Inaugural dissertation

for

obtaining the doctoral degree

of the

Combined Faculty of Mathematics, Engineering and Natural Sciences

of the

Ruprecht - Karls - University

Heidelberg

Presented by

Rene Helmut Snajder, M.Sc

born in: Wagna, Austria

Oral examination: 2023-03-24

Methods for Epigenetic Analyses from Long-Read Sequencing Data

Referees: Prof. Dr. Benedikt Brors
Prof. Dr. Oliver Stegle
Prof. Dr. Karl Rohr
Prof. Dr. Moritz Gerstung

Abstract

Epigenetics, particularly the study of DNA methylation, is a cornerstone field for our understanding of human development and disease. DNA methylation has been included in the "hallmarks of cancer" due to its important function as a biomarker and its contribution to carcinogenesis and cancer cell plasticity. Long-read sequencing technologies, such as the Oxford Nanopore Technologies platform, have evolved the study of structural variations, while at the same time allowing direct measurement of DNA methylation on the same reads. With this, new avenues of analysis have opened up, such as long-range allele-specific methylation analysis, methylation analysis on structural variations, or relating nearby epigenetic modalities on the same read to another.

Basecalling and methylation calling of Nanopore reads is a computationally expensive task which requires complex machine learning architectures. Read-level methylation calls require different approaches to data management and analysis than ones developed for methylation frequencies measured from short-read technologies or array data. The 2-dimensional nature of read and genome associated DNA methylation calls, including methylation caller uncertainties, are much more storage costly than 1-dimensional methylation frequencies. Methods for storage, retrieval, and analysis of such data therefore require careful consideration. Downstream analysis tasks, such as methylation segmentation or differential methylation calling, have the potential of benefiting from read information and allow uncertainty propagation. These avenues had not been considered in existing tools.

In my work, I explored the potential of long-read DNA methylation analysis and tackled some of the challenges of data management and downstream analysis using state of the art software architecture and machine learning methods.

I defined a storage standard for reference anchored and read assigned DNA methylation calls, including methylation calling uncertainties and read annotations such as haplotype or sample information. This storage container is defined as a schema for the hierarchical data format version 5, includes an index for rapid access to genomic coordinates, and is optimized for parallel computing with even load balancing. It further includes a python API for creation, modification, and data access, including convenience functions for the extraction of important quality statistics via a command line interface. Furthermore, I developed software solutions for the segmentation and differential methylation testing of DNA methylation calls from Nanopore sequencing. This implementation takes advantage of the performance benefits pro-

vided by my high performance storage container. It includes a Bayesian methylome segmentation algorithm which allows for the consensus instance segmentation of multiple sample and/or haplotype assigned DNA methylation profiles, while considering methylation calling uncertainties. Based on this segmentation, the software can then perform differential methylation testing and provides a large number of options for statistical testing and multiple testing correction.

I benchmarked all tools on both simulated and publicly available real data, and show the performance benefits compared to previously existing and concurrently developed solutions. Next, I applied the methods to a cancer study on a chromothriptic cancer sample from a patient with Sonic Hedgehog Medulloblastoma. I here report regulatory genomic regions differentially methylated before and after treatment, allele-specific methylation in the tumor, as well as methylation on chromothriptic structures.

Finally, I developed specialized methylation callers for the combined DNA methylation profiling of CpG, GpC, and context-free adenine methylation. These callers can be used to measure chromatin accessibility in a NOMe-seq like setup, showing the potential of long-read sequencing for the profiling of transcription factor co-binding.

In conclusion, this thesis presents and subsequently benchmarks new algorithmic and infrastructural solutions for the analysis of DNA methylation data from long-read sequencing.

Zusammenfassung

Das Studium der Epigenetik, insbesondere in Bezug auf DNA-Methylierung, ist von großer Bedeutung für unser Verständnis der menschlichen Entwicklung und Pathologie. Epigenetik spielt eine wichtige Rolle als Biomarker und hat Einfluss auf die Karzinogenese und die Plastizität von Krebszellen, weshalb sie auch mittlerweile als eines der "Hallmarks of Cancer" verstanden wird. Die Entwicklung von Long Read Sequencing Technologien, d.h. die Sequenzierung von langen DNA Molekülen, hatte einen großen Einfluss auf die Analyse von strukturellen Varianten und erlaubte die Gruppierung von gelesenen Sequenzen in Haplotypen. Des Weiteren ermöglichen Long Read Sequencing Technologien die direkte Messung des DNA-Methylierungsstatus gemeinsam mit genetischer Variation auf denselben DNA Molekülen. Daraus entspringende Möglichkeiten zur integrativen Analyse beinhalten unter anderem die allelspezifische Methylierungsanalyse, die Untersuchung von Methylierung in Relation zu struktureller Variation, und die Verknüpfung nahegelegener epigenetischer Modalitäten.

Sowohl die Übersetzung der gemessenen Nanopore Signale auf eine DNA Sequenz, als auch die Bestimmung des Methylierungsstatus, sind rechenintensive Aufgaben und benötigen komplexe mathematische Modelle. Methylierungsdaten mit Read-Information unterscheiden sich in ihren Anforderungen an die Datenverwaltung und Analyse von solchen, die durch Short Read oder Array Technologien erlangt wurden. Die zweidimensionale Struktur von Methylierungsdaten mit Read-Zuweisung und genomischen Koordinaten, sowie die zusätzlichen Messunsicherheiten der Methylierungsbestimmung, haben viel kostspieligere Anforderungen an den Datenstrukturen als gewöhnliche eindimensionale Methylierungsfrequenzdaten. Methoden zur Datenspeicherung, -abfrage, und -analyse sollten daher mit besonderer Aufmerksamkeit auf diese Eigenschaften entwickelt werden. Weiterführende Analysen, so wie die Segmentierung des Methyloms oder die differentielle Methylieriungsanalyse, können von diesen zusätzlichen Informationen profitieren und unter Berücksichtigung der Fortpflanzung der Messunsicherheit stattfinden.

In meiner Arbeit untersuchte ich das Potential von Long Read Methylierungsanalysen und stelle mich mit Hilfe moderner Software Architekturen und Machine Learning Methoden den Herausforderungen der Datenverwaltung und Analyse.

Ich entwickelte ein Dateiformat für Methylierungsdaten mit Read-Information und genomischen Koordinaten, welches auch Messunsicherheiten speichert. Zusätzlich können Reads mit Annotationen, so wie die Zugehörigkeit zu Proben oder Haplo-

typen, versehen werden. Das Dateiformat wurde als Schema für das Hierarchical Data Format Version 5 definiert, beinhaltet einen Index für schnellen Zugriff auf genomische Koordinaten, und ist optimiert für paralleles Rechnen mit gleichmäßiger Lastverteilung. Ergänzend wurde das Dateiformat gemeinsam mit einer python API veröffentlicht, welche die Erstellung, Manipulation, und Abfrage von Methylieriungsdaten ermöglicht. Als nächstes entwickelte ich Software für die Methylomsegmentierung und differentielle Methylierungsanalyse basierend auf Nanopore Sequenzierungsdaten. Die Methylomsegmentierung wurde hier als Bayessche Methode implementiert und erlaubt es, eine gemeinsame Segmentierung von multiplen Methylieriungsprofilen, wie z.B. von mehreren biologischen Proben oder Haplotypen, durchzuführen. Das Bayessche Modell berücksichtigt dabei die Messunsicherheiten der Methylierungsdaten. Segmente können dann von der Software auf differentielle Methylierung zwischen Gruppen untersucht werden. Dabei stehen dem Benutzer eine Vielzahl an Optionen für statistische Tests und zur Beherrschung des multiplen Testproblems zur Verfügung.

Ich überprüfte die genannten Softwarewerkzeuge in einem Benchmark-Test überprüft, sowohl auf simulierten als auch auf echten Methylierungsdaten, und zeige hier die Vorzüge im Vergleich zu bestehenden Methoden auf. Als Teil einer Krebsstudie untersuchte ich auch die allelspezifische Methylierung eines Patienten mit Sonic Hedgehog Medulloblastoma, sowie Unterschiede zwischen dem Methylierungsprofil vor und nach Behandlung und die Methylierung von strukturellen Variationen.

Im letzten Teil entwickelte ich spezialisierte Methylierungsbestimmungssoftware für die gemeinsame Bestimmung von CpG-, GpC-, und kontextfreier Adeninmethylierung. Diese Software wurde zur Messung von Chromatinzugänglichkeit in einem NOMe-seq ähnlichen Framework entwickelt, und ich zeige damit das Potential von Long Read Sequenzierungstechnologie zur Untersuchung von Transcription Factor Co-Binding.

Zusammenfassend, diese Arbeit präsentiert und testet neue algorithmische und infrastrukturelle Lösungen für die Analyse von Methylierungsdaten von Long Read Sequenzierungstechnologien.

Contents

Abstract v

Zusammenfassung vii

List of Figures xiii

List of Tables xv

Abbreviations xvii

1. Introduction 1

1.1. Epigenomics and transcriptional regulation 1

1.1.1. DNA methylation 2

1.1.2. Chromatin remodeling 11

1.1.3. State of the art technologies for epigenetics profiling 12

1.1.4. Chromatin accessibility profiling 16

1.2. Principles of machine learning in epigenetics 18

1.2.1. Bayesian methods 19

1.2.2. Deep learning 25

1.2.3. Canonical problems in epigenetics 26

1.3. Nanopore sequencing and analysis 29

1.3.1. Nanopore sequencing and basecalling 29

1.3.2. Nanopore methylation calling 31

2. MetH5 - efficent storage format for methylation calls from Nanopore 35

2.1. Requirements analysis 36

2.1.1. Functional requirements 36

2.1.2. Nonfunctional requirements 38

2.2. MetH5 implementation . . . 39
2.2.1. MetH5 as a coordinate format sparse matrix . . . 39
2.2.2. HDF5 schema defining MetH5 format . . . 40
2.3. MetH5 python API . . . 41
2.3.1. Construction of a MetH5 file . . . 42
2.3.2. Random genomic access algorithm . . . 43
2.4. Evaluation . . . 43
2.5. Code availability . . . 46
2.6. Conclusion . . . 47

3. pycoMeth - differential methylation analysis toolbox 49
3.1. Bayesian methylome segmentation algorithm . . . 50
3.1.1. Segmentation HMM . . . 52
3.1.2. Hyperparameters . . . 55
3.1.3. Evaluation . . . 56
3.2. Differential methylation calling . . . 63
3.2.1. Hypotheses and tests . . . 64
3.2.2. Adjustment for multiple testing . . . 65
3.2.3. Evaluation . . . 66
3.3. Code availability . . . 72
3.4. Conclusion . . . 72

4. Application on Medulloblastoma study 73
4.1. Contributions . . . 73
4.2. Study design . . . 74
4.2.1. Data preparation . . . 74
4.3. Differential DNA methylation between tumor samples detected by pycoMeth . . . 76
4.3.1. Functional analysis of DMRs . . . 77
4.4. Allele-specific DNA methylation detected by pycoMeth . . . 79
4.4.1. Functional analysis of ASM . . . 79
4.5. DNA Methylation patterns on structural variations . . . 80

4.6. Code availability 80
4.7. Conclusion 83

5. High resolution chromatin accessibility prediction 85
5.1. Contributions 86
5.2. Multi-modification Bayesian methylation caller 86
5.2.1. Methylation caller design 87
5.2.2. Training 91
5.2.3. Evaluation 92
5.3. Code availability 97
5.4. Conclusion 97

6. Summary 99

7. Future research 101

Appendices

A. Example uncertainty propagation 103

B. MetH5 CLI 105
B.1. Writing MetH5 files 105
B.2. Reading MetH5 files 106

C. Figures 107

Acknowledgements 115

Bibliography 117

Colophon 137

List of Figures

1.1. Epigenetics . . . 2
1.2. CpG methylation and demethylation . . . 6
1.3. CpG islands, shores, and shelves . . . 8
1.4. Transcription and epigenetic regulation . . . 13
1.5. Bisulfite sequencing . . . 15
1.6. NOMe-sequencing . . . 18
1.7. Hidden Markov Model . . . 21
1.8. Artificial Neural Networks . . . 26
1.9. DNA Methylation segmentation . . . 29
1.10. Nanopore flow cell . . . 30
1.11. Guppy ONT basecaller . . . 31

2.1. COO format sparse matrix . . . 39
2.2. MetH5 architecture . . . 42
2.3. GIAB Data preprocessing . . . 45
2.4. MetH5 benchmark . . . 46

