW [RK04].

4.1.1.1 Sakoe-Chiba Band

Sakoe & Chiba constraint, as first described in [SC78] allows only the warping paths that satisfy $|i-j| \leq k$ where k is some integer specifying the tightness of the constraint. The constraint is commonly referred to as "Sakoe & Chiba band" as the cost matrix has a shape of a band around the main diagonal (see Figure 4.4b). Such constraint reduces the complexity of DTW to $O(k \times max(n, m))$ where n and m are the lengths of the two sequences [Mül07].

Intuitively, the constraint requires that each point on sequence A is mapped to a point no more than k points away on the sequence B. That is, it assumes that there won't be any warping of the time axis greater than by k points. Note that setting the k parameter to zero would give us Euclidean distance between the two sequences.

Figure 4.4 compares the same regions in the Figure 4.3 using the DTW constrained with Sakoe & Chiba band of length $k = 12$. One can see that the peaks on the right-hand side of the two sequences are still aligned correctly, but the relatively inactive region in the first sequence is now forced to be mapped to the left peak in the second sequence, resulting in a huge cost penalty and making the sequences quite distant from each other. This alignment results in a total distance of 11920 using squared euclidean distance as the local distance measure. Compare that to the distance of 3494 obtained by unconstrained warping.

As far as the author is aware, there is no established way of picking the value

of parameter k. Historically, k was set to 10% of the length of the sequences. For nearest-neighbour classification tasks, [RK04] has shown that even a window sufficiently smaller than 10% gives the highest accuracy for some datasets, though the actual number strongly depends on the actual dataset. Due to this, the constraint k is not imposed in the Dynamic Genome Warping package and is left as a free parameter the user could specify.

4.1.1.2 Slanted Band Constraint

Sakoe & Chiba band makes DTW incomputable for sequences whose lengths differ by more than k items, as the last points of those sequences, point n and point m, will not satisfy the constraint $|i - j| \leq k$. The reason for this failure is the fact that $i = j$ is not the real diagonal of the cost matrix, if the lengths of the two sequences, n and m, are not equal.

Toni Giorgino was well aware of this problem when implementing the DTW module for R[1] where he included a modified version of Sakoe & Chiba constraint, he calls a slanted band constraint [Gio09].

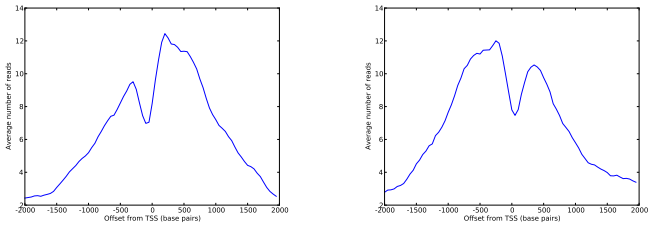
Slanted band constraint solves the problem of different sequence lengths by changing the constraint to $|i - \frac{m}{n} \times j| <= k$, where m and n are the lengths of the two sequences being compared. This essentially changes the slope of the diagonal line to $\frac{m}{n}$ (cf. to slope being 1 in [SC78]).

Unfortunately, the meaning of the parameter k is inconsistent in Giorgino's definition of the constraint. If the second sequence is longer than the first one, thus $m > n$, the constraint can be thought to scale the second sequence down to the length of the shorter sequence and then apply the Sakoe & Chiba constraint of size k, meaning k is in the units of the shorter sequence.

However, if the second sequence is shorter than the first one, it will be essentially rescaled to match the length of the longer sequence and then the parameter k will be measured in units of the longer sequence. In this particular case, DTW distance might not even be computable as in order for the warping path to remain continuous, for all j, there should be some i within the allowed warping window, that is also either in the window of $j + 1$ or is by at most one unit away from some i' that is within the window of $j+1$. More formally, the following inequality must hold:

$$(j + 1)\frac{m}{n} - k \leq j\frac{m}{n} + k + 1 \tag{4.3}$$

[1]The R Project For Statistical Computing - `http://www.r-project.org/`



(a) Forward strand (b) Reverse strand

Figure 4.5: H3K4Me3 profiles for regions around transcription start sites within ENCODE pilot regions [EBS+07]. Genes known to be transcribed on the forward strand are pictured in (a), whereas genes known to be transcribed on the reverse strand are pictured in (b). Note the symmetry between these marks.

After some algebraic calculations, inequality 4.3 can be simplified as:

$$\frac{m}{n} \leq 2k + 1 \tag{4.4}$$

This inconsistency might not be important in the situations where we have a query sequence that we want to match to a reference sequence as the order of parameters will not change. However, symmetry is crucial for hierarchical clustering, therefore Giorgino's constraint was modified slightly to always assume that the first sequence is longer than the second one as described below.

The slope parameter is again calculated as $s = \frac{m}{n}$, however it is now always less than or equal to one. This slope is now applied to the first sequence, (not the second one as in [Gio09]) essentially scaling it down to the length of the shorter sequence. The new constraint is then formally defined as:

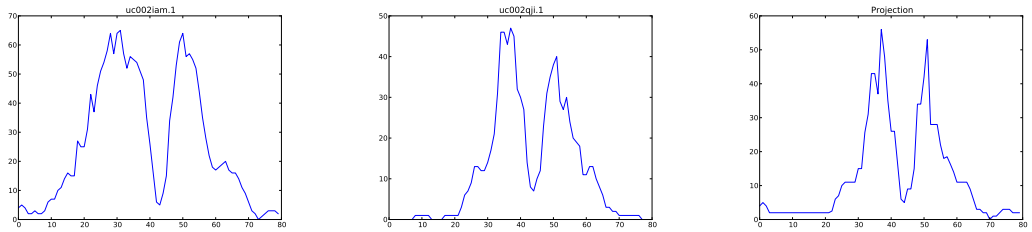
$$|\lceil i \times \frac{m}{n} \rceil - j| \leq k \tag{4.5}$$

Of course, there are no technical limitations preventing the slope to be applied on the second sequence as in the original paper [Gio09]. The reason why this was changed and the slope applied on the first sequence is purely because the implementation is a bit more intuitive this way.

This modified constraint is now always consistent, with the parameter k always measured in units of the shorter sequence.

4.1.1.3 Correction for Antisense Regions

Figure 4.5 shows the average profiles of regions around annotated transcription start sites in the ENCODE pilot regions of the genome [EBS+07]. The average



(a) Sequence (b) Basis (c) Projection

Figure 4.6: Result of projecting sequence, pictured in 4.6a onto the basis, pictured in 4.6b with unconstrained DTW used to generate the warping path.

patterns are evidently symmetrical along the horizontal axis. Even though this directionality is generally known for patterns around well-annotated regions, such as the transcription start sites, the directionality for other regions of interest, such as peaks found by a peak caller, is rarely known.

In order to be able to account for this opposing directionality, every DTW comparison that is made when clustering, is actually made twice: once with the two sequences unchanged, and once with one of them reversed. The smaller distance between the two is then returned as the true distance between the patterns.

Of course, DGW also supports an option to turn this behaviour off, in case we would like to force the different strands into different clusters, or an option to use the annotated strandedness, if it is provided.

4.1.2 DTW Projection

Just like one can project a vector in euclidean space onto a new basis, one can project some time series onto the time axis of another time series under the DTW distance.

Given the original sequence **a**, the basis sequence **b**, and the warping path between the two sequences, p, we can project **a** onto **b** by computing the average value of all points mapped to every point of sequence **b**.

More formally, each point x_k of the projected sequence is computed by the following equation:

$$x_k = \frac{\sum_{(i,j)\in p} a_i \times \delta_{j=k}}{\sum_{(i,j)\in p} \delta_{j=k}} \tag{4.6}$$

Where $k = 1..|\mathbf{b}|$ and $\delta_{j=k}$ is the delta function that is 1 when $j = k$, and 0 otherwise.

Figure 4.6 illustrates the result of projecting one histone mark around the transcription start site onto another.

4.2 Clustering

Previous chapters have defined the mechanics of DTW distance. The important question that is yet to be raised is how the data can be clustered under it.

It has been shown, that under the DTW distance, computation of a sequence that minimises the distance to all other sequences in the cluster, so called centroid sequence, is a NP-Complete problem as the search space grows exponentially with number of sequences being averaged [HNF08; PG12].

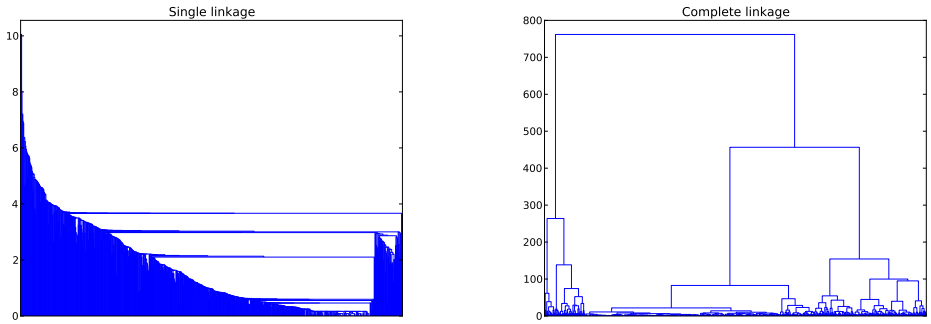
Even though a variety of methods to approximate this calculation have been proposed through the years, [GMTS96; NR09; PKG11; PG12], one needs to be careful before using any of them for clustering, because incorrect averaging methods might create incorrect clusterings, e.g. allow the averaged center drift outside the sequences being averaged [NR07]. Due to this, implementation of centroid-based clustering methods (e.g. k-means) under Dynamic Time Warping distance is usually tricky.

Hierarchical clustering methods, on the other hand, do not require a notion of centroid. One of the hierarchical clustering methods, agglomerative clustering, works by first computing the distances between all pairs of points in the dataset, then joining the two closest points into a cluster, forming a new cluster this way. Then the second two closest points are joined forming yet another cluster, and so on, eventually joining all of the points.

The most common ways to compare two clusters that do not make any assumptions about the underlying distance metric are:

- *Single linkage:* The distance between two clusters is equal to the smallest distance between the any two elements in the two clusters.
- *Complete linkage:* The distance between two clusters is equal to the largest distance between any two elements in the two clusters.
- *Average linkage:* The distance between two clusters is equal to the average of the distances between the elements in the clusters.

Agglomerative clusterings that use single linkage criterion are vulnerable to the phenomenon known as *chaining*. This phenomenon occurs, when instead of forming distinct clusters early, agglomerative clustering algorithm forms a single cluster and incrementally joins more and more elements to it, one at the time. This



(a) Single linkage (b) Complete linkage

Figure 4.7: Comparison of dendrograms obtained by clustering using single and complete linkages. The datasets being clustered are the same, generated from K562 H3K4Me3 dataset, and unconstrained DTW distance metric was used to cluster both of them. The only thing that is different is the linkage method. One can see chaining of the items in the single linkage clustering dendrogram (a), where each element is subsequently added to the same cluster. Compare this to (b) where lots of smaller clusters are formed and then joined together.

is illustrated in Figure 4.7a. Here it seems that majority of data is in one cluster and each data point is joined to the cluster one by one, creating a smooth downwards facing slope seen in the figure.

This is by no means a meaningful clustering, as if we were to cut the dendrogram at pretty much any threshold we would get the majority of the data in one of the clusters, with the remaining data points having a separate cluster for themselves.

We can see that chaining behaviour is not visible when complete linkage is used. Here, the clusters are formed early, and then joined later on, as one would expect (Figure 4.7b). In fact, Hirano et al. [HT05] have shown empirically that out of the three linkage methods discussed above, only complete linkage was able to produce meaningful clusters when clustering time series data under DTW. Due to this reason, DGW computes the distances between two clusters using the complete linkage criterion.

4.2.1 Forming Flat Clusters

It is also worth noting, that there have been attempts to relax the requirement of having a notion of centroid in centroid clustering methods by using a medoid – a representative object from within the cluster. For instance, k-medoids algorithm, implemented e.g. in [PJ09], could in theory be used to cluster DTW distances.

The disadvantage of this method against hierarchical clustering is the fact that the number of clusters, the k in k-medoids, needs to be picked before the clustering

was done. Several iterations of the algorithm could be run, and the results with different ks could be compared, of course, yet it is both not efficient to do so, and it is not clear what metric should be used to measure the meaningfulness of the clustering in this case.