3.1. pycoMeth suite . . . 51
3.2. Simulation of Nanopolish calls . . . 58
3.3. Segmentation benchmark comparison setup . . . 60
3.4. Segmentation evaluation method . . . 62
3.5. Segmentation performance on high coverage simulated data . . . 63
3.6. Recall, Precision, and F1-score on simulated data . . . 68
3.7. DMR predictions on GIAB data . . . 69
3.8. FDR estimation from permutation test . . . 70
3.9. Positive percent agreement . . . 71

4.1. Medulloblastoma study design 75
4.2. Overview Medulloblastoma DMR results 78
4.3. ASM vs ASE in medulloblastoma 81
4.4. Methylation of SVs in medulloblastoma 82

5.1. Flank marginalization for chromatin accessibility 90
5.2. Mapping from LLR to ASCII 91
5.3. Training of chromatin accessibility models 92
5.4. Evaluation of chromatin accessibility model 93
5.5. Classification of transcription factor binding sites 95
5.6. Power analysis by simulation 96

C.1. 2D methlyation storage 108
C.2. MetH5 load distribution 108
C.3. Low coverage simulation benchmark 109
C.4. Confounders of test power 110
C.5. Segmentation permutation validation 111
C.6. Precision and recall in low effect-size DMRs 112
C.7. DMR versus expression in medulloblastoma 113

List of Tables

5.1. Resolution by motif 86

Abbreviations

5mC	5-methylcytosine
6mA	6-methyladenine
ANN	artificial neural network
APOBEC3A	apolipoprotein B mRNA editing enzyme catalytic polypeptide-like 3A
ASIC	application specific integrated circuit
ASE	allele-specific expression
ASM	allele-specific methylation
bp(s)	base pair(s)
CCrM	Cell cycle regulated methyltransferase
CLI	Command line interface
CTC	connectionist temporal classification
Dam	deoxyribonucleic acid adenine methylase
D.mel	Drosophila melanogaster
DNA	deoxyribonucleic acid
Dnmt1	deoxyribonucleic acid methyltransferase 1
Dnmt3a	deoxyribonucleic acid methyltransferase 3 alpha
Dnmt3b	deoxyribonucleic acid methyltransferase 3 beta
DMR	differentially methylated region
FPR	false positive rate
FDR	false discovery rate
GIAB	Genome in a Bottle
GRCh38	GRCh38 Genome Reference Consortium Human Build 38
GRU	gated recurrent unit
HDF5	Hierarchical Data Format version 5

HMM	hidden markov model
I/O	input/output
lncRNA	long non-coding ribonucleic acid
MBD	methyl CpG binding domain
miRNA	micro ribonucleic acid
mRNA	messenger ribonucleic acid
ONT	Oxford Nanopore Technologies
PacBio	Pacific Biosciences
PCR	polymerase chain reaction
pp	percentage point
RNA	ribonucleic acid
RNN	recurrent neural network
SMRT-seq	single-molecule real time sequencing
SNV	single nucleotide variation
TET	ten-eleven translocation
TPR	true positive rate
TSG	tumor suppressor gene
TSS	transcription start site

1. Introduction

Long-read single-molecule sequencing has revolutionized epigenetics by allowing for read-level, haplotyped analyses without fragmentation or bisulfite conversion [1]. Modern machine learning technologies and software architectures are required in order to process, store, and analyze the large amounts of data generated by platforms like Oxford Nanopore Technologies (ONT) or Pacific Biosciences (PacBio) SMRT-seq. The following chapters provide an introduction to the field of epigenetics with a focus on DNA methylation and reviews long-read technologies for DNA methylation profiling as well as associated computational methodology.

1.1. Epigenomics and transcriptional regulation

Epigenetics refers to the field studying molecular mechanisms which layer on top of (from Greek *epi*, meaning in addition) genetic variation, affect transcription, and result in changes to phenotype which cannot be explained by genetics alone [2]. Some of the most well studied epigenetic drivers are DNA methylation, histone modification, and expression of regulatory RNA such as short and long non-coding RNA (miRNA and lncRNA, respectively). These modalities can affect RNA expression, modulation of chromatin accessibility or organization, as well as interaction with DNA binding transcription factors. Analogously to genomics, the totality of ones epigenetic profile is referred to as the epigenome.

The epigenome is generally inherited at mitosis, forming a so-called epigenetic memory which allows for the inheritance of cellular identity [3]. Consequently, the epigenome is highly tissue specific and indicative of cell-type. Unlike the genome, it undergoes massive changes during development [4]. The study of the epigenome is crucial for our understanding of transcriptional mechanics, cell-type identity, development and aging [5]. Epigenetics is also of particular interest in the study of disease, as disruption of epigenetic processes can lead to abnormal expression changes and cellular behavior and even cause genetic point mutations [6]. The "hallmarks of cancer" [7, 8] describe an (ever changing) number of mechanisms important to cancer development, progression, and treatment. As of recent, epigenetics has been included due to its role in tumorigenesis, malignant cell identity, immune modulation and inhibition of DNA repair mechanisms [6], often through promotion of oncogene and disregulation of tumor suppressor gene expression [9]. Being a reversible cellular feature, the epigenome also prospectively represents a potential target for

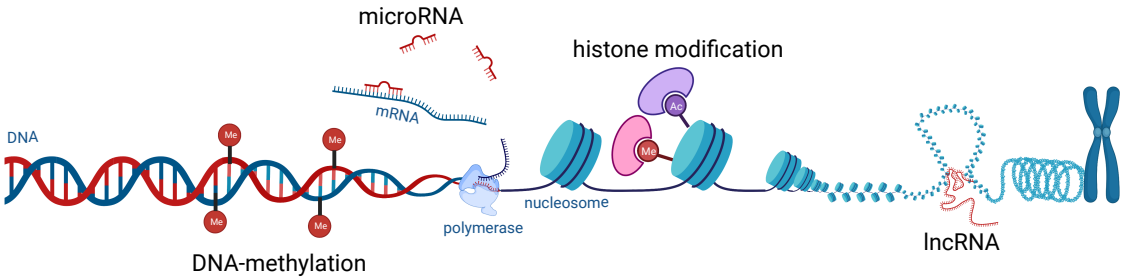

Figure 1.1.: Illustration of different types of epigenetic modifications and drivers. DNA methylation annotates DNA by binding of methyl-groups to nucleobases, affecting the binding of transcription factors which enhance or repress gene expression (Chapter 1.1.1). Histone modifications, here illustrated as acetylation and methylation, modulate chromatin organization and accessibility (Chapter 1.1.2). MicroRNA affects translation by modifying mRNA postranscriptionally. LncRNA is illustrated here in one of its many functions, as an architect of chromatin organization.

therapy, making the identification of epigenetic drug targets of great interest for cancer treatment [10].

1.1.1. DNA methylation

The modification of DNA by way of methylation is an important epigenetic driver found in both prokaryotes and eukaryotes. In general, it describes the molecular process of binding a methyl group (CH_3) to a nucleobase in a DNA sequence. This binding process is mediated by a DNA methyltransferase, an enzyme which selectively interacts with DNA in order to transfer a methyl group from the universal methyl donor *S-adenosylmethionine* to the appropriate DNA base. While DNA methylation in mammals is involved in complex regulatory mechanisms affecting gene expression and alternate splicing [11], DNA methylation was first discovered in bacteria [12], where it mostly serves as a rudimentary immune system [13].

Adenine methylation

Adenine methylation primarily occurs in bacteria, where it turns adenine into N^6-methyladenine (6mA) by binding a methyl group to the 6th atom in the adenine structure. Seeing how bacterial genomes lack histones to be modified or the chromatin structure to be manipulated by lncRNA, DNA methylation represents the main epigenetic mechanism in these organisms. Depending on species, adenine methylation may be mediated by a number of methyltransferases and occurs in different sequence contexts. For example, *DNA adenine methylase* (Dam) is known to bind unmethylated 5'GATC3' (hereon shortened to just GATC) context immedi-

ately after DNA replication in order to copy the methylation state of the template strand to the newly synthesized strand. It is important to note at this point, that DNA methyltransferases tend to be sequence specific and prefer self-complementary sequences [14]. Self-complementary sequences are sequences which read the same when read in the 5'-3' direction on one strand as when read on the 5'-3' direction the other strand. This can be seen in the GATC motif, whose reverse complement sequence is also GATC.

Dam mediated adenine methylation has been found to fulfill a number of different roles. One of the first discovered roles is its part in the restriction-modification system [15], a mechanism in which restriction enzymes cleave foreign DNA to protect the bacteria from viral infections. GATC methylation is used to protect the bacteria's own DNA from restriction enzymes, or in other cases to promote cleavage of methylated DNA in the case of methylation dependent restriction enzymes [16]. Another important function of GATC methylation is the regulation of gene transcription. Genes containing GATC motifs in their promoter may be transcriptionally enhanced or repressed, by invitation of methylation dependent or blockage of methylation adverse activator or repressor enzymes [17]. Furthermore, GATC methylation has also shown to change the curvature of DNA, affecting chromatin structure in a way that regulates gene transcription [18].

Cell cycle regulated methyltransferase (CcrM), another adenine methyltransferase found in bacteria, also binds in a self-complementary sequence, namely the 5'GANTC3' context. As its name suggests, expression of CcrM is regulated by cell cycle. It is limited to the final stage of chromosome replication, such that chromatin is hemimethylated (methylated in one strand only) during most of the replication stage and becomes fully methylated at the beginning of the cell division stage [19, 20]. GANTC methylation therefore represents an important epigenetic marker of cell stage in bacteria, affects transcription of cell-cycle related genes, and appears to be crucial to maintaining the proper chromatin structure required for cell division [19].

Occasionally, novel methyltransferases are discovered in individual strains of lower organisms such as bacteria or viruses. Their function is probably akin to that of an innate immune system and they are therefore not involved in regular cellular function. Nevertheless, these methyltransferases can be of interest for *in vitro* studies, where methylation of certain nucleobases is desired. For example, they may be used to mark chromatin accessible to DNA binding proteins (Chapter 1.1.4). Among these are sequence-independent methyltransferases, which are a special class of methyltransferases which methylate nucleobases in arbitrary sequence context and therefore also create a strand-specific methylome. The first mostly sequence-independent methyltransferase was discovered in a strain of Mu phage virus [21]. The gene expressing Dam has in this strain been replaced with another methyltransferase which methylates adenine in any context other than poly-adenine. EcoGII, the first truly sequence-independent methyltransferase, has been cloned from a strain

of *E.coli*, where it had been encoded but not expressed. It has since been used successfully to methylate adenine in arbitrary sequence context *in vitro* [22].

Cytosine methylation and oxidative derivatives

While more common in mammalian DNA, cytosine methylation has been found both in prokaryotic as well as in eukaryotic genomes. Cytosine is converted to 5-methylcytosine (5mC) or the lesser studied 4-methylcytosine (4mC) by binding a methyl-group to the 5th or 4th atom in the cytosine structure [23, 24]. *DNA cytosine methyltransferase* (Dcm) is found in bacteria, where it methylates the inner cytosine in the self-complementary CCWGG sequence. Cytosine methylation has been shown to play a limited role in the restriction-modification system [25] and has been hypothesized to be regulatory of bacteria growth rate [26]. M.CviPI is a methyltransferase cloned from Chlorella virus, which methylates cytosine specifically in the self-complementary 5'GC3' context [27], typically written as GpC to represent guanine and cytosine connected by a phosphate link in DNA. Like EcoGII with adenine, M.CviPI has been used for *in vitro* methylation of cytosine in GpC context [28].

In mammalian organisms, including humans, cytosine methylation represents one of the most important epigenetic marks. GpC methylation in humans has been detected in mitochondria [29], but the most common and well studied form of cytosine methylation in mammals is the conversion of cytosine in 5'CG3' (CpG) context to 5mC in chromosomal DNA. The functions and methods of regulation of CpG methylation in mammals are far more complex than that of DNA methylation in prokaryotes. CpG methylation levels in mammals are regulated by a complex mechanism involving over ten different enzymes, controlling the targeted methylation and demethylation of CpG sites [11] (Figure 1.2).

There are two types of methyltransferases responsible for CpG methylation. *DNA methyltransferase 3 alpha* and *beta* (DNMT3a and DNMT3b, respectively) are responsible for *de novo* methylation of CpG sites. DNMT3a and DNMT3b bind unmethylated and hemimethylated DNA alike, and are mostly active in pluripotent cells and early embryonic development [30]. Continuous *de novo* methylation is an important mechanism for cell differentiation, setting the tracks for a cell's path down Waddington's landscape [31]. Fully differentiated somatic cell are found to be approximately 70-80% methylated in CpG sites [11, 32]. This gradual differentiation process, however, requires the methylome to be inherited in mitosis, a task performed by maintenance *DNA methyltransferase 1* (DNMT1). DNMT1 specifically binds hemimethylated DNA during the synthesis stage of mitosis, and copies the methylation state of the original template DNA strand to the newly synthesized strand. It is this mechanism that not only inherits methylation state in mitosis, but also ensures symmetric methylation of CpG sites on both strands, as the CpG motif is again a self-complementary sequence.

Unlike eukaryotic cells, mammalian cells also contain mechanisms for the controlled demethylation of CpG sites. Demethylation can occur in two different ways, described as passive demethylation and active demethylation. Passive demethylation describes the dilution of methylation by failure of methylation maintenance through DNMT1. If the methylation state of a CpG site is not copied during mitosis, first generation child cells will be hemimethylated, and two of the four second generation child cells will be unmethylated (Figure 1.2).

Active demethylation is performed through oxidation followed by passive dilution or base repair mechanisms. The *ten-eleven translocation* (TET) enzyme is a DNA binding protein which interacts with 5mC, successively oxidizing the methyl group (CH_3) to a hydroxymethyl group (CH_2OH), then a formyl group (CHO), and ultimately a carboxyl group (CO_2H). The respective oxidative derivatives of 5mC, called 5-hydroxymethylcytosine (5hmC), 5-formylcytosine (5fC), and 5-carboxylcytosine (5caC), have then shown to be more difficult to maintain by DNMT1 during mitosis, leading to passive dilution [33]. Alternatively, *thymine DNA glycosylase* (TDG), a DNA repair enzyme responsible for excising mismatched thymine from G/T pairs, has also been found to react to 5fC and 5caC, excising these nucleobases and leaving behind an abasic nucleotide [34]. These abasic nucleotides are then repaired by base excision repair (BER) mechanisms, replacing the removed nucleobase with a canonical (unmethylated) cytosine base, completing the active demethylation process. Furthermore, there has been evidence of direct decarboxylation of 5caC, indicating another potential pathway of active demethylation [34] (Figure 1.2).

While oxidative derivatives of 5mC are often thought of as mere transient states on the way to demethylation, studies suggest 5hmC, 5fC, and 5caC might also be functional epigenetic marks. This is indicated by the stability of 5hmC in mitosis, enrichment of 5fC in regulatory elements, and the existence of DNA binding proteins which preferentially bind to 5hmC, 5fC, or 5caC [35, 36]. A clear picture of potential regulatory mechanisms by these DNA modifications, however, requires further study.

CpG organization and functions in mammals

Early on it had been discovered that while most of mammalian DNA is methylated, unmethylated regions appeared to be enriched with short sequences of high CpG-density [37]. Furthermore, these unmethylated and CpG-enriched regions were found to be mostly located upstream of actively expressed genes, giving rise to suspicion that CpG-methylation may be potential regulator of gene expression [32]. Later, the term CpG-Island (CGI) was coined [38, 39], which is now typically defined as 200-3,000bps long sequences with a GC-content higher than 50% and normalized CpG fraction greater than 0.6 [40–42], found at most gene promoter regions.

Normalized CpG fraction is typically calculated as:

$$CpG_{Normalized} = \frac{CpG_{Observed}}{CpG_{Expected}}$$

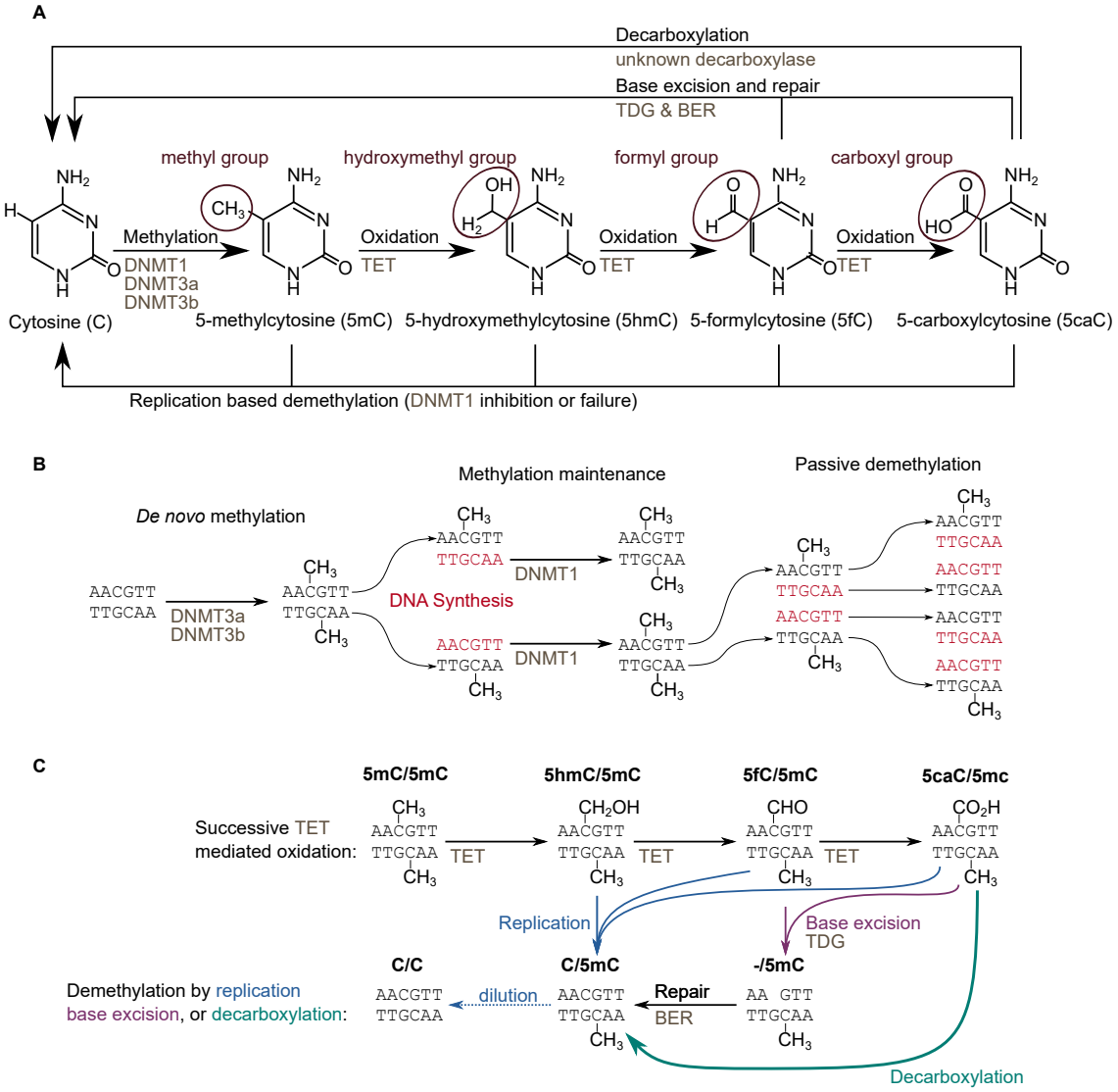

Figure 1.2.: DNA Methylation and demethylation pathways for CpG methylation in mammals. **A)** Chemical composition of cytosine nucleobases, from canonical (unmethylated) cytosine to 5mC, to its oxidative derivatives 5hmC, 5fC, and 5caC. Illustration inspired by [11]. **B)** Methylation and passive demethylation on double stranded DNA. DNMT3a and b *de novo* methylate both strands. At mitosis, new DNA strands are synthesized and the methylation state of the original strand is copied by the maintenance methylase DNMT1. Passive demethylation occurs when DNMT1 does not perform and the methylation state is not copied at mitosis. In this case, each generation of cell division dilutes the methylation rate further. **C)** Active demethylation pathways. TET successively oxidates methyl group, transforming it into a hydroxymethyl-, formyl- and ultimately carboxyl group. All three can be demethlyated passively. The latter two may have the entire modified nucleobase excised by TDG. Repair mechanisms (BER) then fill the missing base with a canonical cytosine. Hemimodified DNA will eventually be completely demethylated through replication based dilution.

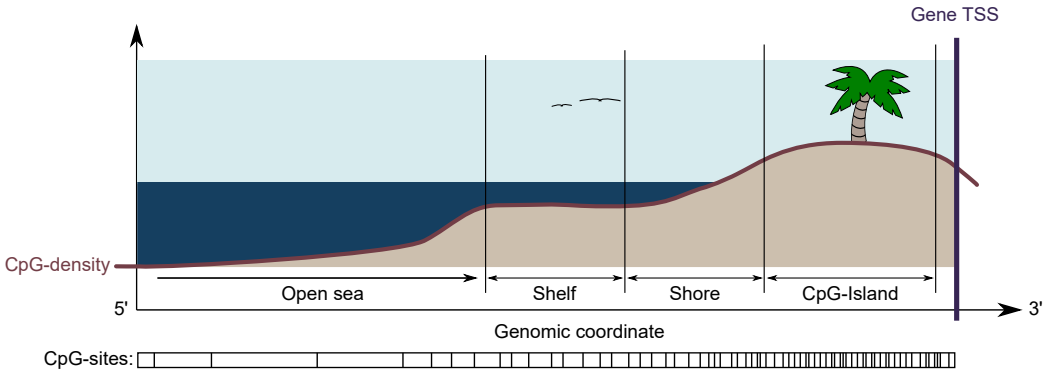

Figure 1.3.: Sketch representing promoter CpG islands, shores and shelves in the promoter region upstream of a gene's transcription start site. CGIs are short regions with high CpG density [38, 39]. Shores are typically defined as the 2kbp region upstream of the CGI, and shelves describe the region of intermdiate CpG density 2kbps further upstream. CpGs in low-density regions outside of CGIs, shores, or shelves are called open-sea CpGs [41, 43]. Illustration inspired by [44].

where

$$CpG_{Expected} = \frac{N_{\mathrm{C}} N_{\mathrm{G}}}{N_{\mathrm{N}}}$$
$$CpG_{Observed} = N_{\mathrm{CG}}$$

and N_b for $b \in \{\mathrm{C}, \mathrm{G}, \mathrm{N}, \mathrm{CG}\}$ is the number of sequence b observed and N is the total sequence length.

Further studies have shown that while full CGI methylation is an effective suppressor of transcription, the lower-density regions upstream of CGIs appear to have more variation and correlate stronger with gene transcription than the actual CGI itself. This has been observed both in disease as well as in healthy cell differentiation. These regions 2,000bps from CGIs have since been named CGI shores [41]. Keeping with the metaphor, the 2,000bps beyond CGI shores are named CGI shelves, and CpG-sites not within islands, shores, or shelves are commonly referred to as open-sea CpGs [43] (Figure 1.3).

Functions of CpG methylation in mammals

The most well known and well studied function of CpG methylation in mammals is the repression of gene expression through promoter methylation. Upregulation of genes can then occur by active promoter demethylation through the recruitment

of TET and TDG enzymes [45]. This function is particularly apparent in the case of X chromosome inactivation in early female embryonic development [46]. Female organisms contain two copies of the X chromosome, and require that one of the two copies is completely silenced from transcription, to avoid a 2-fold overexpression of X chromosome genes. Therefore, one of the two copies undergoes global *de novo* methylation where the majority of CGIs, both in gene promoter regions but also outside of gene promoter regions, become hypermethylated.

Transcriptional silencing via methylation, however, has been observed in autosomes as well [47], where it is highly cell-type specific [48]. The exact mechanism by which promoter methylation silences gene expression varies. It is typically ascribed to one of two mechanisms: (i) by blocking the binding of methylation adverse transcription factors, or (ii) by inviting the binding of repressor enzymes which in turn activate a pathway of transcriptional silencing via chromatin remodeling.

Supporting the first model, a number of transcription factors required for gene expression which specifically bind GC-rich motifs have been identified. These require cytosine to be canonical in order to successfully bind [49]. Methylation of genes targeted by these transcription factors can therefore inhibit transcription factor binding and suppress gene expression [50]. Since this model can only apply to a limit number of genes with GC rich promoters, it is therefore assumed that the more significant mechanism is silencing via invitation of a family of proteins known methyl CpG binding domains (MBDs). Several of these proteins have been identified, like *methyl-CpG binding protein complex 1* and *2* (MeCP1, MeCP2) as well as MBD1 through MBD4. MBDs binding in methylated CGIs have been found to further invite structural proteins and histone deacetylase, modifying nearby histones leading to compact and inaccessible chromatin (Chapter 1.1.2). This chromatin inaccessibility then leads to inability of polymerase and transcription factor binding, therefore silencing gene expression [51].