Hierarchical clustering methods, on the other hand, do not need to have the number of clusters decided beforehand, as the number of flat clusters can be chosen *after* the clustering was done. This allows inspection of the clustering results to guide this process.

This assignment to clusters is done by cutting the dendrogram at some threshold which can either be selected automatically, if there is clear way to do this, e.g. if the distances in the dendrogram correspond to p-values, like in [WLJ12], or by simple visual inspection of the resulting dendrogram. The latter approach is implemented in this project.

4.2.2 Cluster Prototypes

After the flat clusters are formed, the clusters need to be visualised to the user somehow. Just displaying the original data (i.e. in the form of a heatmap) is insufficient, as such representation of the data does not does not explain why the data was clustered together in the first place.

The end user of the software, would want to see the data represented after warping, i.e. average profile of the warped data, as to be able to get the structure and the shape of the resulting cluster marks. The problem of extracting this average warped profile lies in the fact that only two sequences are compared with each other one at the time, and both of them can be arbitrarily warped. What this means, unfortunately, that the same sequence might be warped in as many ways as there are sequences to compare it against. In order to be able to give an average warping of sequences in a cluster, one needs to agree on the base sequence upon which to warp the remaining sequences on. In other words, one needs to find a prototype for the cluster in order to be able to visualise the warped items. Once this prototype sequence is chosen, DTW projection (subsection 4.1.2) can be used to warp the original data onto the basis defined by the prototype.

Remember that inability to compute perfect centroids was one of the reasons why hierarchical clustering was chosen in favour of centroid-clustering in the first place. Then we were worried about the bias in the data clustering producing clusters that are not meaningful. However, at this case the data is already clustered and flat clusters are already formed. Some prototyping error is tolerable here as it won't be able to accumulate as easily as it would during clustering. The only requirement that is posed on the prototype is that it expresses a similar shape than the marks being clustered visually.

4.2.2.1 Prioritised Shape Averaging

A method called Prioritised Shape Averaging (PSA), proposed in [NR09], was chosen to generate the cluster prototypes. The reason why PSA was chosen, in favour of other methods, is the fact that it can exploit the hierarchical structure, we've already computed in prototype generation stage.

The basic idea under the PSA can be thought of in terms of a bottom-up algorithm. Initially, make all of the leaf nodes to be the prototypes of themselves, with the weight of one assigned to each of them. Moving up the clustering tree, for each node, average the two prototypes of the left and right child of the node (remember that hierarchical clustering trees are always binary), with some averaging procedure that takes the weights of the two nodes into account. Make the weight of this newly formed node equal to the number of elements in that node.

The proposed method to be used in PSA is Scaled Dynamic Time Warping (SDTW) averaging algorithm that is also defined in the paper. This algorithm works similarly to the standard path averaging algorithm, which is defined as follows: For each pair of points on the DTW warping path, (i, j), average the corresponding points in the two sequences a_i, and b_j, in the following way:

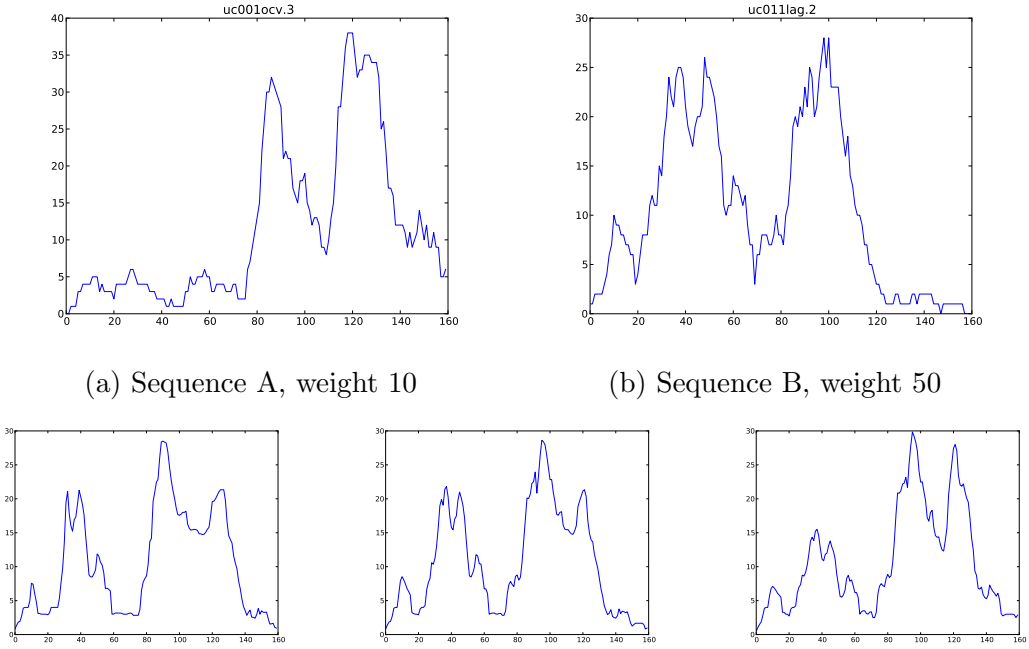
$$x = \frac{\lambda_a \times a_i + \lambda_b \times b_j}{\lambda_a + \lambda_b} \tag{4.7}$$

Essentially, it is doing weighted averaging of these paths.

SDTW extends this definition, by making the algorithm "stretch" the resulting point x, λ_a times, if the point in sequence **a** was unchanged compared to previous pair of points in the path, i.e. the warping path moved straight upwards; λ_b times if the warping path moved horizontally, $\frac{\lambda_a+\lambda_b}{2}$ times if the warping path moved diagonally.

The major problem with this averaging method is that the paths created will get longer and longer as the algorithm progresses up the tree. The possible solution for this problem, as proposed in [NR09] is to uniformly rescale the resulting average path to be of the length of one of the two sequences. DGW uses this solution and always rescales the computed prototype to match the length of the longer sequence.

Figure 4.8 compares the STDW averaging method with standard averaging method as in Equation 4.7 as well as to standard unweighted averaging method (where $\lambda_a = \lambda_b = 1$). One can see that the average sequences produced by both SDTW and standard averaging methods favour the sequence with more weight strongly and therefore resulting average does resemble the second sequence more than the first one. Compare this to unweighted average of the two sequences, where the new shape resemble both features equally.



(a) Sequence A, weight 10

(b) Sequence B, weight 50

(c) SDTW

(d) Standard

(e) Standard-Unweighted

Figure 4.8: Visualisation of the three prototyping methods discussed in subsubsection 5.3.1.2. Two sequences in 4.8a and 4.8b were averaged with each other using DTW with Sakoe Chiba band constraint set to 12. The first sequence had a weight of 10, whereas the second sequence had a weight of 50. The results using various averaging methods are shown in 4.8c, 4.8d and 4.8e.

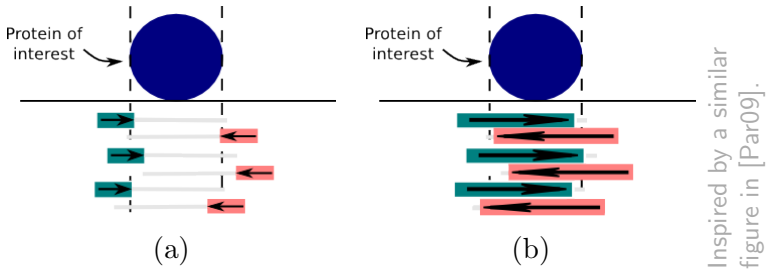


Figure 4.9: Illustration of the reasoning for extending the ChIP-Seq aligned reads. The DNA attached to the protein is pictured in grey, while the sequenced reads are pictured as coloured rectangles. In general, these short sequenced reads will not capture the true structure of the DNA that is bound to protein, as seen in (a). This can be mitigated by extending these reads to some length after the alignment, as shown in (b).

From these figures, it is hard to spot the difference between prototypes generated by SDTW and Standard methods is minimal. Surprisingly, I did not find any but minuscule improvements SDTW makes to the end result of the cluster prototype along the whole hierarchical clustering tree. These improvements do not seem to be worth the extra overhead incurred by the clever stretching of the warping path, and therefore the DGW uses standard path averaging by default, but the user is able to override this.

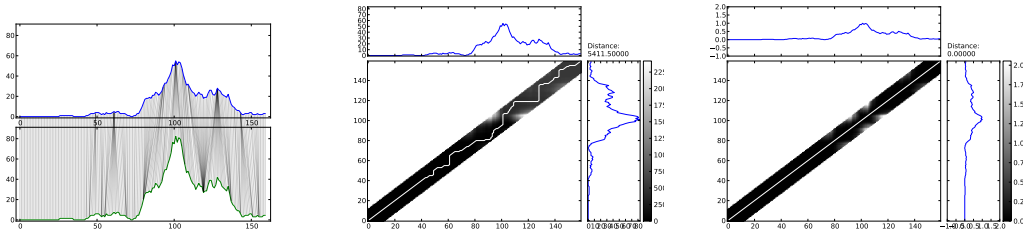
4.3 Reading and Preprocessing Data

As previously mentioned in section 2.4, the end result of a ChIP-Seq experiment is a collection of short reads that are aligned to the reference genome, as pictured in Figure 2.6. In order to compute the activation profiles, each of the regions of interest are split into a series of bins of equal width. This bin width is simply referred to as *read resolution*. The number of reads falling within these bins is then counted to produce the activation mark.

Of course, as the reader might remember, usually only a short fragment of the DNA attached to the target protein is sequenced. In some cases, these short reads will not be able to capture the true distribution of DNA attached to the target proteins, as pictured in Figure 4.9a. In order to mitigate this problem, DGW extends all reads to be 200 base pairs wide as pictured in Figure 4.9b.

4.3.1 Highest Pileup Filter

Intuitively, the less reads collect to the same bin, the more likely this could have happened due to one kind of noise or another. Given a long-enough region, even in a setting where reads arrive to bins completely at random, there will be at



(a) (b) (c)

Figure 4.10: DTW sensitivity to changes of scale. The number of reads falling to the region around transcription start site for `uc003okl.3` was increased 1.5 times. Even though this increase does not change the shape of the region (a), just the scale, the DTW distance between the original and modified regions has grown to 5411.5 (b) which is about half of the distance between two distinct marks seen in Figure 4.4. A non-diagonal warping path between the regions can also be seen. After the pileup height normalisation, the distance is zero again and the path is diagonal (c).

least a few bins within that region that will have at least a few reads piled up. Even though these regions provide little meaning biologically, they still need the same amount of CPU operations to be processed as every other region. With this in mind, DGW employs a filtering strategy that automatically filters regions that do not have a single bin with more than 10 reads piled up inside. Again, as with other parameters this number can be easily changed by the user.

4.3.2 Pileup Height Normalisation

DTW distance is sensitive to changes in scale, if the local distance measure is sensitive to changes of scale. In case of squared euclidean distance, it is very easy to artificially inflate the DTW warped distance between the two regions. For instance, consider the same region used in Figure 4.4. If one was to multiply all the bin counts by 1.5, the underlying shape of the region will not change, however, as pictured in figures 4.10a and 4.10b, the DTW distance will increase dramatically.

In some settings, e.g. differential expression studies, this might be a perfectly valid behaviour, as we might indeed want to treat these two regions as different. In other settings, e.g. when we are trying to compare the marks at the $5'$ splicing sites, it might make more sense to compare only the shape of the mark, and not the shape and expression value. This comparison of shape can be achieved by normalising the values of the bins with the most reads falling into it to be equal to one for all regions of interest.

That is, if we thought the mark as of a vector $\mathbf{x}$, then the normalisation equation

could be written as:

$$\mathbf{x}_{norm} = \frac{\mathbf{x}}{max(\mathbf{x})} \tag{4.8}$$

After the normalisation the bin with the highest number of reads in it, will always have a value equal to 1, so the bins will be compared at the same scale. One can see the normalisation turning the distance between the original and modified regions to zero again in Figure 4.10c.

5. Implementation Details

The methods described in the previous chapter were compiled together into a software package, named Dynamic Genome Warping, or DGW for short.

This chapter describes the implementation of this software, including its structure, interface and solutions to key computational challenges faced.

5.1 Dependancies

The core of DGW package was written in Python 2.7[1]. The software exploits the maturing scientific community around Python and builds upon a large variety of packages produced by it.

One of the core dependancies of DGW is the `pandas` package, available from `http://pandas.pydata.org/`. This package provides efficient, R-like data structures that are extremely easy to use. DGW uses these data structures throughout the code as to be able to both store and manipulate the histone marks efficiently.

Packages in PyLab, namely `numpy`, `scipy` and `matplotlib` also play a crucial part in this software. Together they bring functionality of MATLAB[2] to Python. The first package, `numpy` provides efficient computational routines on arrays and matrices; the second package, `scipy` has a majority of scientific procedures available in MATLAB implemented, including the ones required for drawing and analysing hierarchical clustering dendrograms; whereas the third one, `matplotlib` [Hun07], provides a plotting API that resembles the one in MATLAB.

Furthermore, Daniel Müllner's `fastcluster` package [Mül11], is used for efficient computations of hierarchical clustering linkage matrices. Whereas `pysam` (`https://code.google.com/p/pysam/`) package is used for it's Python interface to the *SAMtools* ([LHW+09]) toolkit.

Finally, implementation of the DTW algorithm is based on `mlpy` package [AVM+12]. However, the author of this project had to extended the functionality of this package: variations of DTW algorithm were implemented, as described in section 4.1; the canonical DTW algorithm in the package as to support alignment between sequences of vectors, rather than sequences of scalars. These improvements are in the process of being contributed back to MLPY.

[1] See Python's homepage for more information `http://www.python.org/`

[2] See `http://www.mathworks.co.uk/products/matlab/` for more information about MATLAB

5.2 File formats

Since the ENCODE project ([RDL+11]) is probably the largest repository of genomic data to date, the data formats used in this repository can be considered to be the industry standard. An exhaustive list of the data formats used in this project is provided at the FAQ section on their website, `https://genome.ucsc.edu/FAQ/FAQformat.html`. This section describes the two most important ones from this list, BAM and BED.

BAM file format is a successor to the SAM file format the *SAMtools* toolkit is named after. The package stores the sequence alignments from a NGS experiment in a compressed binary format. Due to the amount of sequencing data generated by these experiments, BAM files are often a few gigabytes in size. To keep data access efficient, BAM files are often distributed along with an index file, that provides pointers to specific genetic locations in the compiled BAM file.

This format is not designed to be processed by hand, and the use *SAMtools* package is suggested instead. As already described earlier, DGW uses a python wrapper around this library, `pysam` to be able to extract the reads from these files.

Another file format, BED, is perfect for describing a set of genomic regions of interest, and is the default output format of the MACS peak caller due to this reason. It's structure is extremely simple: a set of tab separated fields for chromosome, start of the region, end of the region, name of the region and a few other optional fields. Due to this it is easy to parse by hand and a majority of methods that are designed to process CSV can be used to do this. DGW uses one of such CSV processing methods, that is implemented in `pandas` package to quickly load the contents of this file into `pandas` data structure.

These two file formats are used for input to the software – the ChIP-Seq datasets are expected to be in BAM format, whereas the regions of interest to cluster are expected to be in BED format.

DGW module also allows tracking of interesting locations both before and after the warping, for instance, tracking the locations of transcription start sites overlapping with clustered regions. In some cases there might be more than one such point to be tracked within a given region, e.g. more than a single transcription start site overlapping some peak called by MACS. Since BED format does not support describing disjoint regions by default, a simple file format for defining such regions was created for the sole purpose of using it in DGW.

Each line in this file format starts with the name of the region of interest that was clustered (e.g. name of the MACS peak), followed by a colon, followed by a list of comma-separated locations of points of interest within that region, for

instance:

```
MACS_peak_84:1655238,1655387,1655705,1655696,1656202
MACS_peak_86:1677162
MACS_peak_87:1689803
MACS_peak_89:1709727,1710184,1711343
```

Regions that contain no such points of interest, are simply ignored.

Even though the format is simple, it is still an *ad-hoc* file format users will not be used to working with. Due to this an utility script to automatically generate points-of-interest file overlapping regions in two BED files was developed. With the help of this script, a majority of the users will not need to know how the points-of-interest file is structured as the script will do the work for them.

5.3 Structure of the software

Dynamic Time Warping is an expensive distance metric to use, as the runtime of the algorithm scales quadratically with the length of the sequences (compare this with e.g. Euclidean distance which is $O(n)$ instead). Combining that with tens of thousands regions of interest that need to be compared to each other in order to hierarchically cluster them, one will get an algorithm that is going to take a while to complete.

CPU-demanding software like this is not uncommon in the field of life sciences, due to the amount of data modern experiments can produce, and therefore a majority of these labs have access to a supercomputer.

With this in mind, DGW is designed as a combination of two standalone modules: a computationally heavy module designed to be run on a supercomputer, and an exploratory module, that can analyse the results obtained from the worker module on a less-powerful computer.

5.3.1 The Worker Module

Figure 5.1 outlines the workflow of the worker module. The module first reads the regions of interest supplied by the user in a BED file. These regions are then divided into a set of bins of specified width (50 base pairs wide by default), and the histogram of number reads falling within those bins is computed, much like in Figure 2.6, to get a collection of profiles to be analysed.

These profiles are then by keeping only the ones that have at least one bin with more than 10 reads falling into it. This filtering removes a substantial part of the

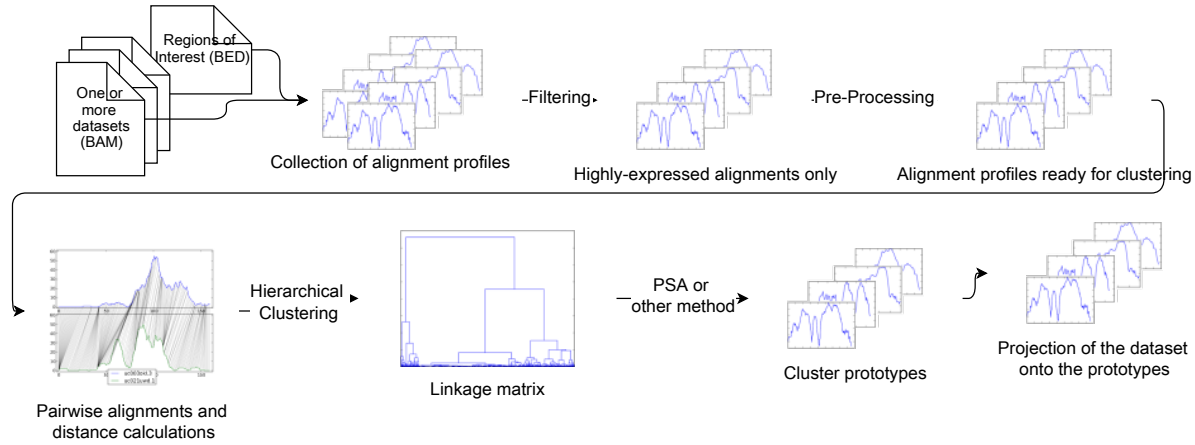


Figure 5.1: Workflow of *DGW-Worker* submodule. The datasets are read at the regions of interest specified by the user. Then, the resulting collection of alignment profiles is filtered leaving only highly-expressed alignments. These alignments are then pre-processed by converting them to log-scale and, optionally, normalising their values. The module then performs pairwise alignments between the items in order to calculate the pairwise distance matrix, which is then used to compute the linkage matrix for hierarchical clustering. After that, prototypes for each of the possible clusters are calculated from this linkage matrix, and the data is projected onto these prototypes.

dataset and thus reduces the runtime significantly. The underlying assumption of this is that relatively unexpressed regions are not as interesting from the scientific point of view, and more prone to variations in patterns due to random noise. The resulting data is then converted to log scale and, optionally, normalised to be on the same scale. Please refer to section 4.3 for an in-depth discussion of these methods.

5.3.1.1 Clustering

During the clustering step, pairwise distances between each of the profiles are calculated. Since this is the most time-demanding step of the algorithm, an extra effort has been put to make the clustering as efficient as it gets.

To achieve this, the DTW clustering routines are implemented in native C. These native C modules are then called from Python code using the C-Extensions for Python suite, Cython (`http://www.cython.org/`). Native C code reduces the time required to compute the distance between two alignments from milliseconds to microseconds.

Microseconds might not seem much, but the number of computations that are needed to compute the pairwise distance matrix is extraordinary. If the DTW distance computation on average takes around 161 microseconds to complete for two sequences of length 80, computing the pairwise distances for 30000 regions

would take around 20 hours to complete as there would be $\binom{30000}{2} \approx 4.5 \times 10^8$ comparisons to make. Fortunately, computation of pairwise distances is easy to parallelise as each computation is completely independent from the others.

Parallelisation in software engineering is usually done by splitting the execution into multiple threads. Python supports multithreading out of the box, but with a catch: the Global Interpreter Lock (GIL), prohibits more than one thread interacting with the Python interpreter at the same time. Unfortunately, this means that the multithreading module is only useful for off-loading long-running non-Python procedures, such as I/O waits, to a separate thread. Due to this, instead of creating new threads, parallel programs in Python create new processes each having separate Python interpreters attached to them and therefore separate GILs.

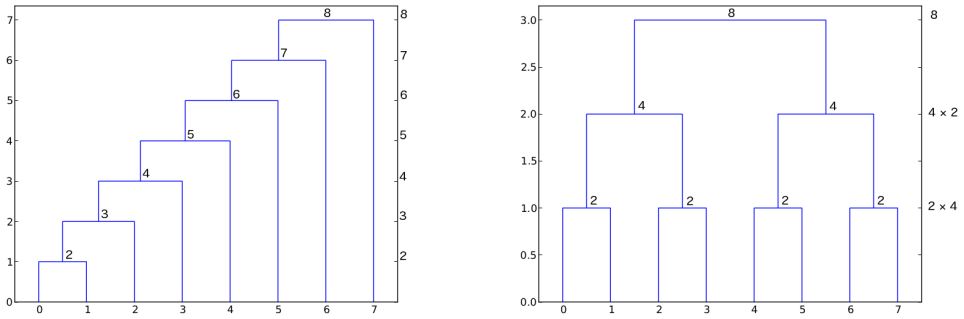
Multiprocessed architecture, however, makes sharing of information between threads challenging. By default, Python copies the memory contents from the parent process to the child process. This behaviour is sufficient in nine cases out of ten, but, not this one. ChIP-Seq alignment profiles, once read, could take up to a few gigabytes of memory. While it is not uncommon today to see computers with 8 or more gigabytes of RAM that can easily fit datasets like this, few computers could afford copying this dataset in the memory four or more times. With this in mind, DGW implementation is careful to share critical bits of memory between the processes so the same amount of memory is used when splitting the computation between the processes.

This multiprocessing reduces the computational time by a factor equal to the number of processes used, usually upper-bounded by the number of CPUs in the computer. For instance, it would take eight times less to cluster the dataset on a machine with eight CPU cores, than it would take to cluster the same dataset on a computer with a single CPU only.

The clustering step is completed by computing the hierarchical clustering linkage matrix. In order to do this, an efficient algorithm from the `fastcluster` package is used.

5.3.1.2 Prototype Generation and Data Projection

An important property of hierarchical clustering is that there will always be $n-1$ nodes in the resulting dendrogram, where n is the number of elements being clustered. This means that there are at most $n-1$ distinct clusters possible from any particular clustering of data, therefore there will at most be $n-1$ cluster prototypes possible. The worker exploits this and computes all $n-1$ prototypes using the bottom-up algorithm described in section 5.3.1.2.



(a) Worst case (b) Best case

Figure 5.2: Illustration of best-case and worst-case linkage matrices that may result from clustering of eight items. The number above the horizontal nodes corresponds to the number of items in the cluster. The number on the right side shows the number of items per level.

In order to compute the warped dataset, each element in each of the possible clusters has to be projected onto the cluster prototype. This sounds like a tedious task, and indeed, in the worst case, where the nodes would form a structure pictured in Figure 5.2a, this computation would take $\frac{1}{2}n(n+1)-1$ operations to complete, what is a bit more than $\binom{n}{2}$ operations needed to compute the pairwise distance matrix.

On the other hand, in the best case where the $n-1$ nodes would form a perfect binary tree, as pictured in Figure 5.2b, only $O(n \log n)$ DTW projection computations would be needed. Luckily, as the reader might remember from section 4.2, the complete linkage criterion makes dendrograms like the one in Figure 5.2a highly unlikely, and instead makes the dendrograms look closer to the one in Figure 5.2b on average.

The worker module exploits this and precomputes the DTW warping paths needed to project the dataset so the worker module does not have to. Computing these paths in the worker module has another advantage of the worker being able to parallelise these tasks within the multiple CPUs of the supercomputer. Only the paths, and not the actual projections are computed, as the former takes up less memory to store. Also, having the precomputed paths allows the explorer module to be able to efficiently track locations of points-of-interest in the warped data, as described in the next section.