While gene silencing via promoter methylation is the most well known regulatory mechanism of CpG methylation, more recently it has been found that methylation can also have the opposite effect on transcription. A number of transcription factors have been found which are methylation dependent and cannot bind promoters with unmethylated CGIs [52]. In this case, transcription can be upregulated through promoter methylation [53, 54].

Alternative splicing regulation is another function of CpG methylation in mammals which is not yet entirely understood and likely a complex interplay of different mechanisms. To understand this, the we first gene transcription as a three-stage process: (i) transcription intiation, (ii) elongation, and (iii) termination. The result is an unspliced pre-mRNA molecule, which then undergoes selective splicing out of introns, retaining only the short exons in the final mRNA molecule. While upregulation and downregulation of gene expression via promoter methylation shows the effect DNA methylation has on the initiation stage of transcription, it has been shown that DNA methylation can also effect the elongation stage resulting in alternative

exon inclusion.

Exons are distinguished between constitutive exons and alternative exons. Constitutive exons are typically included in mRNA, while alternative exons, also called "weak" exons, are often spliced out together with introns. While splicing occurs after transcription, it is already during the elongation stage of the transcription process when exons are prepared for inclusion, and it has been shown that a slower elongation phase is more likely to detect a weak exon boundary and therefore lead to inclusion of an alternative exon [55]. The discovery that exons typically contain slightly higher (17%) methylation rates than neighboring introns [56], and included exons tend to have higher methylation rates that excluded exons [57] was the first indication that exon methylation may be regulatory for exon inclusion. A number of mechanism have since been proposed to explain how DNA methylation may regulate alternate splicing.

Unmethylated regions near alternative exons are prone to be bound by conserved zinc finger protein (CTCF), an important transcription factor. When CTCF binds downstream from an alternative exon, the polymerase may be blocked and the elongation phase of transcription may be slowed down [58]. Similarly, however, methylated DNA might invite MBDs such as MeCP2 to bind near the exon, having a similar effect as CTCF, also slowing the elongation phase. In either case, the extended elongation phase can then lead to improved detection of the alternative exon's weak exon boundary and inclusion of the exon in the final spliced mRNA products [56].

Aberrant methylation in disease

Aberrations in DNA methylation are an important clinical factor in the understanding, identification, and treatment of a number of diseases. The methylome undergoes large scale changes not only in the early embryonic development, but also during aging, and age releated hypomethylation has been linked to a number of diseases, from autoimmune disease [59] to neurological disorders [60].

DNA methylation aberrations can be caused by disease or can be causal for disease development themselves. Escape from X chromosome inactivation, for example, can be causal for a number of X-linked diseases such as autoimmune disorders, leading to an increased prevalence in female organisms [61–63]. On the other hand, mutation accumulation in cancer can cause alterations to methylation relevant genes such as DNMT or TET, which interrupt the regular processes of DNA methylation maintenance and ultimately lead to changes in methylation patterns [64, 65]. These methylation changes may then further lead to improved tumor survival and immune escape, as a result of which DNA methylation has been named one of the hallmarks of cancer [7]

While DNA methylation changes in cancer are tumor specific, a widespread theme observed in many cancer types is described as global hypomethylation, primarily observed in repetitive sequences, combined with focal hypermethylation [66–68].

Hypomethylation has been shown to contribute to cancer development by transcriptional activation of so called "oncogenes", genes that when expressed contribute to uncontrolled cell proliferation and thus cancer growth [69]. Another way in which global hypomethylation might contribute to cancer development is by causing genomic instability and thereby leading to progressively increased genomic rearrangements [70, 71]. While mechanics of regulation via promoter or enhancers methylation in CGIs and shores are relatively well understood, hypomethylation related to genomic instability has been shown to be relevant in shelves and open-sea CpGs as well, indicating that epigenomic analysis should not be restricted to CGIs and shores alone [72].

Focal hypermethylation results in gene promoter methylation leading to downregulation, or in rare cases upregulation, of gene expression. Tumor suppressor genes (TSGs), also called anti-oncogenes, are genes encoding proteins which act as a regulator of cell proliferation, typically by controlling rate of cell division or by inducing apoptosis in carcinogenic cells [69, 73]. Inactivation of TSG via hypermethylation of TSG promoters can therefore lead to runaway cell proliferation and thus cancer development.

However, genome-wide DNA methylation analysis in a large number of tumors revealed that not only promoters are focally methylated in cancer. Like promoter methylation, the methylation of cis-regulatory elements such as gene enhancers and super-enhancers is found to have an equal - perhaps even stronger - silencing effect on gene expression than promoter methylation [74]. These studies suggest that enhancers make up the most differentially methylated regions between cells of different cancer stages, implying that enhancer methylation plays a role in the development and plasticity of cancer cells required for cancer cell survival and treatment evasion [75, 76].

Both hypomethylation and focal hypermethylation have been shown to occur in an allele-specific manner [77], meaning that they may not be applied symmetrically to both chromosomal copies. While allele-specific methylation (ASM), where one allele is differentially methylated from the other at the same genomic locus, is not uncommon in healthy cells [78], it has been shown that ASM is indeed increased in cancers [77, 79]. This makes the analysis of allele-specific methylation of particular interest in cancer studies.

DNA Methylation as treatment target or biomarker

Motivated by the prominent role of DNA methylation in cancer development and survival, it has also long been of interest as a potential treatment target [80]. For example, reactivation of TSG through targeted hypomethylation of promoters and enhancers has been accomplished using the TET mediated active demethylation pathway [81, 82].

Aside from treating DNA methylation targets causal to disease progression, disease-

specific methylation patterns may also be used as biomarkers for the identification of disease subtypes and progression. DNA Methylation profiles have been used in clinics for diagnosis, triage and choice of therapy [66, 83]. In cancer, DNA methylation profiles have been used to classify cancer subtypes [84, 85], help infer clonal evolution [86], or to determine the tissue or organ of origin for metastatic cancer [87, 88]. In some cases, cancer biomarkers may also be detected in cell-free DNA collected from liquid biopsies (e.g. blood), making such DNA methylation biomarkers particularly accessible [83, 89].

1.1.2. Chromatin remodeling

Chromatin is a tightly packaged accumulation of nucleosomes, each nucleosome consisting of a histone octamer with DNA wound around it. The histone octamer itself is a combination of eight histone proteins, two copies of each canonical histone H2A, H2B, H3 and H4. This tight packaging allows for DNA to be compactly stored in a cell's nucleus and also serves to protect DNA from damage [90]. Heterochromatin describes chromatin which is so thightly packed, that it is entirely inaccessible to DNA binding proteins, such as polymerase, repair proteins, transcription factors, MBDs, or DNA methyltransferases, and is therefore not actively transcribed. Euchromatin is more loosely packed and consists of open regions of accessible DNA between nucleosomes. The majority of eukaryotic chromatin is heterochromatin, however heterochromatin can be further divided into faculive and constitutive heterochromatin. Consitutive heterochromatin typically remains inaccessible and does not change its composition, wheras facultative heterochromatin can be remodeled to euchromatin during development and cell differentiation [91].

Like DNA bases, euchromatin histones are regularly modified in order to regulate chromatin accessibility and affect transcription. Histones are prone to a large number of modifications, including methylation, acetylation, phosphorylation, deamination, and others [92], resulting in a large number of variant histones. Histone acetylation results in weaker nucleosome-DNA binding, allowing DNA to unspool and become accessible for transcription. Gene expression can therefore be regulated by the expression of histone acetylase and deacetylase proteins [92], leading to higher or lower accessibility for transcription factor binding. Histones can be methylated in a number of positions, and there are numerous methylation derived modified histones. Some are associated with upregulation and some with downregulation of gene expression [93].

Aside from chromatin accessibility, the 3-dimensional organization of chromatin can also be important for transcription. Cis-regulatory elements such as enhancers can be located megabases away from gene promoters, yet their accessibility may be required for transcription initiation (Figure 1.4). The explanation for this is that chromatin may be looped such that enhancer and promoter are in physical proximity and can be co-bound by transcription factors [94, 95]. This 3d-structure can in

turn be modulated by other epigenetic factors, including lncRNA [96], which bind chromatin and may pull regions of DNA together in order to promote transcription factor co-binding.

1.1.3. State of the art technologies for epigenetics profiling

Seeing how DNA methylation plays such an important role in our understanding of human development, cell identity and differentiation, as well as global diseases such as cancer, the measurement of DNA methylation in biological samples is of great interest. Early methods were limited to the overall quantification of methylated material in a sample, typically by labeling of methylated DNA with radioactive isotopes [97] or more recently MBDs whose binding could be visualized on an optical plate reader [98]. Epigenetic programs in mammals, however, are highly localized, and such methods were not sufficient to aid our understanding of the mammalian methylome. The solution to this came with advances in genotyping and sequencing methods, which could be repurposed for the methylation measurement either through base modification (bisulfite sequencing, enzymatic methylation sequencing) or direct measurement (ONT, PacBio).

Bisulfite based methods

Bisulfite based methods utilize sodium bisulfite ($NaHSO_3$) to treat methylated DNA. Sodium bisulfite reacts with canonical cytosine in a three-step process, leading to a deamination of cytosine, converting it to canonical uracil. In the first step, the bisulfite ion (HSO_3) attaches to the sixth position in cytosine's carbon ring. This triggers deamination via hydrolysis, followed by the bisulfite ion being freed again in order to regenerate the double bond between the fifth and sixth carbon [99] (Figure 1.5).

The key for purposing this method for the sake of DNA methylation quantification, is that this process depends on cytosine being canonical. Modified cytosine (5mC, 5hmC, 5fC, 5caC) do not interact with sodium bisulfite and therefore remain unchanged after bisulfite treatment.

Bisulfite treated DNA can then be amplified using polymerase chain reaction (PCR). In the amplification process, the complementary base for methylated cytosine is guanine. For canonical cytosine, which has been converted to uracil, the complementary base is adenine (since uracil is but unmethylated thymine). In the following steps of amplification, the methylated cytosine location is therefore amplified as C/G bases and the originally canonical cytosine is amplified as A/T bases.

Subsequently, standard genotyping methods can be used to identify whether a C/G location has been replaced by an A/T base. To do so, genotyping arrays designed specifically for the genotyping of cytosines in CpG context on the human genome were developed. Two well established examples are the Illumina Infinium Human-

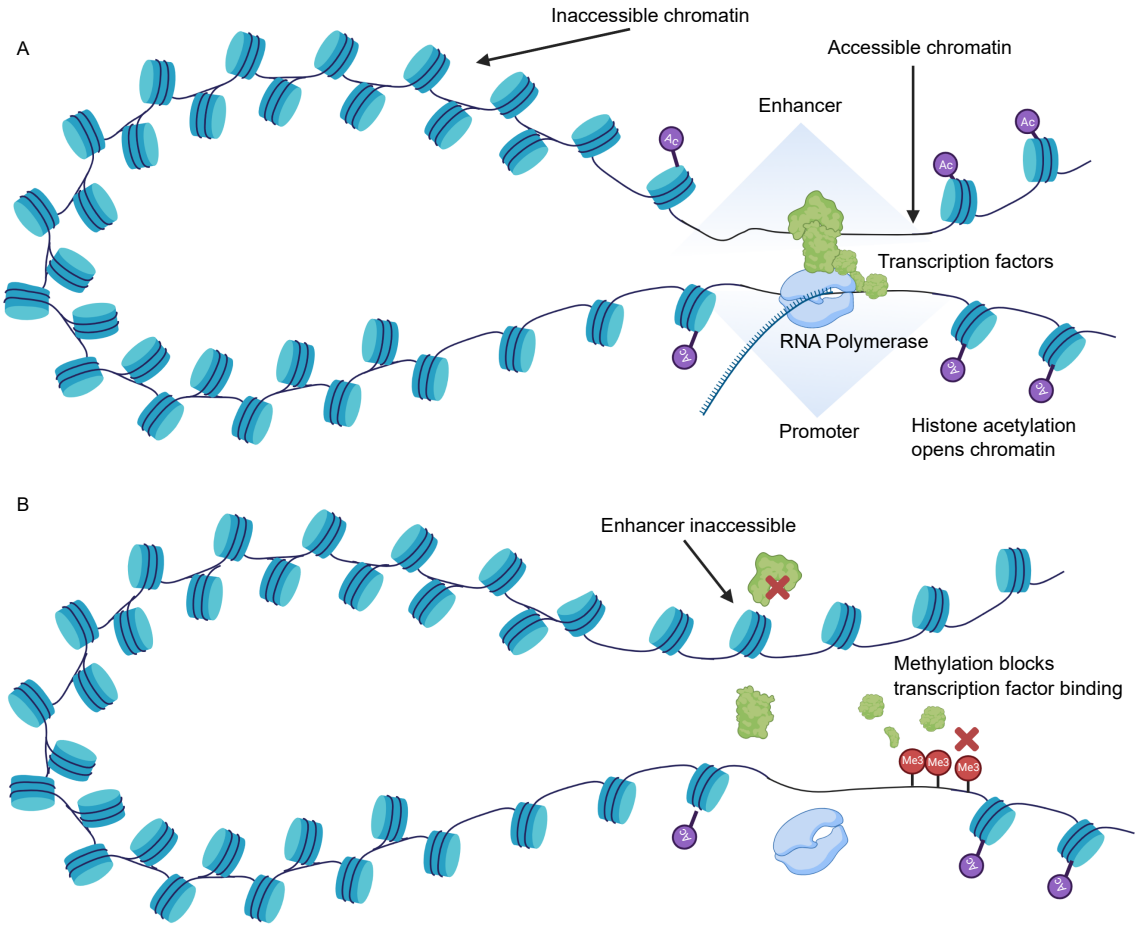

Figure 1.4.: Schematic representation of transcription with co-binding of enhancer and promoter. **A)** Chromatin is made accessible through histone acetylation, allowing unspooling of regulatory DNA, in enhancer and promoter region, enabling co-binding of transcription factors and RNA polymerase thus increasing the chance of transcription. **B)** Examples of epigenetic downregulation. Enhancer region is not made accessible due to deacetylation of nearby histones. Promoter region is methylated and transcription factors fail to bind.