5.3.2 The Explorer module

The explorer module is a complimentary module to the DGW-worker, that is responsible for result visualisation and data exploration.

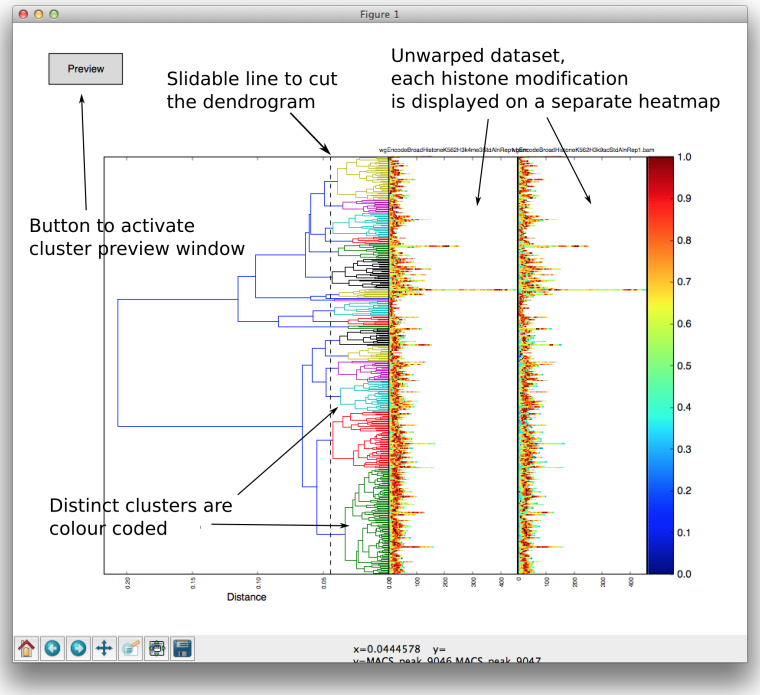


Figure 5.3: Annotated screenshot of the main window of DGW-Explorer application

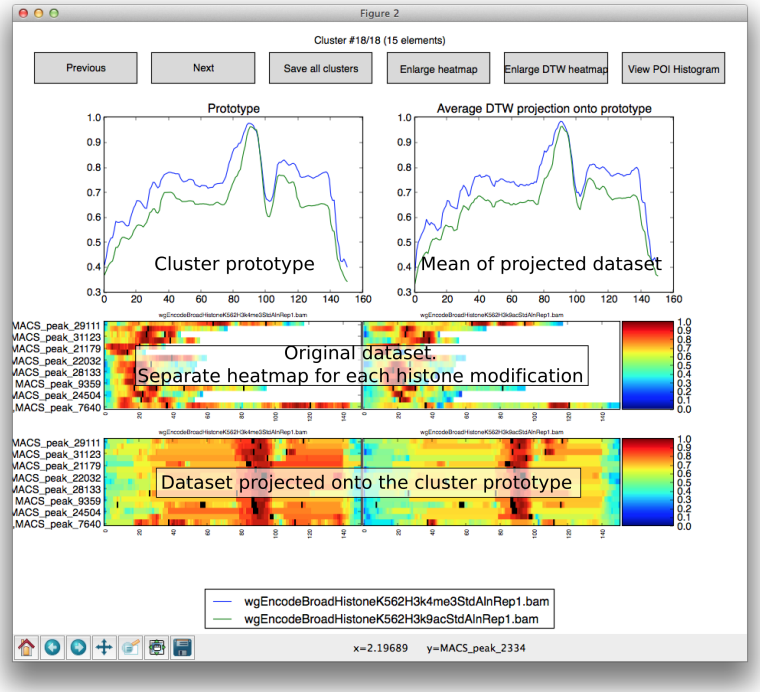


Figure 5.4: Annotated screenshot of the Cluster Previewer

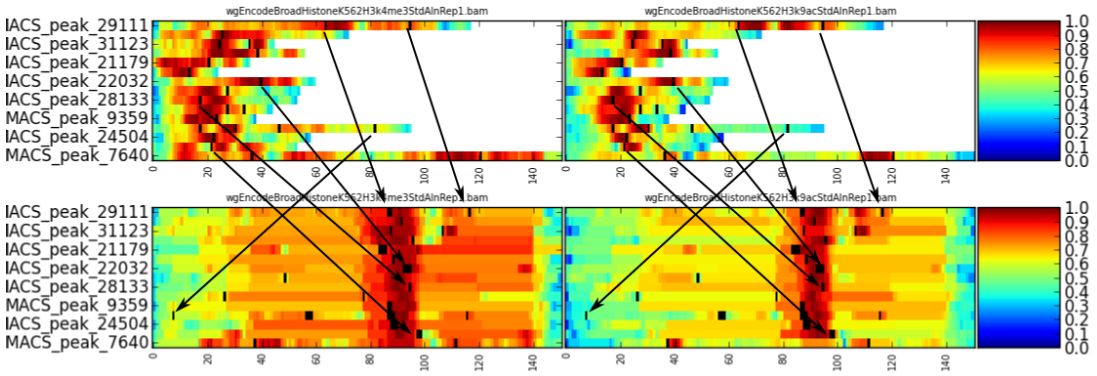


Figure 5.5: The heatmap part of explorer window in a close-up view. In this experiment, annotated first splicing site locations were tracked and highlighted on the heatmap as black dots. The bins containing these first splicing sites after warping are also highlighted, as seen in the figure. Black arrows were added with the help of image editing software to visualise the connection between original and warped heatmaps. Note that one of the arrows is facing an opposite direction from the others, because the region was flipped during the warping.

The main window of *DGW-explorer* is shown in Figure 5.3. Here the dendrogram and zoomable heatmap of the original dataset is shown. The user is able to select the dendrogram cut threshold by simply right-clicking anywhere on the dendrogram. The right-click was chosen instead of left-click in order to not interfere with the default zoom controls available in the bottom-left corner of `matplotlib` windows. After the dendrogram is cut, a vertical line marking the threshold is drawn and the resulting clusters are immediately colour-coded, as seen in the figure. At this stage, pressing the *Preview* button, brings the Cluster Previewer window up.

Cluster Previewer window is pictured in Figure 5.4. This window summarises the clustering on a cluster-by-cluster basis. The resulting cluster prototype is shown in the top-left panel. The original data is shown below it. This dataset is then projected onto the prototype using the DTW projection technique, described in section 4.1.2 and displayed in the bottom-most panel.

The user can navigate between the clusters by the help of *Previous* and *Next* buttons in the top-left corner of the window, or can choose to save the clusters to a set of BED files that can then be processed by other means, e.g. viewed in the Genome Browser ([KSF+02]).

5.3.2.1 Point-of-interest tracking

The heatmap of the data projected onto the prototype is the single most useful piece of information in the cluster explorer window as it allows visual comparison of cluster items on a common ground. In order to make these comparisons

even more effective, *DGW-explorer* allows the user to specify interesting genomic locations, so called points-of-interest (POI), *a priori*. These locations are then marked both in the original (unwarped) and warped data heatmaps as seen in Figure 5.5.

Note that the POI locations are not used during the clustering or prototype generation and are here to aid the testing of hypotheses of what each of the warped patterns could mean. Since the DTW warping paths were already computed by the worker module, the POI points can be changed without rerunning the experiment, making the process easier when multiple hypotheses are available.

The results chapter gives an example on what insights can be generated by this method.

6. Results

Evaluation of results of clustering methods has always been a challenging problem. Since clustering is unsupervised by definition, there hardly is any ground truth on what the correct way of assigning data into clusters is. One clustering might be perfectly sensible in one application, but completely useless in another. Hierarchical clustering has no clear way of deciding where dendrogram should be cut at, as already discussed earlier, making it even harder to evaluate clusterings like this. Finally, the highly exploratory goal of this project makes it even harder to sensibly evaluate the cluster assignments without doing a downstream biological analysis on them.

Despite this, this chapter aims to provide some form of evaluation for this method.

As previously described, Bieberstein and colleagues have shown that H3K4Me3 signal peaks at the location of $5'$ splicing site [BOSN12]. The experiments described in this chapter all inspired by these results.

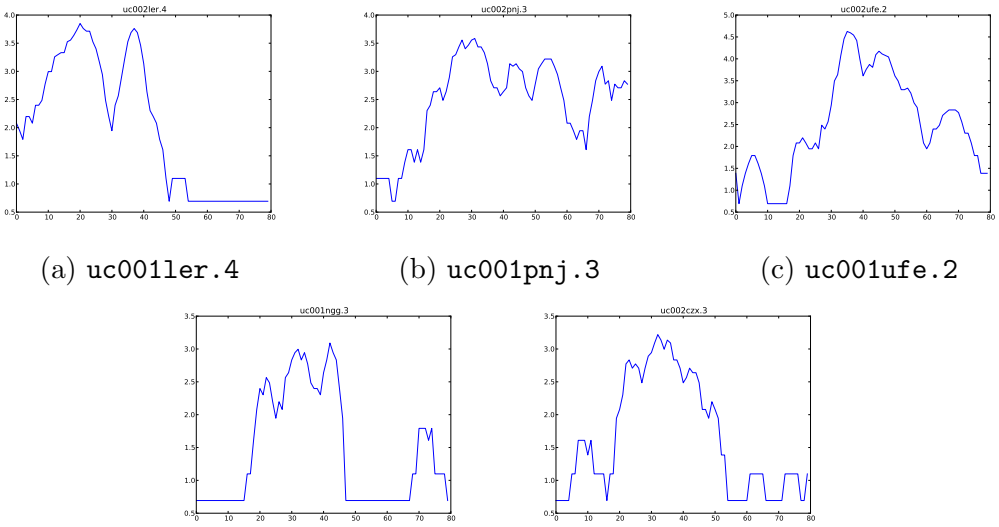
6.1 Clustering an Artificial Dataset

In order to demonstrate what DGW clustering can achieve, the algorithm was tested on an artificially created dataset. In the spirit of the paper that has been mentioned previously, this artificial dataset was generated by stretching the exon regions of a few randomly-selected genes from the UCSC known gene dataset.

More precisely, 5 genes were chosen at random. A particular care has been taken to choose among only those genes, that have their first splicing site no further than 2000 base pairs away from the transcription start site. Alignments that fall within 2000 base pairs window around transcription start sites of these genes were read at 25 bp resolution.

The exons falling within this window were then extended randomly to be up to 60 bins, or in other words, $60 \times 25 = 1500$ base pairs longer. The exons were uniformly extended, keeping the shape of the underlying pattern the same, just stretched. Using this technique, dataset of size 500 was generated from the 5 seed genes depicted earlier.

These newly-generated regions were then randomly mutated, with probability of 0.33, by changing a value of a data point, to some random value picked from the uniform distribution $U(0.5x, 1.5x)$ where x is the previous value of the point. Half of these regions were then randomly reversed to simulate antisense patterns.



(a) `uc001ler.4` (b) `uc001pnj.3` (c) `uc001ufe.2`

(d) `uc001ngg.3` (e) `uc001czx.3`

Figure 6.1: The five seed regions that were used to generate the artificial dataset. Note the similarity between the patterns in 6.1d and 6.1e, i.e. if one of the patterns was reversed

Figure 6.1 shows the five seed patterns that were selected. Notice that two of them, namely `uc001ngg.3` (Figure 6.1d) and `uc001czx.3` (Figure 6.1e) seem to have a relatively similar mark: the former having a large peak followed by a smaller one, whereas the latter having a smaller peak followed by a larger one. Intuitively, we would expect the algorithm to be able to join the items generated from these two samples relatively early.

And indeed, the dendrogram in Figure 6.2 shows the resulting hierarchical clustering of these regions. Only four distinct clusters were created at the selected cut threshold, because the items from two similar prototypes were joined into the largest green cluster earlier than other clusters were formed.

This early merging can be explained by the nature of the mutation function that is being used in this experiment: since the level of mutation depends on the actual value of the point being mutated, seed regions that have a majority of regions expressed poorly, will undergo less mutation than highly-expressed regions, due to this the distance at which they are joined will be lower.

Despite the early merging, the prototypes of these four clusters, pictured in Figure 6.3, seem to more or less resemble the seed motifs in the figure Figure 6.1. The prototype for the green cluster seems to join the seed motifs `uc001ngg.3` and `uc001czx.3` together by averaging the features that occur in both of them.