Methylation450 (in short often referred to as 450k-array) BeadChip, which covers over 480,000 CpG sites and includes the majority of CGIs in the human genome, and the Infinium MethylationEPIC (often referred to as the 850k-array), covering over 850,000 CpG sites and offering improved coverage of gene regulatory sites [100].

Advances in next generation sequencing later allowed for high throughput sequencing, using the sequencing-by-synthesis principle [101]. Whole genome sequencing (WGS) experiments can read an entire genome for the purpose of genome assembly or genotyping via mapping to an established reference genome. Whole genome bisulfite sequencing (WGBS) refers to the combination of bisulfite treatment and WGS, allowing for genome-wide base-pair resolution methylation measurement.

Bisulfite conversion based methods come with a number of limitations. Conversion rate depends on the dosage of sodium bisulfite and incubation time, and low conversion rate leads to an overestimation of DNA methylation. A more aggressive treatment with high dosage and sufficient incubation time, however, leads to DNA degradation resulting in fragmentation of DNA molecules [102]. While fragmentation is a necessary part for sequencing-by-synthesis, it can be problematic if sequencing adapters are damaged. This lead to the development of the post-bisulfite-adapter-treatment (PBAT) protocol, in which sequencing adapters are ligated after the bisulfite treatment [103]. PBAT now represents the gold-standard protocol for WBGS.

A more recent conversion-based approach promises to reduce DNA degradation by replacing bisulfite treatment with an enzymatic reaction based on first oxidizing 5mC to 5hmC using TET and then deaminating canonical cytosine using *apolipoprotein B mRNA editing enzyme catalytic polypeptide-like 3A* (APOBEC3A), which reacts with canonical and methylated cytosine, but not its oxidative derivatives [104].

Fragment size and thus read length are inherently limited in a WGBS experiment, making this approach incompatible with modern long-read sequencing technologies. Furthermore, it is difficult to distinguish between canonical cytosine, which is read as A/T, and actual C>T single nucleotide variation (SNV), be it a point mutation or single nucleotide polymorphism (SNP). The large number of induced cytosine variations on top of regular genomic variation makes the alignment of such short reads particularly challenging, leading to the development of specialized alignment algorithms [106]. Even more challenging is the assessment of ASM using WGBS. Assigning methylation values to a specific allele requires the presence of heterozygous SNVs other than C>T, which may not be given on short reads, as well as the ability to reliably detect them. These and other factors contribute to the interest in longer read lengths and bisulfite-free technologies for DNA methylation detection such as the methods.

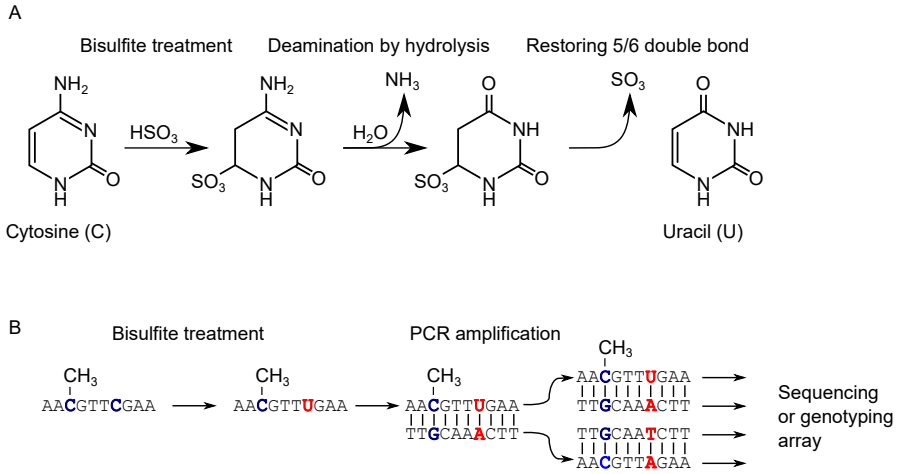

Figure 1.5.: **A)** Three-step process for deamination of cytosin to uracil via bislufite treatment and hydrolysis. Illustration inspired by [105]. **B)** Bisulfite conversion followed by genotyping for base-resolution DNA methylation measurement. Methylated cytosine is protected from bisulfite conversion, whereas canonical cytosin will be read as thymine.

Direct measurement of modified bases

Read length in sequencing experiments is a key determining factor for the quality of reference assembly and alignment, and of particular importance when studying structural variations. Traditional sequencing-by-synthesis platforms rely on simultaneous synthesis of multiple strands which must perform in perfect synchronicity in order to measure the synthesized base. With increased read length, synthesis lag accumulates and sequencing quality drops, resulting in maximum read lengths of about 300bps in current Illumina sequencing systems [107, 108].

Novel sequencing technologies were developed to overcome the read length limitations, two of which are Nanopore sequencing, implemented by ONT, and single-molecule real time (SMRT) sequencing by PacBio.

Nanopore sequencing measures perturbation in electrical current as DNA or RNA molecules flow through a Nanopore, producing a distinct electrical current read for each molecule. The current perturbation can then be analyzed in order to reconstruct the original sequence by solving the inverse problem of signal perturbation. This is a non-trivial problem, which requires computationally expensive machine learning algorithms detailed in Chapter 1.3.1 and Chapter 1.3.2. Nanopore sequencing suffers from lower base accuracy than sequencing-by-synthesis approaches. As a comparison, standard quality control for ONT sequencing data suggests discarding reads with a mean Q-score of 7, that is keeping reads with 20% error rate or smaller. In comparison, short read sequencing experiments quality control standards typically recommend discarding reads with a Q-score of less than 30, that is a maximum read error rate of 0.1% [109].

However, Nanopore sequencing allows for reads of theoretically unlimited read length, with the current record being a 2Mbps read [110]. It also allows for direct measurement of chemical base modifications like 5mC or 6mA and even oxidative derivatives like 5hmC, by distinguishing the signal perturbation of modified from that of canonical bases [111]. Computational tools for base- and methylation calling are being continuously developed, and previously generated Nanopore data can be be re-analyzed with new software versions in order to improve data quality.

SMRT-seq is a modified synthesis-based sequencing approach. Instead of synthesizing a multitude of molecules concurrently, SMRT-seq synthesizes a single circularized molecule multiple times in sequence, while measuring fluorescence labeled synthesized bases [112]. While fluorescence values do not reveal information about DNA methylation state, it was found that temporal information such as duration of fluorescent pulse and delay between pulses can be used to determine base modification state, and a number of software tools have been developed and successfully applied to determine 5mC and 6mA methylation [112, 113].

Both Nanopore sequencing and SMRT-seq allow for simultaneous interrogation of genome and methylome on the same molecule. Long reads allow for better mapping haplotype assignment for the analysis of ASM, better mapping to repetitive or highly rearranged regions, and therefore enable genome-wide haplotype phased DNA methylation analyses.

1.1.4. Chromatin accessibility profiling

Chromatin accessibility profiling is of great interest for the discovery of gene regulatory regions and their functions. Technologies have evolved from indirect measurement, to direct measurement in bulk and single cells, and eventually to direct single-molecule profiling [114]. Indirect measurement of chromatin accessibility has been performed using chromatin immunoprecipitation combined with sequencing (ChIP-seq) [115]. With ChIP-seq, DNA binding proteins like histones and transcription factors can be labeled with specific antibodies. DNA is then fragmented, and bound DNA is isolated from the rest, which can then be sequenced and mapped to a reference genome in order to determine which regions in the genome were bound by the targeted protein. In order to estimate chromatin accessibility, specific antibodies for modified histones are used to measure the histone modification state. Chromatin accessibility can then be estimated from neighboring histone methlyation or acetylation state [115, 116].

Cleavage based methods

Direct measurement of chromatin accessibility has been achieved by instead applying DNA binding proteins which bind accessible chromatin in order to label it. Cleavage based methods, such as MNase-seq [117, 118], DNase-seq [119], and ATAC-seq [120]

introduce enzymes which cleave DNA in accessible regions. Cleaved DNA is then sequenced and mapped to a reference genome, after which cleavage events can be analyzed in order to identify regions of accessible chromatin. Since cleavage enzymes also occasionally bind linker DNA between nucleosomes, it is necessary to distinguish between regions of open chromatin and background noise from other sources of cleavage/fragmentation. This task is called "peak calling" and implemented as a machine learning task [121, 122].

Methylation based methods

Another direct method of chromatin profiling is to label accessible chromatin using methyltransferases in order to introduce a type of DNA methylation not originally found in the organism. Methylation profiling is then performed to determine which regions were accessible to the methyltransferase.

NOMe-seq describes a protocol where GpC methyltransferase M.CviPI is used to treat DNA, followed by WGBS [28, 123] in order to simultaneously profile DNA methylation and accessibility (Figure 1.6). Since mammalian DNA generally does not contain cytosine methylation in GpC context outside the mitochondria, methylated GpC sites can be assumed to have been in open chromatin. Since M.CviPI only methylates GpC sites, other cytosine methylation can be assumed to be endogenous. Sites in a GpCpG context are ambiguous, and are typically dropped in interpretation, as methylation of the center cytosine might be endogenous CpG methylation or *in vitro* GpC methylation. While this method allows for accurate measurement of chromatin accessibility in GpC sites, it is limited by the occurrence of the GpC motif and regulatory regions depleted in the GpC motif cannot be profiled.

Another advantage of this method is that chromatin accessibility can be measured for an entire DNA molecule and is limited only by the read length of the sequencing method. If reads are large enough to cover an entire transcription factor binding site, it is possible to detect the footprint left by transcription factor binding, observed as an accessible region with a short inaccessible segment in the middle where the transcription factor is actively binding [95, 124].

Furthermore, the method can be translated to long-read sequencing systems such as ONT sequencing, in order to profile chromatin accessibility in long reads [125] and different methyltransferases can be used to profile regions depleted in the GpC motif [126].

1.2. Principles of machine learning in epigenetics

Bioinformaticians are faced with many challenges which require machine learning tools for the analysis of epigenetic data. In this chapter I give a general introduction into machine learning, starting with principals of Bayesian inference leading up

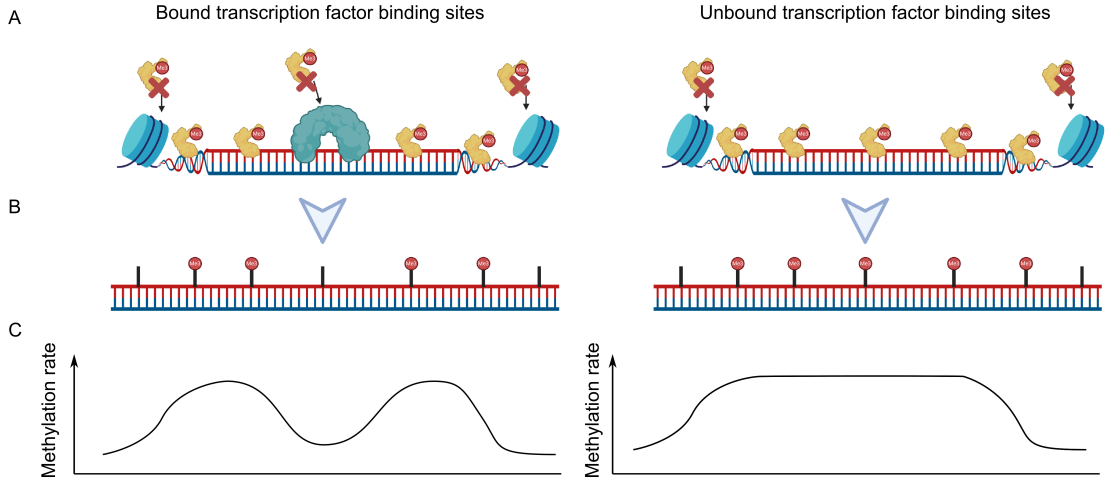

Figure 1.6.: Methylation based chromatin accessibility measurement such as NOMe-seq. Methyltransferases bind accessible chromatin in order to methylate DNA. **A)** Nucleosome occupied chromatin and transcription factor bound chromatin are inaccessible to methyltransferases. **B)** Resulting DNA is methylated in the unbound transcription factor binding site, and in the flanking regions of a bound transcription factor. **C)** Measuring DNA methylation rates over multiple transcription factor binding sites results in footprints such as sketched here, for used an unused sites.

to hidden markov models, which are a corner stone of existing methods for DNA methylation analysis in short and long reads. Finally, I briefly go over deep learning technologies applied for base and methylation calling of Nanopore reads.