Looking at the DTW projections of the data onto these cluster prototypes in Figure 6.4, it can be seen that the exons (marked in black) align more or less

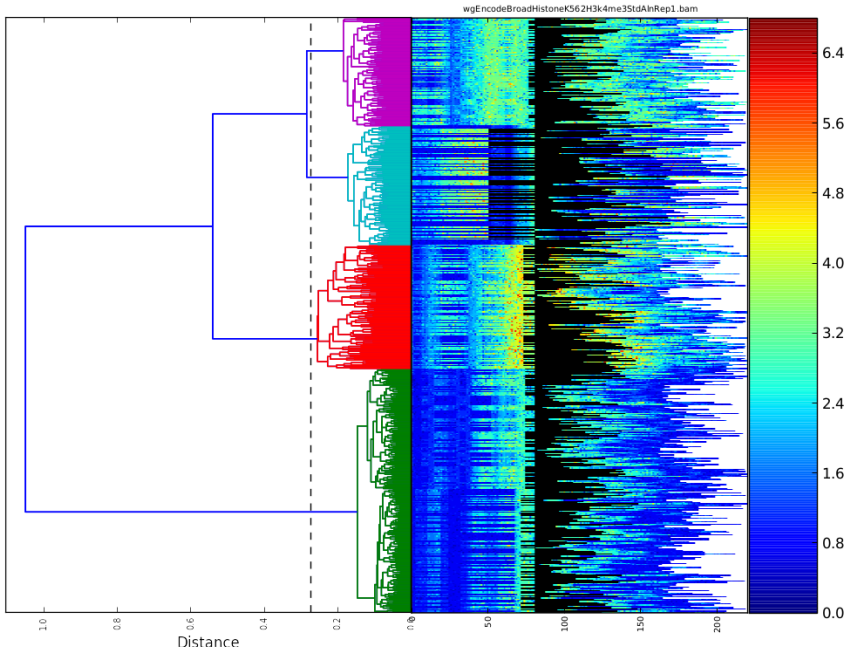


Figure 6.2: Result of clustering the dataset that was generated by stretching exon regions from the five seed samples in Figure 6.1. The extended exons are marked as black points in the heatmap.

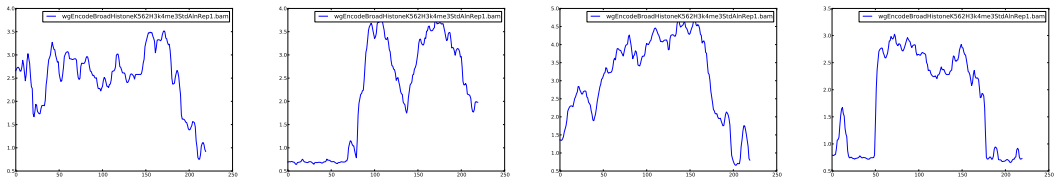
after the warping, with some slight inaccuracies due to the artificial mutations of the dataset.

Since items in these heatmaps are in the same order as they were in the dendrogram, the separation between two subclusters forming the green cluster is very clear in the heatmap as well as seen in Figure 6.4d. Note that is clear that items of one of the seed patterns were reversed to align better with the items from the other seed pattern. This can be seen by looking at the positions of the exons of the two subclusters.

6.2 Clustering the Results of a Peak Caller

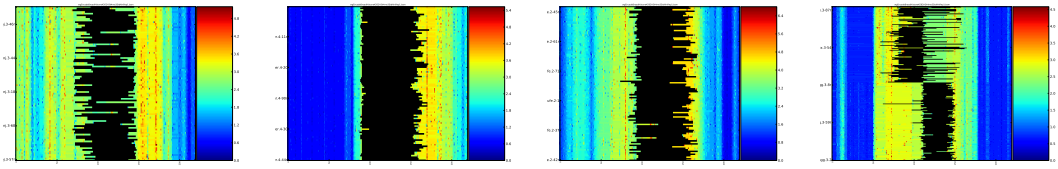
Tests on the artificial dataset have shown that the DGW algorithm is able to do what it says on the label. However, this does not give any insights on the whether the method will prove useful for actual research. In order to test this, an experiment to simulate a scenario that is likely to occur in research was created.

In this experiment, MACS peak caller with default configuration [ZLM+08] was run on the ChIP-Seq results for H3K4Me3 marks in K562 cell type. Data from the control ChIP-Seq experiment was used to allow MACS anticipate between



(a) Purple cluster prototype (b) Cyan cluster prototype (c) Red cluster prototype (d) Green cluster prototype

Figure 6.3: Prototypes of the four clusters resulting from the dendrogram in Figure 6.2. Note how closely they resemble the five seed motifs in Figure 6.1.



(a) Purple cluster (b) Cyan cluster (c) Red cluster (d) Green cluster

Figure 6.4: Data dynamically warped onto the prototypes. Exon locations are marked in black. Note the clear split between two seed motifs that are combined into the green cluster.

significantly and insignificantly enriched regions relative to the control.

The output of this peak caller was then preprocessed by joining together resulting regions that are within 50bp of each other, using the BEDTools package [QH10]. All transcription start sites and first splicing sites that overlap these regions were identified and compiled into a list of points of interest. Regions containing no such points of interest were dropped.

Then the datasets for H3K4Me3 and H3K9Ac histone modifications, the same histone modifications that were observed to peak at first splicing sites by [BOSN12], were read to generate multidimensional histone profiles for these regions.

Histone marks at the regions of interest were then clustered by DGW at the resolution of 50 base pairs per bin. Squared euclidean distance was selected to be the local distance measure and slanted band constraint of width 12, as described in subsubsection 4.1.1.2 was chosen. The maximum histogram heights were all normalised to be equal to one as described in subsection 4.3.2.

Figure 6.5a shows the resulting dendrogram and the chosen cut. Explorer window for one of the clusters is pictured in Figure 6.5b. Note the highly varying lengths of the peaks in the original data. Figure 6.5c shows the enlarged heatmap of the data projected onto the cluster prototype. Here, the heatmap of the left represents the projected H3K4Me3 mark, whereas the heatmap on the right corresponds to H3K9Ac.

Looking closely, it can be seen that the first splicing sites (marked in black) are aligned to the same location after warping. Figure 6.5e quantifies this alignment by plotting the points-of-interest distribution for each of the warped bins. The top panel corresponds to the distribution of $5'$ splicing sites, whereas the bottom one corresponds to the transcription start site distribution. From this figure, and from the heatmap, it is clear that not only first splicing sites are aligned, but also the transcription start site locations are aligned a few bins to the right from the $5'$ splicing site.

This alignment is not visible in the original data, as pictured in Figure 6.5d. In order to generate this histogram, the regions were stretched to match the length of the longest region first, using a procedure similar to the one used to stretch the exons in the previous experiment. The distributions for these stretched regions were then computed in the same way as for the warped data.

6.3 Distinct Patterns Around Transcription Start Sites

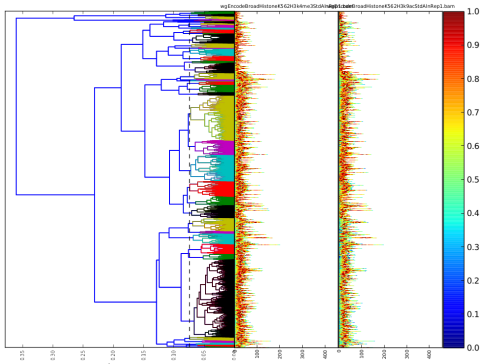
The final experiment is designed to test DGW's ability to identify and visualise the distinct patterns within the regions of interest. In this experiment, well documented regions around transcription start sites were clustered.

More specifically, 2000 base pair windows around the transcription start sites of UCSC known genes dataset (see subsection 2.4.1) were selected. If two regions were overlapping each other, both of them were discarded as to only leave regions that contain a single transcription start site and no other transcription start sites in the neighbourhood.

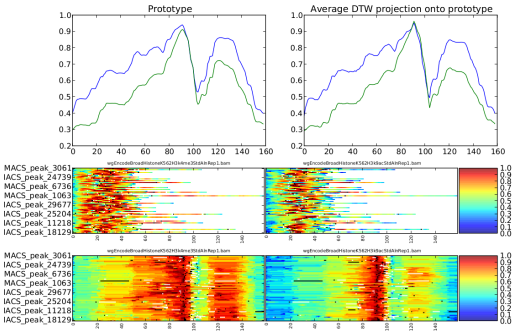
The data at these specific regions was extracted from the ENCODE H3K4Me3 dataset for the K562 leukaemia cell line at the resolution of 25 base pairs per bin. Before processing, the dataset was normalised as described in subsection 4.3.2. This data was then clustered using DTW constrained with slanted band of size 12.

The resulting dendrogram can be seen in Figure 6.6. Looking at the cluster prototypes, pictured in Figure 6.7, the well-known bimodal patterns around TSS can be seen in figures 6.7c, 6.7d and 6.7e. 6.7j.

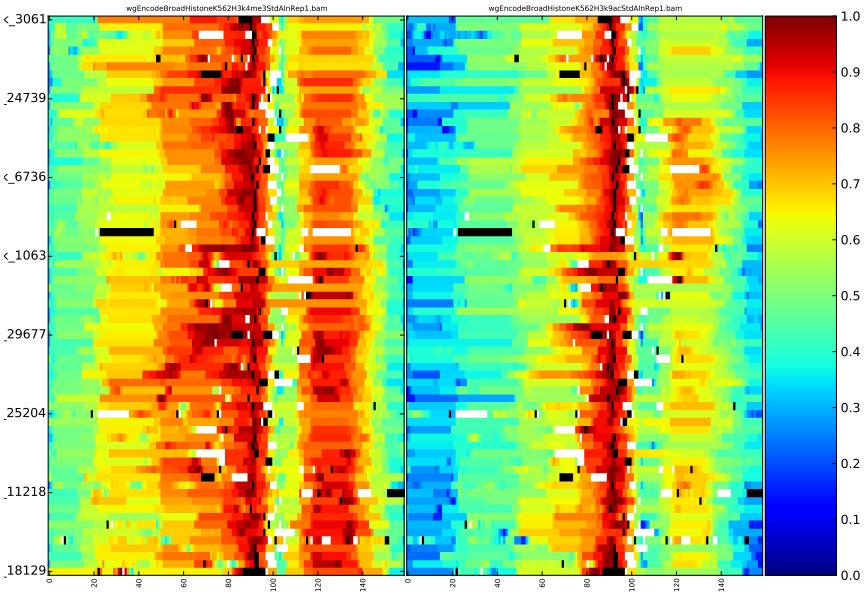
Perhaps more interestingly, a few unusual patterns can be seen from the prototypes. For instance, the pattern in 6.7b has two distinct valleys, rather than one. Prototypes of clusters pictured in 6.7f, 6.7g and 6.7i seem to be much more narrow than the remaining clusters, with only a little evidence of bimodality.



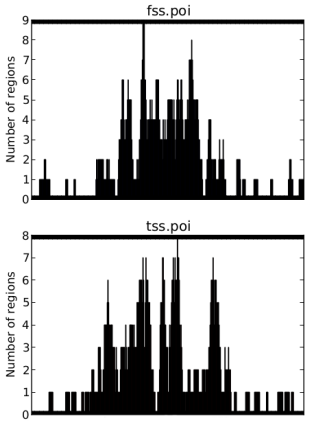
(a) Resulting Dendrogram



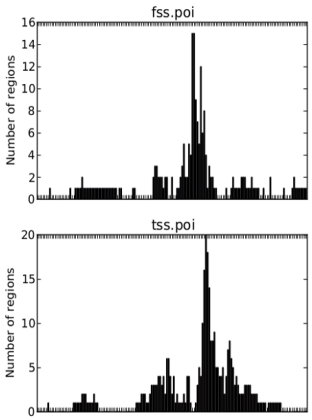
(b) Explorer window for cluster 35



(c) Cluster data projected onto the cluster prototype. First splicing sites are marked in black, whereas the trasncription start sites are marked in white.



(d) Distribution of POI before warping.



(e) Distribution of POI after warping.

Figure 6.5: Results of clustering the peaks found by MACS peak caller. Transcription start sites are marked in white and the first splicing sites are marked in black in the heatmaps. The two histograms in 6.5d and 6.5e show the distribution of the number of regions containing a transcription start site or a first splicing site in each of the bins. For original data, all regions were stretched to be of the same length before calculating the distributions.