Machine learning is a field in the broad area of artificial and computational intelligence. It involves the study and application of algorithms designed to learn functions without them being explicitly programmed [127]. Common problems of machine learning include regression, classification, and clustering. Methods can range from being strictly modeled and probabilistic such as Bayesian inference, to more free-form pragmatic, perhaps even heuristic, approaches such as neural networks for pattern recognition. Some machine learning problems can be solved by a human experts, but intelligent systems are designed in order to automate the task such that it can be applied to large datasets even in the presence of natural variability and signal noise. Examples of this include image classification or segmentation tasks, audio transcription, or machine translation. Other machine learning problems are designed to solve problems too complex for human experts to solve, such as when learning inference on highly dimensional data or learning complex functions not easily intuited.

Machine learning tasks are further classified as supervised and unsupervised methods [128]. Supervised methods require training data with ground truth labels, and typically include classification and regression tasks. Unsupervised methods do not

require ground truth labels and include such tasks as clustering or dimensionality reduction. Weakly or semi-supervised methods describe methods that are trained with a minimal training dataset.

1.2.1. Bayesian methods

Bayesian methods of machine learning describe a family of methods based on the Bayes theorem

$$P(L|X) = \frac{P(X|L)P(L)}{P(X)} \tag{1.1}$$

$$\text{posterior} = \frac{\text{likelihood} \cdot \text{prior}}{\text{evidence}} \tag{1.2}$$

In Bayesian modeling, L typically refers to the parameters of a model, and X the observations (e.g. measurements), in which case $P(L|X)$ is the posterior probability of model parameters L being correct given observations X, $P(X|L)$ the likelihood of observing X given a model with parameters L. $P(L)$ is the prior probability of model parameters L being correct, and $P(X)$ the marginal probability of observing X independent of model parameters, often referred to as the evidence.

Assumptions have to be made for the prior distribution $P(L)$, and the likelihood term $P(X|L)$ must be explicitly defined. The Bayes theorem is then used to infer parameters L which best describe observations X using approximate solutions. Methods for estimation of L include variational inference [129] or Markov chain Monte Carlo [130]. The maximum *a posteriori* (MAP) estimator is defined as

$$L_{MAP} = \max_{L} \frac{P(X|L)P(L)}{P(X)}$$

Since the evidence term $P(X)$ is independent of L the MAP estimator can be simplified to:

$$L_{MAP} = \max_{L} P(X|L)P(L)$$

The choice of a prior distribution $P(L)$ is crucial, as it represents the prior belief about parameters. A strongly informative prior has high certainty, high bias, and a strong effect on parameter estimation. A weakly informative prior has high uncertainty, low bias, and observations X will have a stronger impact on parameter estimation. The parameters defining the uncertainty of the prior distribution are called hyperparameters [131].

A Bayesian network describes a graphical model of multiple connected probability distributions in the form of a directed acyclic graph, where the posterior probability of one distribution may describe the prior of another distribution.

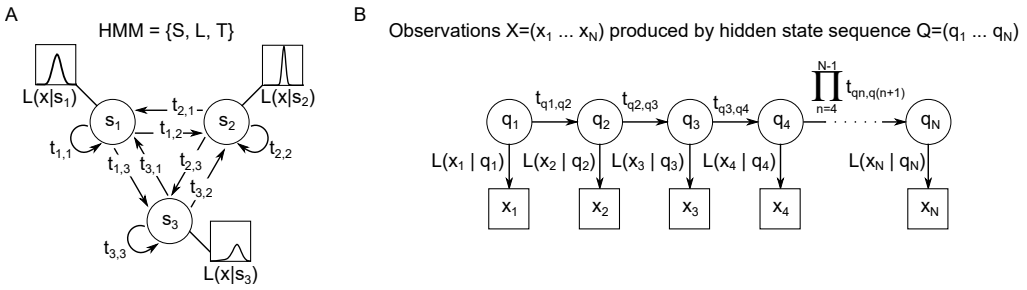

Figure 1.7.: Illustration of a Hidden Markov Model (HMM) **A)** An HMM is defined by a set of states S, a set of transition probabilities between states T, and a likelihood function L, defining distributions of state-dependent emissions. **B)** Example of a sequence of emissions $X = (x_n)$ emitted by a sequence of states $Q = (q_n)$.

Hidden Markov Models

A Markov chain [132] describes a sequential process defined over a set of states S and transition probabilities between states T. An n^{th} order Markov chain describes the transition probability to the next state based on the previous n states, and thus has the form $T = \{t_{i,j} : i \in S^n, j \in S\}$, where S^n is the n^{th} cross product of the set S. Let $t_{0,j}$ be the initial probability of starting in state j. For example, if a Markov chain represents a genomic sequence, states could be the genomic bases `T,C,G,A` and transition probability $t_{\texttt{C,A}}$ would be the probability of a adenine following an cytosine base in a first order Markov chain.

Let $Q = (q_n)$ be a series of states where $q_n \in S$, then a first order Markov chain allows the prediction of the most likely next state q_{n+1} by maximizing $t_{q_n,q_{n+1}}$.

$$P(q_{n+1}|q_n) = t_{q_n,q_{n+1}}$$
$$\hat{q}_{n+1} = \arg\max_{q_{n+1}} t_{q_n,q_{n+1}}$$

A Hidden Markov Model (HMM) [133, 134] is an extension of the first order Markov chain, in which the state sequence Q is assumed to be unobservable (hidden). Instead, one a sequence of observations $X = (x_n), 1 \leq n \leq N$, also called emissions, is observed, and the emission distributions $P(x_n|q_n = p)$ are modeled as part of the HMM'S definition (Figure 1.7). The emission distributions are also referred to as the emission likelihood functions $L(x_n|p)$.

HMMs are associated with three canonical problems, each associated with a canonical algorithm. Assuming that for each problem, a series observations X and a set of states S is given:

1. **Likelihood estimation**: Given transition probabilities T and likelihood function L, compute the marginal likelihood $P(X)$. The canonical algorithms for

this problem are the **forward and backward algorithms** [135].

2. **Decoding**: Given transition probabilities T, and likelihood function L, predict the most likely sequence of hidden states $\hat{Q} = (\hat{q}_n)$. The canonical algorithm for this problem is the **Viterbi algorithm** [136].
3. **Learning**: Learn both the transition probabilities T as well as the likelihood function L. The canonical algorithm for this problem is the **Baum-Welch algorithm** [137]. Seeing how the algorithm depends on intermediate results computed from both the forward and the backward algorithm, the Baum-Welch algorithm is sometimes also referred to as the **forward-backward algorithm**.

The **forward and backward algorithms** are two algorithms which both compute $P(X)$ for a given HMM $\lambda = \{S, T, L\}$ by iteratively computing $P(X, Q)$ while marginalizing out Q. Here, the the forward algorithm moves forward through timepoints $1 < n < N$ and the backward algorithm backwards. Let X_n be the series of observations up to and including timepoint n, then the marginal likelihood $P(X)$ can be expanded to

$$P(X) = \sum_{p \in S} P(X_N, q_N = p)$$

The quantity in the sum can be generalized as the forward path probability $f_{n,p} = P(X_n, q_n = p)$ which can be derived as a recursive function:

$$\begin{aligned} f_{n,p} &= P(X_n, q_n = p) \\ &= \sum_{p' \in S} P(X_{n-1}, x_n, q_n = p, q_{n-1} = p') \end{aligned}$$

and since $P(x_n \perp\!\!\!\perp q_{n-1} = p', X_n | q_n = p)$ and $P(q_n = p \perp\!\!\!\perp X_{n-1} | q_{n-1} = p')$ (conditional independence):

$$\begin{aligned} &= \sum_{p' \in S} P(X_{n-1}, q_{n-1} = p') P(q_n = p | q_{n-1} = p') P(x_n | q_n = p) \\ &= \sum_{p' \in S} f_{n-1,p'} t_{p',p} L(x_n | p) \end{aligned}$$

The forward path probability can be represented as a matrix $F = (f_{n,p})$ and is iteratively computed using a dynamic programming algorithm consisting of:

$$\begin{aligned} \text{Initialization:} \quad & f_{0,p} := t_{0,p} L(x_1 | p) \\ \text{Update:} \quad & f_{n+1,p} := \sum_{p' \in S} f_{n,p'} t_{p,p'} L(x_{n+1} | p) \\ \text{Termination:} \quad & P(X) = \sum_{p \in S} f_{N,p} \end{aligned} \tag{1.3}$$

Alternatively, for the backward algorithm let Y_n be the series of observations after timepoint n, so $Y_n = (x_{n'}, n < n' \leq N)$. Then, $P(X)$ can be expanded as:

$$\begin{aligned} P(X) &= \sum_{p \in S} P(Y_0 | q_1 = p) P(q_1 = p) \\ &= \sum_{p \in S} P(Y_0 | q_1 = p) t_{0,p} \end{aligned}$$

the backward path probability $b_{n,p} = P(Y_n | q_n = p)$ can then be derived as another recurisve function:

$$\begin{aligned} b_{n,p} &= P(Y_n | q_n = p) \\ &= \sum_{p' \in S} P(Y_{n+1}, x_{n+1} | q_n = p, q_{n+1} = p') P(q_{n+1} = p' | q_n = p) \\ &= \sum_{p' \in S} P(Y_{n+1} | x_{n+1}, q_{n+1} = p') P(x_{n+1} | q_n = p, q_{n+1} = p') P(q_{n+1} = p' | q_n = p) \end{aligned}$$

and since $P(x_{n+1} \perp\!\!\!\perp q_{n-1} = p', X_n | q_n = p)$ and $P(q_n = p \perp\!\!\!\perp X_{n-1} | q_{n-1} = p')$:

$$\begin{aligned} &= \sum_{p' \in S} P(Y_{n+1} | q_{n+1} = p') P(x_{n+1} | q_{n+1} = p) P(q_{n+1} = p' | q_n = p) \\ &= \sum_{p' \in S} b_{n+1,p'} L(x_{n+1} | p) t_(p' | = p) \end{aligned}$$

With the backward matrix $B = (b_{n,p})$ iteratively computed as:

$$\begin{aligned} \text{Initialization:} \quad b_{N,p} &:= 1 \\ \text{Update:} \quad b_{n-1,p} &:= \sum_{p' \in S} b_{n,p'} t_{p',p} L(x_n | p) \\ \text{Termination:} \quad P(X) &= \sum_{p \in S} b_{1,p} t_{0,p} L(x_0 | p) \end{aligned}$$

The **Viterbi algorithm** computes the most likely path $\hat{Q} = \max_Q P(X|Q)$ with a dynamic programming algorithm similar to the forward algorithm. It iteratively computes the matrices $V = (v_{n,p})$ and $B = (b_{n,p})$ where $b_{n,p} \in S$ stores the likelihood of the most likely path up to timepoint n and the matching prior state. Backtracing through B then results in the most likely path $\hat{Q}$. The steps of the Viterbi algorithm are derived analogously to the forward algorithm, replacing the sum with a max and arg max operation:

$$
\begin{aligned}
\text{Initialization:} \quad v_{0,p} &:= z_{0,p} = t_{0,p} L(x_1|p) \\
b_{0,p} &:= 0 \\
\text{Update:} \quad v_{n+1,p} &:= \max_{p' \in S} v_{n,p'} t_{p',p} L(x_n, p) \\
b_{n+1,p} &:= \arg\max_{p' \in S} v_{n,p'} t_{p',p} L(x_n, p) \\
\text{Termination:} \quad \hat{q}_N &= \arg\max_{p \in S} v_{N,p} \\
\hat{q}_{n<N} &= b_{n,q_n}
\end{aligned}
$$