These patterns can be studied in help with the cluster heatmaps, pictured in Figures 6.8 and 6.9.

Note that common patterns between the regions can be spotted by inspecting the original data visually. This is because the experiment was designed with this in mind: regions are all of the same length, and share a common landmark – the transcription start site in the middle. In a general case, one would not not have such luxury, as seen in the previous experiment where MACS peaks were clustered.

Now if the patterns were identifiable from the original data, they are unmistakable in the warped data. For instance, the regions in Figure 6.8a look bimodal. After warping, these patterns are shown to actually be unimodal as seen in Figure 6.9a, suggesting that the bimodality of the two patterns was just an illusion created by a mixture of sense and antisense patterns. From the warped heatmap, it is also clear that patterns in e.g. Figure 6.9e or Figure 6.9j are bimodal.

The warping seems to uncover some curious features of the data, that are not visible in the unprocessed heatmaps. For instance, the already mentioned trimodal pattern in cluster Figure 6.9b. An interesting repeating pattern is seen in some clusters, e.g. Figure 6.9f and Figure 6.9i.

The hypotheses about the origin of these patterns can be made and verified using DGW. For instance, the repeating pattern seen in the previous chart can be hypothesised to reflect the nucleosome positioning in these regions. This hypothesis could then be tested by either clustering the ChIP-Seq data alongside the dataset from a NGS experiment designed to capture nucleosome boundaries; or the annotated nucleosome positions, if known, could be visualised using the point-of-interest tracking mechanism. Neither of those analyses have been performed in this project, however, and are listed here as possible directions for further research.

Please also note that the cluster projections are not perfect. The warped heatmaps have a noticeable amount of "bleeding" away from the prototype shape. This *bleeding* is clearly visible in, say, figure Figure 6.9d, where the central patterns seem to have a bit more red in the left-hand side. This *bleeding* suggests that there is a definite sub-cluster of data within this cluster that is close enough to the other items to be assigned to the same cluster, yet different enough from them so the prototype is not able to model it correctly.

This might not be desirable when analysing the dataset using points-of-interest tracking as this mechanism relies on the prototype to model the data well. Due to this, it might be desirable to cut the dendrogram a bit closer to the leaves, as to separate these subclusters into distinct clusters. This separation would mean that each of the subpatterns will get it's own prototype, what should model the underlying data better.

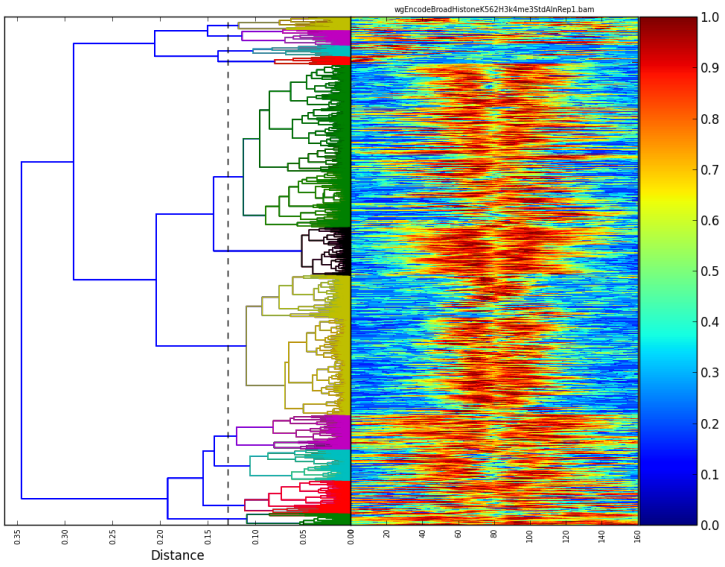


Figure 6.6: Hierarchical clustering of H3K4Me3 marks around transcription start sites of known genes, as described in section 6.3. The threshold for forming flat clusters is drawn as dashed line, and the resulting clusters are colour-coded. Prototypes for each of these 11 clusters are shown in Figure 6.7

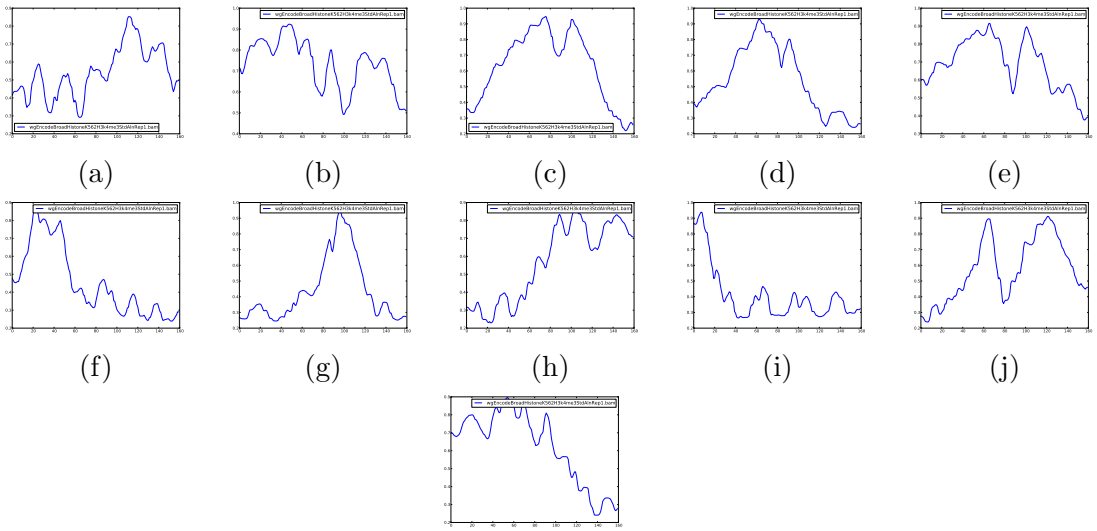


Figure 6.7: Prototypes of eleven clusters resulting from the hierarchical clustering in Figure 6.6. Well-documented bimodal patterns are seen in some of the clusters, e.g. (c), (d), (e), (j). Other prototypes seem to show an intriguing set of patterns. For instance, cluster (b) seems to have two throughs rather than one, or clusters (f) and (i) that seem to capture a set of regions that are not activated at the annotated transcription start sites, but are rather active in one of the edges of the window. The actual contents of these clusters can be seen in Figures 6.8 and 6.9

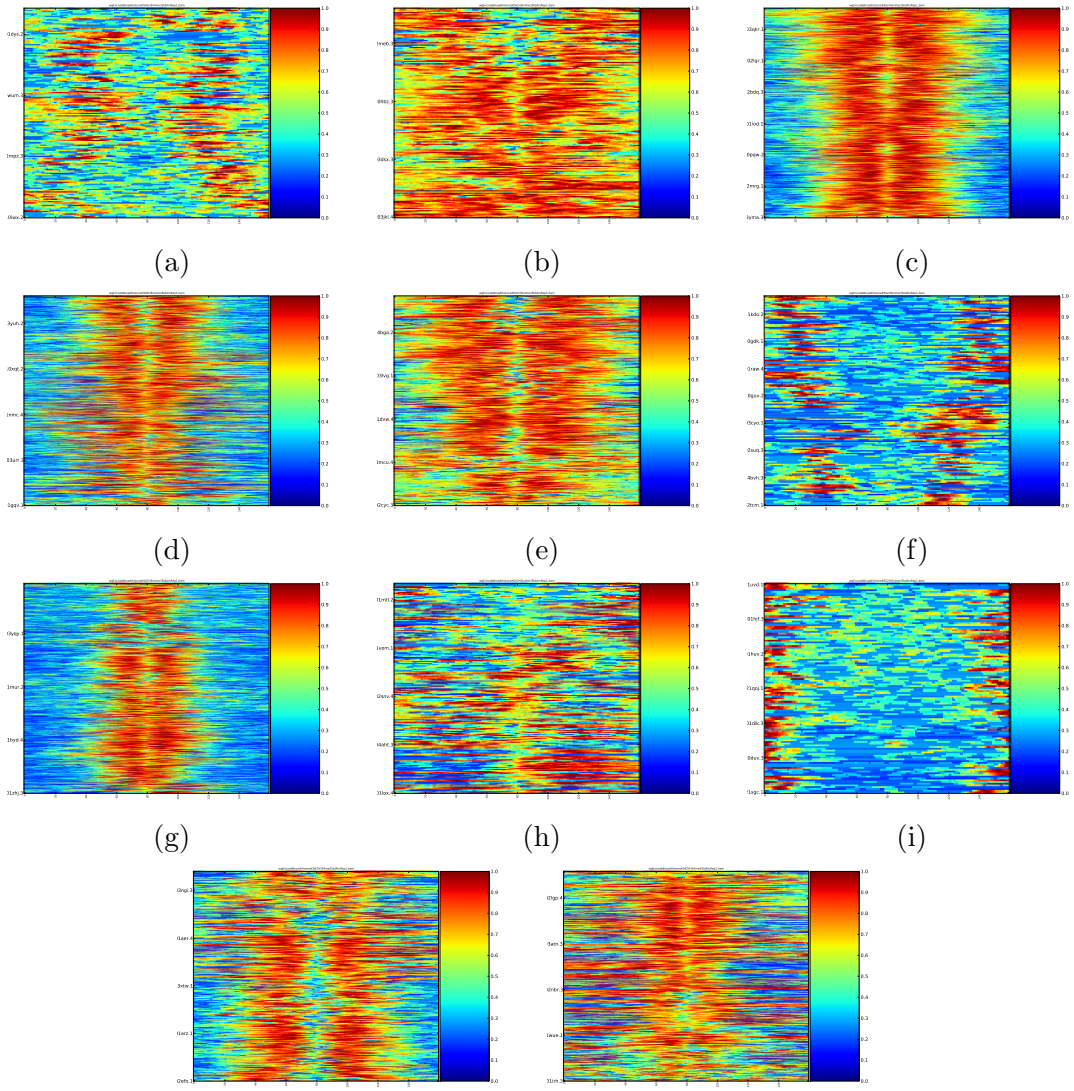


Figure 6.8: Regions forming the eleven clusters resulting from the hierarchical clustering in Figure 6.6. Actual, unwarped, data is pictured in this figure. Looking closely at the figures, common patterns of items within the clusters can be seen from this data. However, it is the warped data, that makes these patterns obvious, as seen in Figure 6.9. Note that some clusters, e.g. clusters (g) and (k), have a visible division of the items inside them, suggesting that the dendrogram may be cut a bit lower to capture the specifics of these sub-clusters better.

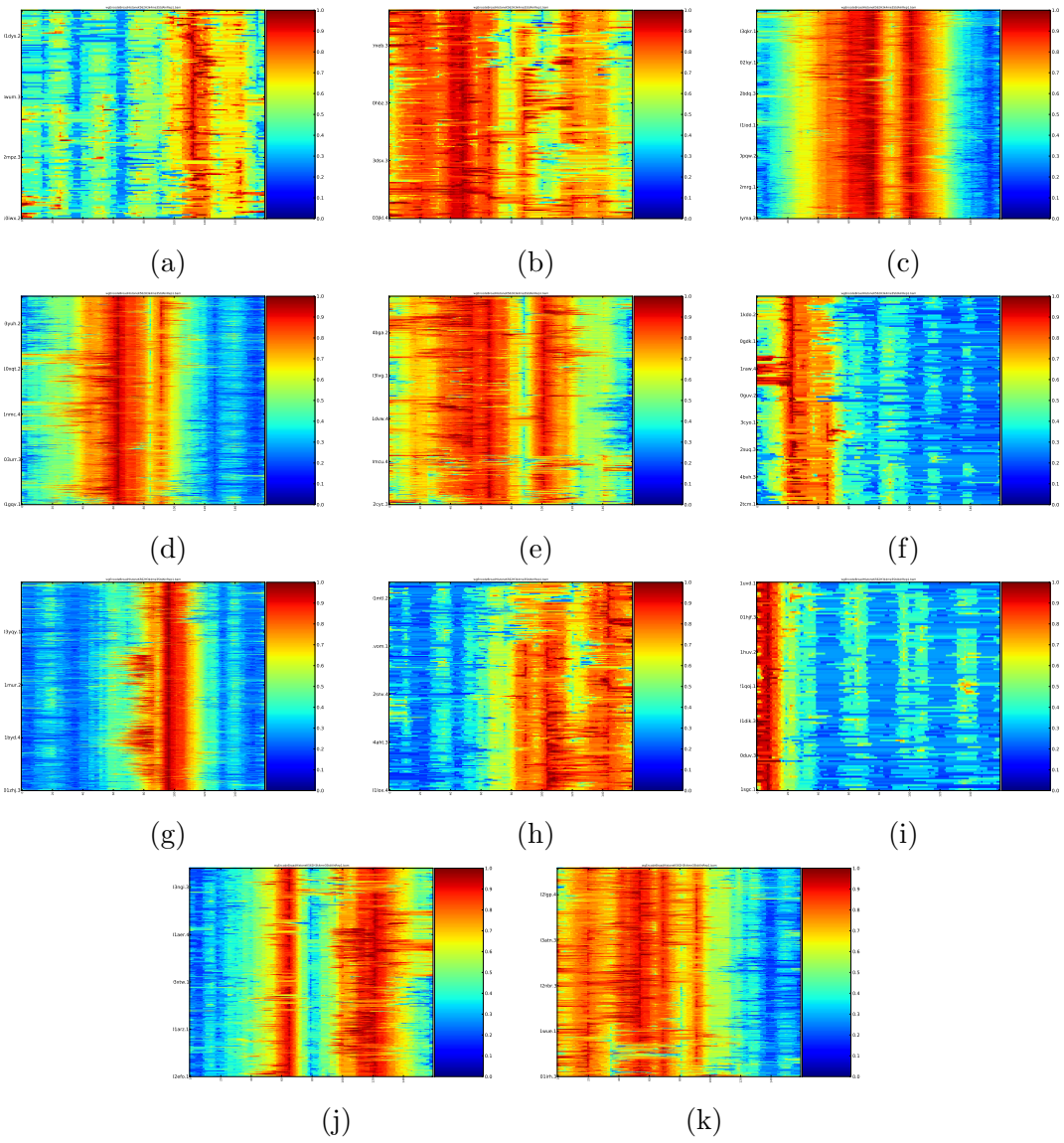


Figure 6.9: Clustered regions warped onto their cluster prototypes. Compare this figure with Figure 6.8 in terms of visualisation of the cluster contents. The patterns inside the clusters seem to be easier to distinguish here, rather than in Figure 6.8. Also note how the deviations from the prototype patterns, e.g. clearly visible in (g) where there the highly-expressed red region seams to be leaking to the left from the center, are grouped together and seem to correlate with the visible subgroups of the patterns seen in Figure 6.8. This indicates that the prototype is not able to capture the diversity of the data completely, suggesting the dendrogram should be cut at a lower threshold to separate these sub-clusters, if that is desired.

7. Conclusions

In conclusion, the project aimed to develop a method that would allow clustering of the epigenetic marks from next-generation sequencing experiments.

This was achieved by proposing a new method, Dynamic Genome Warping (DGW), that combines Dynamic Time Warping, a well-established method from speech recognition community, with hierarchical clustering to group similar histone marks together.

The software package that implements this method efficiently has been released to the Python Package Index (`https://pypi.python.org/pypi`). The source code and the documentation of the software have been made public via Github (`http://sauliusl.github.com/dgw/`).

Pilot experiments have shown this method to succeed in align the distinct epigenetic marks present near $5'$ splicing sites as first noted by [BOSN12] in a completely unsupervised way. Another experiment has shown the method's ability to detect and visualise unusual histone modification profiles near the transcription sites.

These experiments prove the effectiveness of this method for exploratory analysis of the data. The next logical step for this project would be to use this tool in order to make and test biological hypotheses about the origin of the clustered patterns, however, realising the author's limited background in biology, and the tight deadline of the project, this extensive downstream analysis was not performed.

Instead, an effort has been made to spread the word about this method and what it is capable of to the scientific communities that might be interested.

Specifically, the project has already been agreed to be presented in the poster session of *The Next NGS Challenge: Data Processing and Integration* conference (`http://www.thenextngschallenge.org/`). Furthermore, an application note describing the method and the MACS peak clustering experiment, has also been submitted to *Bioinformatics* (published by Oxford University Press) and is currently under peer review. This application note is attached in the appendix.

With a bit of luck, the research community will find this way of looking to ChIP-Seq data interesting, and will be able to generate useful insights out of it.

From the informatics perspective, further research could focus on improvement of this method. The author believes that improvement of the dendrogram cutting mechanism would be the most valuable contribution to this method. This improvement could be achieved by either providing meaningful guidance on where

the best cut is, or by defining a metric upon which the cluster assignments could be automated.

Future research may also focus on improving the prototyping algorithm, either by getting closer to the true centroid sequence, or by finding a better way of dealing with the ever increasing length of the prototypes than the one described in subsubsection 5.3.1.2.

Together these improvements could control the *bleeding* effect seen in Figure 6.9.

Finally, pivotal uses for the method proposed in this project could also be explored. For instance, instead of looking for a way to group similar patterns within the regions of interest into clusters, this method could be applied to study motifs that are warped very little, or not warped at all. This is essentially similar to the idea behind the ChromaSig approach described earlier ([HRW08]). Conversely, regions of the genome that are almost always warped to a shorter region, would be also interesting to study as to find the reason why.

To summarise, the project has succeeded in developing a novel method to look at the next-generation sequencing data. The practicality of this tool has been demonstrated by showing how previously described genomic features can be recaptured using this tool; as well as demonstrating how the visualisations of the warped data be helpful in the search for anomalous patterns within the data. Ground work for introducing this project to the research community has already been laid, hopefully aiding the formulation of hypotheses from ChIP-Seq data for the time to come.

Bibliography

[AJL+02] Bruce Alberts, Alexander Johnson, Julian Lewis, Peter Walter, Martin Raff, and Keith Roberts, *Molecular Biology of the Cell 4th Edition*, International Student Edition, Garland Science, May 2002.

[Ann08] A Annunziato, *DNA Packaging: Nucleosomes and Chromatin — Learn Science at Scitable*, Nature Education (2008).

[AVM+12] Davide Albanese, Roberto Visintainer, Stefano Merler, Samantha Riccadonna, Giuseppe Jurman, and Cesare Furlanello, *mlpy: Machine Learning Python*, ... preprint arXiv:12026548 (2012).

[BCC+07] Artem Barski, Suresh Cuddapah, Kairong Cui, Tae-Young Roh, Dustin E Schones, Zhibin Wang, Gang Wei, Iouri Chepelev, and Keji Zhao, *High-resolution profiling of histone methylations in the human genome*, Cell **129** (2007), no. 4, 823–837.

[BOSN12] Nicole I Bieberstein, Fernando Carrillo Oesterreich, Korinna Straube, and Karla M Neugebauer, *First Exon Length Controls Active Chromatin Signatures and Transcription*, Cell Reports (2012).

[EBS+07] ENCODE Project Consortium, Ewan Birney, John A Stamatoyannopoulos, Anindya Dutta, Roderic Guigó, Thomas R Gingeras, et al., *Identification and analysis of functional elements in 1pilot project.*, Nature ... **447** (2007), no. 7146, 799–816.

[Fur12] Terrence S Furey, *ChIP–seq and beyond: new and improved methodologies to detect and characterize protein–DNA interactions*, Nature Reviews Genetics **13** (2012), no. 12, 840–852.

[FWA03] W Fischle, Y Wang, and C D Allis, *Histone and chromatin cross-talk*, Current opinion in cell biology (2003).

[GBEa97] W N Grundy, T L Bailey, C P Elkan, and et al, *Meta-MEME: motif-based hidden Markov models of protein families*, ... applications in the ... (1997).

[Gio09] T Giorgino, *Computing and visualizing dynamic time warping alignments in R: The dtw package*, Journal of Statistical Software (2009).

[GMTS96] Lalit Gupta, Dennis L Molfese, Ravi Tammana, and Panagiotis G Simos, *Nonlinear alignment and averaging for estimating the evoked potential*, Biomedical Engineering, IEEE Transactions on **43** (1996), no. 4, 348–356.

[HKC+06] Fan Hsu, W James Kent, Hiram Clawson, Robert M Kuhn, Mark Diekhans, and David Haussler, *The UCSC Known Genes*, Bioinformatics **22** (2006), no. 9, 1036–1046.

[HNF08] Ville Hautamaki, Pekka Nykanen, and Pasi Franti, *Time-series clustering by approximate prototypes*, 2008 19th International Conference on Pattern Recognition (ICPR), IEEE, 2008, pp. 1–4.

[HRW08] G Hon, B Ren, and W Wang, *ChromaSig: a probabilistic approach to finding common chromatin signatures in the human genome*, PLoS Comput. Biol. (2008).

[HSH+07] Nathaniel D Heintzman, Rhona K Stuart, Gary Hon, Yutao Fu, Christina W Ching, R David Hawkins, Leah O Barrera, Sara Van Calcar, Chunxu Qu, Keith A Ching, Wei Wang, Zhiping Weng, Roland D Green, Gregory E Crawford, and Bing Ren, *Distinct and predictive chromatin signatures of transcriptional promoters and enhancers in the human genome*, Nat Genet **39** (2007), no. 3, 311–318.

[HT05] S Hirano and S Tsumoto, *Empirical Comparison of Clustering Methods for Long Time-Series Databases - Springer*, Active Mining (2005).

[Hun07] John D Hunter, *Matplotlib: A 2D graphics environment*, Computing in Science & Engineering (2007), 90–95.

[Kou07] Tony Kouzarides, *Chromatin Modifications and Their Function*, Cell **128** (2007), no. 4, 13.

[KSF+02] W James Kent, Charles W Sugnet, Terrence S Furey, Krishna M Roskin, Tom H Pringle, Alan M Zahler, and David Haussler, *The Human Genome Browser at UCSC*, Genome ... (2002).

[LB10] W Lai and M J Buck, *ArchAlign: coordinate-free chromatin alignment reveals novel architectures*, Genome Biol (2010).

[LHW+09] Heng Li, Bob Handsaker, Alec Wysoker, Tim Fennell, Jue Ruan, Nils Homer, Gabor Marth, Goncalo Abecasis, Richard Durbin, and 1000 Genome Project Data Processing Subgroup, *The Sequence Alignment/Map format and SAMtools.*, Bioinformatics **25** (2009), no. 16, 2078–2079.

[Mar07] E R Mardis, *ChIP-seq: welcome to the new frontier*, Nature methods (2007).

[Met10] Michael L Metzker, *Sequencing technologies—the next generation*, Nature Reviews Genetics **11** (2010), no. 1, 31–46.

[Mül07] Meinard Müller, *Dynamic Time Warping*, springer.com, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 69–84.

[Mül11] Daniel Müllner, *Modern hierarchical, agglomerative clustering algorithms*, arXiv (2011).

[NR07] V Niennattrakul and C A Ratanamahatana, *Inaccuracies of Shape Averaging Method Using Dynamic Time Warping for Time Series Data - Springer*, Computational Science–ICCS 2007 (2007).

[NR09] Vit Niennattrakul and Chotirat Ann Ratanamahatana, *Shape averaging under Time Warping*, 2009 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), IEEE, 2009, pp. 626–629.

[Par09] P J Park, *ChIP–seq: advantages and challenges of a maturing technology*, Nature Reviews Genetics (2009).

[PG12] François Petitjean and Pierre Gançarski, *Summarizing a set of time series by averaging: From Steiner sequence to compact multiple alignment*, Theoretical Computer Science **414** (2012), no. 1, 76–91.

[PHLC05] D K Pokholok, C T Harbison, S Levine, and M Cole, *ScienceDirect.com - Cell - Genome-wide Map of Nucleosome Acetylation and Methylation in Yeast*, Cell (2005).

[PJ09] Hae-Sang Park and Chi-Hyuck Jun, *A simple and fast algorithm for K-medoids clustering*, Expert Systems with Applications **36** (2009), no. 2, 3336–3341.

[PKG11] François Petitjean, Alain Ketterlin, and Pierre Gançarski, *A global averaging method for dynamic time warping, with applications to clustering*, Pattern Recognition **44** (2011), no. 3, 678–693.

[QH10] Aaron R Quinlan and Ira M Hall, *BEDTools: a flexible suite of utilities for comparing genomic features.*

[RDL+11] K R Rosenbloom, T R Dreszer, J C Long, V S Malladi, C A Sloan, B J Raney, M S Cline, D Karolchik, G P Barber, H Clawson, M Diekhans, P A Fujita, M Goldman, R C Gravell, R A Harte, A S Hinrichs, V M Kirkup, R M Kuhn, K Learned, M Maddren, L R Meyer, A Pohl, B Rhead, M C Wong, A S Zweig, D Haussler, and W J Kent, *ENCODE whole-genome data in the UCSC Genome Browser: update 2012*, Nucleic Acids Research **40** (2011), no. D1, D912–D917.