To solve the learning problem, the **Baum-Welch algorithm**, attempts to learn L and T such that $P(X|L,T)$ is maximized. The Baum-Welch algorithm is an expectation-maximization algorithm [138], iteratively approximating a solution for T and L, such that

$$\hat{T}, \hat{L} = \arg\max_{T,L} P(X|T,L)$$

Let $T^{(i)} = (t^{(i)}_{p,p'})$, $L^{(i)}$, $\lambda^{(i)} = \{S, T^{(i)}, L^{(i)}\}$ be the series of updated parameters where i refers to the i^{th} update, then each iteration updates the HMM as:

$$T^{(i+1)} := \mathbb{E}\left[T^{(i+1)}|X, L^{(i)}, T^{(i)}\right] \qquad L^{(i+1)} := \mathbb{E}\left[L^{(i+1)}|X, L^{(i)}, T^{(i)}\right]$$

Updates for transition probabilities T are then derived as:

$$t^{(i+1)}_{p,p'} = \sum_n^N P(q_{n+1} = p'|q_n = p, X, \lambda^{(i)}) = \sum_n^N \frac{P(q_{n+1} = p', q_n = p|X, \lambda^{(i)})}{P(q_n = p|X, \lambda^{(i)})}$$

The numerator and denominator can then be computed from the intermediate results of the forward and backward algorithms. Using $F^{(i)}$ and $B^{(i)}$ computed from $\lambda^{(i)}$, the denominator is derived as:

$$
\begin{aligned}
P(q_n = p|X, \lambda^{(i)}) &= \frac{P(q_n = p, X|\lambda^{(i)})}{P(X|\lambda^{(i)})} \\
&= \frac{P(X_n, q_n = p|\lambda^{(i)}) P(Y_n|q_n = p, \lambda^{(i)})}{P(X|\lambda^{(i)})} \\
&= \frac{f^{(i)}_{n,p} b^{(i)}_{n,p}}{P(X|\lambda^{(i)})} \qquad (1.4)
\end{aligned}
$$

The numerator is derived as:

$$
\begin{aligned}
P(q_{n+1} = p', q_n = p|X, \lambda^{(i)}) &= \frac{P(q_{n+1} = p', q_n = p, X|\lambda^{(i)})}{P(X|\lambda^{(i)})} \\
&= \frac{P(X_n, q_n = p|\lambda^{(i)}) t^{(i)}_{p,p'} L^{(i)}(x_{n+1}, p') P(Y_{n+1}|q_{n+1} = p', \lambda^{(i)})}{P(X|\lambda^{(i)})}
\end{aligned}
$$

If the domain of x is finite, such that L is a matrix $L_{p,x} = L(x|p)$, then L can be updated in a similar fashion:

$$L_{p,x}^{(i+1)} = \frac{\sum_{n \text{ if } x=x_n}^{N} P(q_n = p|X, \lambda^{(i)})}{\sum_{n}^{N} P(q_n = p|X, \lambda^{(i)})}$$

If the domain of x is not finite, the expectation value of L must be derived depending on the modeled probability distribution.

Uncertainty propagation in Bayesian inference

One of the advantages of Bayesian models is that uncertainty of predictions are explicitly modeled. In epigenetic studies, multiple machine learning tasks my be performed in sequence, such as base calling, methylation calling, and downstream methylation analyses such as segmentation or differential methylation calling. In order to avoid overconfident predictions, uncertainties can be considered in downstream analysis steps and propagated throughout a study.

In general terms, when chaining machine learning models, such that the output of one model $f(X)$ is input to another, that is $g(f(X))$, the naïve approach is to reduce $f(X)$ to a concrete prediction, like the expectation value or the maximum likelihood estimator. Even more so when multiple models are involved, where each forwards concrete predictions without a measure of uncertainty, this can lead to overconfident downstream predictions [139]. If the model f, however, allows the estimation of uncertainty of $f(X)$, this uncertainty can be used to inform model g [140, 141]. In Bayesian inference, the uncertainty of prediction $f(X)$ can be read directly from the model, making Bayesian methods particularly well suited for uncertainty propagation [142]. Appendix A illustrates uncertainty propagation on a toy example related to methylation profiling.

1.2.2. Deep learning

Deep learning refers to a set of machine learning models and training paradigms inheriting concepts from artificial neural networks (ANN), which has recently been of increasing interest in analysis of sequencing data, particularly in long-read sequencing. Unlike HMMs, deep learning methods typically do not directly model problem specific functions and then learn parameters, but rather model a function space through compositions of linear and non-linear functions. Model training then aims to learn the correct function from this function space including the appropriate parameters [128].

The core component of an ANN is the perceptron [143] $f(XW)$, consisting of an input tensor X, a linear function represented by a weight tensor W, and a non-linear

activation function f. For a binary classification task, the codomain of f may be in the interval $[0, 1]$, whereas for a regression task the codomain of f may be $(-\infty, \infty)$. In an ANN, also called a multi-layer perceptron, a layered network of perceptrons is constructed, such that each perceptron of the next layer takes the output of all perceptrons from the previous layer as an input. The first layer is typically referred to as the input layer, and the final layer as the output layer, while all other layers in between are referred to as hidden layers.

Training an ANN refers to learning of parameters W from a set of training data X and ground truth output Y and a loss function $\mathcal{L}(Y, \mathrm{ANN}(X))$. The goal of training is to find parameters W such that the loss function is minimized. It is typically implemented using gradient-decent algorithms and backpropagation [144, 145].

It has been shown that a single hidden layer is sufficient to solve the XOR problem [146] (being able to model a logical exclusive OR operation) and is in fact sufficient to represent any continuous function on a closed subset of R^n, given that there are sufficient perceptrons in the hidden layer [128, 147]. Deep architectures with a large number of hidden layers have empirically shown great success in learning complex representations. Prior knowledge about the function to be learned also allows for further considerations in model architecture, such as weight reuse between perceptrons or sparse connections instead of fully connected layers [128].

While the original idea of an ANN was acyclic and therefore feed-forward, deep learning architectures such as recurrent neural networks (RNN) can have cyclic graphs, requiring training regimes to unfold the model $|X|$ times. This allows the learning of sequential dependencies in the input (Figure 1.8) [145]. RNNs have been successfully applied to machine learning tasks operating on sequential input, such as natural language [148] or genomic sequences [149, 150]. However, these models do not scale well to very long sequences, as backpropagation through a long unfolded network leads to numerical problems (gradient explosion, gradient vanishing) [151]. Gold standard natural language processing and genomic sequence learning tasks have thus moved on to attention based deep learning architectures [152, 153]. RNNs, however, are still relevant when processing analog sequential information, such as in base and methylation calling of Nanopore sequencing data, as is illustrated in Chapter 1.3.1 and Chapter 1.3.2.

1.2.3. Canonical problems in epigenetics

Epigenetics offers a host of problems too complex for human experts to intuit, requiring machine learning algorithms to disentangle. The human epigenome is large, with around 28.3 million CpG sites [154], around 30 thousand CGIs [155], and around 30 million nucleosomes [156], which may potentially be modified with 43 location specific histone modifications [157]. These epigenetic modalities can affect the transcription of over 22 thousand protein coding genes and over 34 thousand transcripts [158]. This represents both a modeling challenge, seeing how epigenetic regulation

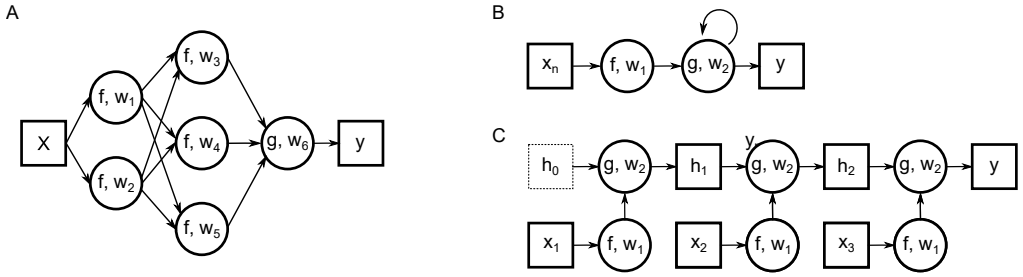

Figure 1.8.: Conceptual representation of artificial neural networks **A)** A feed-forward neural network with 2 hidden layers, totalling 5 hidden units with activation function f and an output layer with activation function g. **B)** A simple recursive neural network with a single hidden unit with activation function f and a recurrent output unit with activation function g, which also takes the previous input's output as a hidden input. **C**) Unfolding of the network in (B) with an input $X = (x_1, x_2, x_3)$ and initial hidden output h_0. The final output is $y = g(w_2 f(w_1 x_3), g(w_2 f(w_1 x_2), g(w_2 f(w_1 x_1), h_0)))$. Alternatively, if the problem requires the output also to be a sequence, the final output could consist of $Y = (h_1, h_2, y)$

is a complex system with an enormous number of interdependent variables, but also a computational challenge, with large amounts of data being stored, retrieved, and computed on at the same time.

Furthermore, recent sequencing technologies require complex machine learning solutions to even do as much as measure these epigenetic modalities in the first place, as with the methylation calling on Nanopore or SMRT-seq data [159, 160].

This section contains a non-exhaustive enumeration of canonical problems in epigenetics requiring machine learning solutions. As the subject of methylation calling on ONT data is covered in Chapter 1.3.2, this section focuses on downstream analysis of methylation or histone modification data.

DMR/ASM testing

Given N biological samples with DNA methylation measurements mapped to the same reference genome, DMR testing aims to find regions in the methylome where the methylation rate is dependent on sample assignment. If methylation data is haplotype phased, methylation calls can be assigned to alleles, and samples are substituted with haplotypes. This analysis is also referred to as ASM testing.

Let $C^m_{i,j}$ be the number of methylated reads mapped to cytosine (w.l.o.g.) j in sample i, and $C^u_{i,j}$ the number of unmethylated reads mapped to the same cytosine. The methylation rate for cytosine j in sample i can then be computed as the beta-score:

$$\beta_{i,j} = \frac{C^m_{i,j}}{C^m_{i,j} + C^u_{i,j}} \tag{1.5}$$

Assuming the sampled methylation rate $\beta_{i,j}$ follows a probability distribution $\mathcal{X}_{i,j}$, the goal is then to determine whether $\forall_{x,y} : \mathcal{X}_{x,j} \overset{d}{=} \mathcal{X}_{y,j}$, and to identify cytosines where this equality does not hold.

Since DNA methylation rates are highly locally correlated, and often show distinct boundaries, the problem is then generally expanded to look not for single cytosines, but rather contiguous regions on the genome with differential methylation. Let R be a partition of all genomic coordinates j, such that each element $r \in R$ is a set of contiguous coordinates. Assuming that all cytosines within the same region r and sample i share the same distribution $\mathcal{X}^r_i$, that is $\forall_{j \in r} : \mathcal{X}_{i,j} \overset{d}{=} \mathcal{X}^r_i$, a test can then be applied for each region.

Numerous statistical tests have been implemented and applied, such as Fisher-Exact test over the contingency matrix of methylated and unmethylated reads per region (only applicable for $N = 2$) or via logistic regression [161, 162].

While the testing of differential methylation between N samples is typically implemented as an explicitly defined statistical test, and thus not technically a machine learning problem, choosing regions to test on is more complex. A naïve solution can be based on known biological annotations, such as transcription factor binding sites, transcription start site, or first exon location. *De novo* identification of DMRs requires genome segmentation, for example based on CpG-density in order to identify CGIs [163] or directly based on methylation signal.

Chromatin instance segmentation

Instance segmentation refers to partitioning of a signal into contiguous regions(Figure 1.9). For the purpose of DNA methylation analysis, an instance segmentation of methylation profiles finds discrete changepoints in methylation rates or other methylation statistics. The R [164] package methylKit [165] implements a methylation segmentation algorithm designed for the segmentation of a single-sample methylome [166] based on the previously published algorithm fastseg designed for the segmentation of copy number profiles [167]. The fastseg algorithm first detects potential changepoints by applying a regularized t-test [168] for each CpG site j, testing whether methylation rates in the regions before and after j are significantly different. Local maxima of the test statistics are then computed and returned as final changepoints.

Alternatively, methylKit also allows for the computation of a 2-sample segmentation by first computing a CpG-level differential methylation $\beta_{\delta,j} = \beta_{1,j} - \beta_{2,j}$ and then performing the fastseg algorithm on this differential methylation profile. Similarly,

the R package MethCP [162] performs segmentation on site-level test statistics such as those calculated by the methylKit package. Segmentation is then performed using the previously published circular binary segmentation algorithm [169].