[RK04] Chotirat Ann Ratanamahatana and Eamonn Keogh, *Everything you know about Dynamic Time Warping is Wrong*, Third Workshop on Mining Temporal and Sequential Data (2004), 22–25.

[SB07] Christoph D Schmid and Philipp Bucher, *ChIP-Seq data reveal nucleosome architecture of human promoters*, Cell **131** (2007).

[SC78] H Sakoe and S Chiba, *Dynamic programming algorithm optimization for spoken word recognition*, Acoustics (1978).

[SK06] M Shogren-Knaak, *Histone H4-K16 Acetylation Controls Chromatin Structure and Protein Interactions*, Science **311** (2006), no. 5762, 844–847.

[The04] The ENCODE Project Consortium, *The ENCODE (ENCyclopedia Of DNA Elements) Project*, Science **306** (2004), no. 5696, 636–640.

[TRM12] H Thorvaldsdóttir, J T Robinson, and J P Mesirov, *Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration*, Briefings in bioinformatics (2012).

[TWY+09] Cenny Taslim, Jiejun Wu, Pearlly Yan, Greg Singer, Jeffrey Parvin, Tim Huang, Shili Lin, and Kun Huang, *Comparative study on ChIP-seq data: normalization and binding pattern characterization.*, Bioinformatics **25** (2009), no. 18, 2334–2340.

[WLJ12] J Wang, V V Lunyak, and I K Jordan, *Chromatin signature discovery via histone modification profile alignments*, Nucleic Acids Research **40** (2012), no. 21, 10642–10656.

[ZLM+08] Y Zhang, T Liu, C A Meyer, J Eeckhoute, and et al, *Model-based analysis of ChIP-Seq (MACS)*, Genome ... (2008).

Appendix A. Application Note About This Project

The application note that has been submitted to *Bioinformatics* (Oxford University Press) is attached to the subsequent pages of this appendix.

BIOINFORMATICS

Vol. 00 no. 00 2005
Pages 1–2

DGW: an exploratory data analysis tool for clustering and visualisation of epigenomic marks

Saulius Lukauskas[1],Gabriele Schweikert[1,2], Guido Sanguinetti[1,3*]

[1]School of Informatics, [2]Wellcome Trust Centre for Cell Biology, [3] SynthSys, Synthetic and Systems Biology, University of Edinburgh, Edinburgh, UK

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Associate Editor: XXXXXXX

ABSTRACT

Motivation: Novel sequencing based technologies such as ChIP-seq are rapidly becoming the dominant experimental tool in functional genomics and epigenomics. These techniques return very large, high dimensional data sets with visually complex structures, such as multimodal peaks extended over large genomic regions. Intuitive tools for visualisation and data exploration which can capture and leverage these complex features are currently lacking.

Results: We present DGW, an open source software package for simultaneous clustering and alignment of multiple epigenomic marks. DGW uses Dynamic Time Warping to capture the structure of epigenomic marks, by adaptively rescaling genomic distances to group peaks with similar shapes. We demonstrate the effectiveness of the approach on an epigenomic data set from the ENCODE project, and show that DGW automatically recognises and aligns important genomic features such as transcription start sites and splicing sites from histone marks.

Availability: DGW is available as an open source Python package at `http://sauliusl.github.com/dgw/`.

Contact: `gsanguin@inf.ed.ac.uk`

1 INTRODUCTION

Novel sequencing based technologies such as ChIP-Seq and DNAse-Seq are revolutionizing our understanding of chromatin structure and function, yielding deep insights in the importance of epigenomic marks in the basic processes of life. The emergent picture is that gene expression is controlled by a complex interplay of protein binding and epigenomic modifications (ENCODE Project Consortium, 2012). While histone marks (and other epigenomic marks) can be measured in a high throughput way, exploratory data analysis techniques for these data types are still largely lacking. Epigenomic marks exhibit characteristics that distinguish them fundamentally from e.g. mRNA gene expression measurements: they are spatially extended across regions as wide as several kilobases, and often present interesting local structures, such as the presence of multiple peaks and troughs. These patterns often have a biological origin, such as the displacement of a nucleosome or the length of the first exon of a gene (Bieberstein *et al.*, 2012), so that analysis tools that take into account these spatial features would be desirable. However, each (combination of) epigenomic mark(s) at different locations in principle represents a multivariate data point of different length (as peaks for the same mark in different locations can have widely differing lengths): this prevents the straightforward extension of well established data analysis techniques such as hierarchical clustering to these data types. In this work, we present Dynamic Genome Warping (DGW), an open source clustering and alignment tool for epigenomic marks which addresses this problem by introducing a local rescaling which allows to match (multiple) epigenomic marks based on maximum similarity between shapes. DGW is freely available as a stand-alone, platform independent and fully documented Python package.

*to whom correspondence should be addressed

2 METHODS AND SOFTWARE

DGW is based on the classic Dynamic Time Warping algorithm; this was originally introduced in the speech recognition community to robustly recognize speech independently of speech speed. Likewise, our aim is to be able to associate peaks which exhibit similar local structure (shape) regardless of their spatial extension. Specifically, given two sequences $\mathbf{a} = (a_1, \ldots, a_N)$ and $\mathbf{b} = (b_1, \ldots, b_M)$, and a local distance between the elements of each sequence (e.g. Euclidean distance or Cosine distance), DGW uses dynamic programming to construct a warping path, i.e. a sequence of points in the two sequences that are mapped to each other. The warping path has the property of minimising the sum of the distances between the aligned points (the warping distance); furthermore, it is monotonic (i.e. there are no inversions in each sequence) and maps the first and last point of sequence $\mathbf{a}$ to the first and last point of sequence $\mathbf{b}$. Figure 1 A gives an example of how two peaks are warped onto each other. A modern review of the basic concepts of Dynamic Time Warping can be found in e.g. Muller (2007).

DGW builds on a number of open source Python packages for numerical computation and machine learning; a full list of these packages and installation instructions for Linux, Windows and Mac platforms are available in the package documentation. DGW consists of two modules: a worker module, which performs the computationally intensive tasks, and an explorer module (described in the results section), which allows visual exploration of the results. DGW-worker is called from the command line and takes as input a set of genomic regions (as a .bed file outputted by a peak finder) and a set of data files (.bam files) for different epigenomic marks. Additional inputs, such as additional files specifying regions of interests (e.g. transcription start sites or first splicing sites), can be passed using specific options (detailed in the package documentation). DGW-worker then computes the warping distances, prototypes and clustering; this is computationally intensive and DGW-worker will automatically distribute the work across multiple cores if available. A typical run of DGW worker on the ChIP-seq data set described below took 420 mins of CPU time distributed across six cores, for a total execution time of just over one hour.

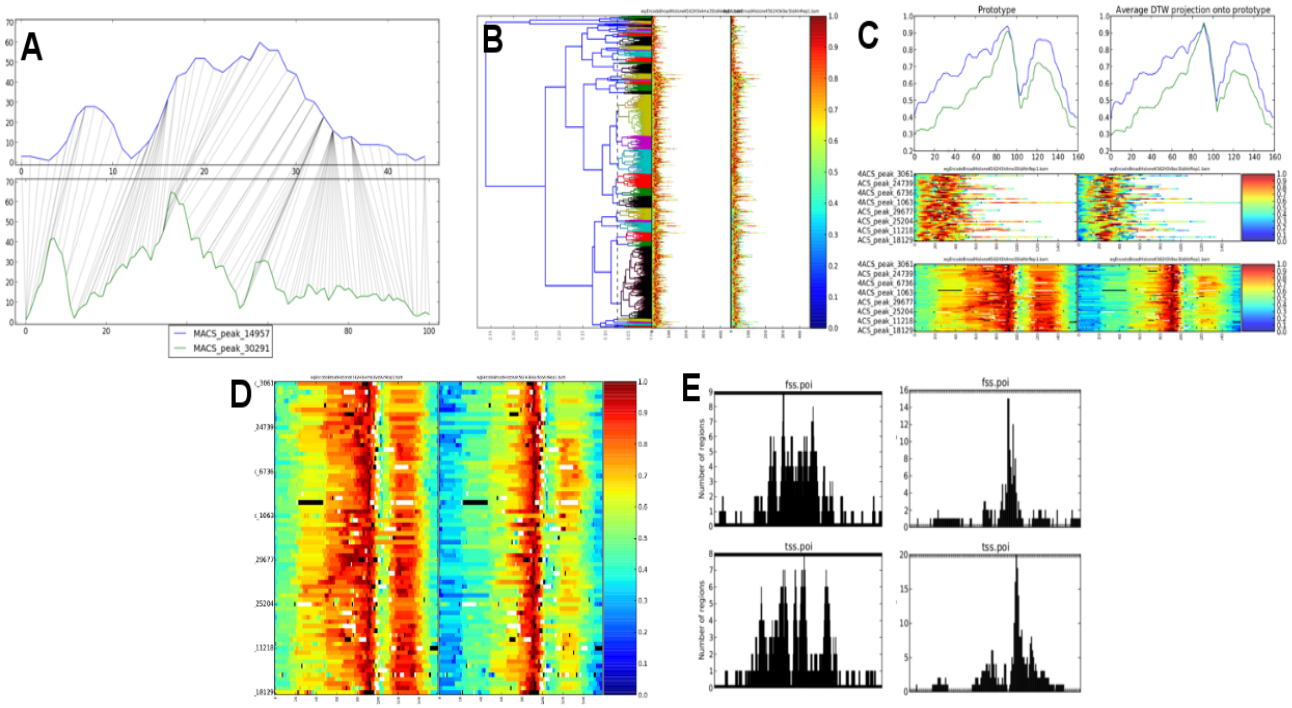


Fig. 1. Illustrative results of DGW: A, warping of peaks; B dendrogram and heatmap of raw data; C, explorer window for a specific cluster; D, heatmap of warped data, with first splicing sites and transcription start sites visible as black and white dots respectively; E, histogram of transcription start sites and first splicing sites in raw data (left) and in warped data (right)

3 EXAMPLE RESULTS

Results on simulated data sets confirm the correctness of the implementation and are displayed in the software documentation. Here we show the use of DGW-explorer on an ENCODE ChIP-Seq data set of histone 3 lysin 4 tri-methylation (H3K4me3) and histone 3 lysin 9 acetylation (H3K9ac) in the leukaemia cell line K562 (accession code wgEncodeBroadHistoneK562). Figure 1 B-E shows the results of this analysis. Upon calling the explorer, a dendrogram with the associated heat map of the peaks is displayed (B; notice the widely varying width of the peaks): the user can adaptively select where to cut the dendrogram by sliding the vertical line. Empirically, cutting the dendrogram near the leaves gives better visualisations, as large clusters force the algorithm to warp together potentially very different peaks. The individual clusters can then be explored (C) in a new window, displaying the prototypes for the marks and the heat maps of the raw and warped data; the heat map of the warped data is shown in a larger panel in D, with first splicing sites and transcription start sites visible as black and white dots respectively. Histograms of the distribution of points of interest in the raw and warped data can be displayed by clicking on the POI button: E shows the histograms for transcription start sites and first splicing sites; notice how the histogram is more peaked in the warped data (right), indicating that these meaningful biological landmarks were automatically aligned by DGW. Clicking on an individual peak in the heat map displays the corresponding marks and a visualisation of how these were mapped onto the prototype. Further details and examples are given in the software documentation.

These results show that DGW provides a practical and user friendly tool for exploratory data analysis of high throughput epigenomic data sets, and that, by exploiting shape features of peaks, DGW can recapture known biological feaures, such as the accumulation of H3K4me3 at first splicing sites (Bieberstein *et al.*, 2012) and the depletion at transcription start sites, and hopefully help in formulating further hypotheses.

ACKNOWLEDGEMENTS

G.Sc. acknowledges funding from the EU under the Marie Curie actions. G.S. acknewledges funding from the ERC under grant MLC366999.

REFERENCES

Bieberstein, N. I., Carrillo Oesterreich, F., Straube, K., and Neugebauer, K. M. (2012). First exon length controls active chromatin signatures and transcription. *Cell Rep*, **2**(1), 62–8.

ENCODE Project Consortium (2012). An integrated encyclopedia of dna elements in the human genome. *Nature*, **489**(7414), 57–74.

Muller, M. (2007). *Information Retrieval for Music and Motion*. Springer.