A segmentation HMM has also been shown to be able to segment a 1-dimensional signal into a fixed number of segments, by defining each state as a segment and designing the transition probabilities such that $t_{n,m} = 0$ except when $n = m$ or $n = m + 1$. Thus, only moving forward to the next state or staying in the same state is permitted. Signal emissions can then be modeled and parameters learned using the Baum-Welch algorithm [170].

Chromatin semantic segmentation

Semantic segmentation refers to prediction of contiguous regions with a set of often predefined class labels (Figure 1.9). The main difference to instance segmentation, is that a semantic segmentation does not separate directly neighboring regions of the same class. For example, methylation data may be semantically segmented into methylated domains (MD), unmethylated domains (UD), and partially methylated domains (PMD), yet no distinction would be made between two neighboring methylated domains even when they are separated by a clear methylation rate changepoint. Fisher-HMM [161] implements an HMM with three states for MD, UD, and PMD, respectively, predicting for each CpG site one of the three hidden states. This results in a 3-class semantic methylome segmentation which is then used for differential methylation calling.

Chrom-HMM [171] implements a multivariate HMM for the annotation of chromatin from a variable number of input measurements, such as DNA methylation or any type of histone modification. It therefore models the combined effect of chromatin states on multiple layers of the epigenome. While the number of hidden chromatin states is user-defined when the segmentation is performed, Chrom-HMM does not require an *a priori* understanding of each state, nor does it offer a ready made interpretation of states.

1.3. Nanopore sequencing and analysis

Short read sequencing technologies based on fragmentation of DNA are limited in their capacity to support analysis of structural variation, haplotype phased epigenetic analysis, and profiling of repetitive chromatin [172]. Long-read/single-molecule sequencing refers to technologies which allow for long molecules to be sequenced without fragmentation or amplification.

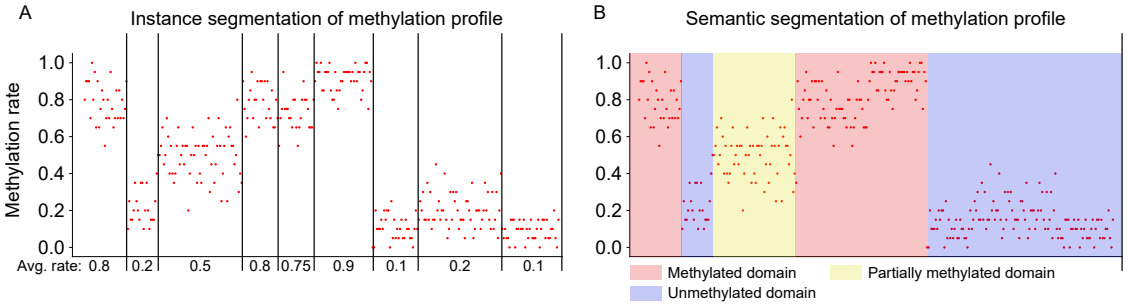

Figure 1.9.: Types of methylome segmentation. **A)** Instance segmentation partitions methylome into a **variable** number of segments. **B)** Semantic segmentation classifies sites as one of a **fixed** number of labels, merging neighboring sites of the same label into segments.

1.3.1. Nanopore sequencing and basecalling

ONT sequencing is currently the only implementation of protein pore based biosensors for the sequencing of long DNA and RNA molecules. Such a biosensor consists of a synthetic membrane separating two chambers filled with an electrolyte solution above an application specific integrated circuit (ASIC). A single pore forming protein derived from proteins naturally occurring in bacteria [173] is inserted in the membrane, creating a channel between the two chambers. Inducing a potential difference between the two chambers leads to a constant ion flow from one chamber to the other, which can be measured by a sensor included in the ASIC. Molecules flowing through the Nanopore temporarily block ion flow and lead to a perturbation in the measured signal (Figure 1.11).

The task of reconstructing the base sequence from the current signal is called basecalling. Since multiple nucleotides fit in the barrel of the Nanopore, the current signal in a version R9.4.1 pore depends on a sequence context of approximately 6 bases, resulting in 4,096 distinct k-mers. Furthermore, while transition speed through the pore is regulated by a motor protein, it still varies depending on the sequence context. Early Nanopore basecallers therefore implemented a two step approach, which first segmented the current signal into instances, where each instance is assumed to be produced by a single k-mer, followed by a classifier which would attempt to reconstruct the k-mer based on deep features computed from the instance signal.

Modern basecallers [174] forego the segmentation step and instead implement end-to-end models with a connectionist temporal classification (CTC) decoder [175]. The current gold standard basecaller is ONT's own Guppy, which implements a neural network consisting of three components: i) a convolutional neural network (CNN) [176, 177] acts as a feature extractor, (ii) a bi-directional gated recurrent unit (GRU) models sequential information flow, and (iii) a CTC decoder maps multiple the GRU's output units to a single character, creating a variable length output

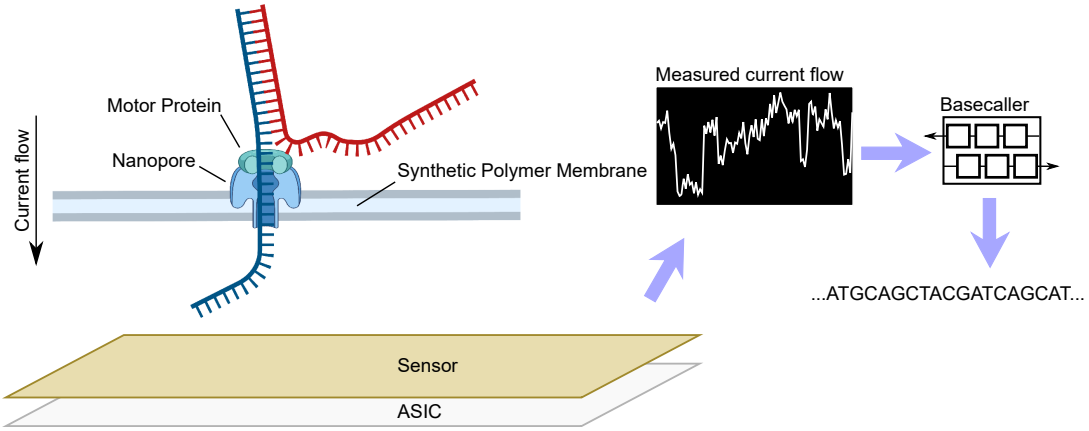

Figure 1.10.: Components of a Nanopore sequencing experiment. Synthetic polymer membrane perforated with a single pore forming protein. A motor protein unwinds double stranded DNA and regulates speed of transition. Measured current signal is basecalled using a deep learning model to arrive at sequence information.

sequence.

While technological developments have lead to significant improvements of basecalling quality, ONT sequencing still struggles to compete with short read sequencing-by-synthesis technologies. In a now outdated, but still most recent, benchmark, Guppy version 2.2.3 has been shown to achieve about 90% read identity (Q-score 10) [174] and reads are typically filtered to include reads of Q-score 7 or greater. The main sources of error include DNA modification, which leads to mistaken basecalls when using a basecalling model not trained for methylated DNA, and homopolymers (sequences with consecutive identical bases) owed to errors in segmentation/CTC decoding. Current promotional material by ONT promises 98.3% read accuracy with current chemistry and software (pore version R9.4.1) with the current Guppy version being 6.1.x [178].

1.3.2. Nanopore methylation calling

Assessments of basecalling quality revealed that cytosine methylation was one of the main sources of basecalling errors, showing that methylated nucleotides result in different current flow perturbations from canonical nucleotides [174]. This indicated that methylation state can be read directly from Nanopore signal, without the need for bisulfite conversion. This lead to the development of a large number of ONT methylation calling software tools [159, 179–181]

While some methylation callers depend on a fully unmethylated control sample [182], modern methylation callers solve the problem as a classification task, classifying

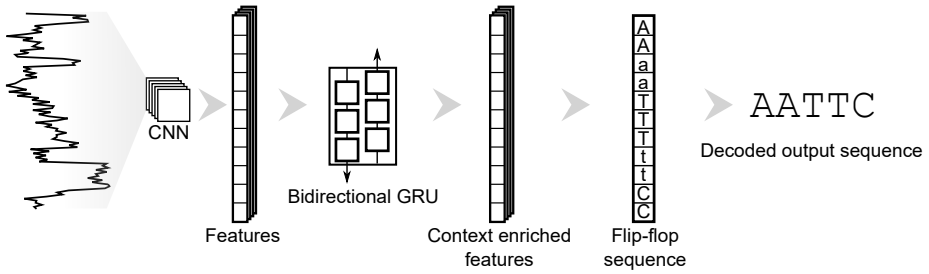

Figure 1.11.: Architecture of ONT basecaller Guppy. A convolutional neural network extracts sequence features. A bi-directional GRU enriches features with contextual information. A CTC decoder translates features to a "flip-flop" sequence, where bases are either capitalized or non-capitalized. The final output sequence is the constructed by merging all consecutive sequences of the same exact character.

basecalled Nanopore signal as either methylated or unmethylated. These existing methylation callers are trained for specific sequence context, like CpG methylation or GpC methylation.

Nanopolish

Nanopolish is a Bayesian methylation caller which models Nanopore signal as emissions in an HMM, given a basecalled and reference aligned sequence and the corresponding raw signal. It accounts for false splits (e.g. base A is read as AA) and false merges (e.g. bases AA are read as A) [180]. The Nanopolish model was originally designed to improve the quality of basecalls for the creation of a polished assembly [183] and is parameterized by emission likelihoods

$$L(X|k-mer) = N(\mu_{k-mer}, \sigma_{k-mer}) \tag{1.6}$$

where $k-mer$ represents the current k length (polished) subsequence, resulting in 4,096 parameter pairs if $k = 6$.

For the purpose of methylation calling, an additional base M is included in the alphabet, and the dictionary is expanded to include k-mers where M replaces cytosine in CpG (w.l.o.g) as well as k-mers ending with cytosine, expanding the model to 6,361 parameter pairs. Parameters were then trained on fully methylated and fully unmethylated samples.

Methylation prediction is then performed by computing emission likelihoods for methylated sequences as well as unmethylated sequences. To do so, sequence S_R is extracted from the reference genome, centered around a CpG-site and expanded on both sides such that the number of bases $|S| = 16$, with the corresponding Nanopore current signal X. If this expands the sequence to cover another CpG-site, it is expanded until each CpG site is flanked by at least k non-CpG bases, since

training data did not contain k-mers with mixed methylation state. A methylated sequence S_M is then computed, where all CG are replaced with MG.

Methylation calls are then computed as log-likelihood ratios (LLR)

$$\mathrm{LLR}_{met}(X, S_R, S_M) = \log \frac{L(X|S_M)}{L(X|S_R)} = \log L(X|S_M) - \log L(X|S_R) \tag{1.7}$$

Nanopolish methylation calls have been shown to predict methylation rates with Pearson $r = 0.8733$ compared to true methylation rates, while tending towards over-calling of methylation [159].

Deep learning based methylation callers

Deep learning based methylation callers DeepSignal [181] and ONT's Megalodon [184] have shown improvement over Nanopolish's Bayesian model, with Pearson $r = 0.9420$ and 0.9860 [159]. These models, however, require expensive GPU hardware, whereas HMM algorithms required by Nanopolish can compute on inexpensive CPUs. Furthermore, while ANN classifiers are trained to predict values in the same range as probabilities $[0, 1]$, they tend to be overconfident and ill calibrated and are therefore not well suited for uncertainty propagation, unless efforts have been made to specifically model probability distributions and calibrate predictions [185].

ONT's newest methylation caller model Remora implements methylation calling after basecalling using a CNN architecture for feature extraction and an RNN classifier. This is not unlike Guppy's basecalling architecture, albeit with fewer parameters [186]. While Remora is developed as open source software, the Remora architecture has since been included in Guppy, allowing for methylation calling in combination with basecalling directly through Guppy. Promotional material promises methylation calling accuracies comparative to bisulfite sequencing, however due to its novelty it has not yet been included in a published benchmark.

Data management

Both DNA methylation calls derived from arrays and on short-read sequencing data are typically stored as tab-delimited text files [106], either based on intensity values (arrays) or as read counts (sequencing). Single-molecule sequencing, however, allows for the read out of many CpG sites per read, and the analysis of these read specific calls requires read-level storage of methylation data. Additionally, if the methylation calls were to be reduced to counts per CpG-site, this would lead to the loss of uncertainty information emitted by the methylation caller and therefore render uncertainty propagation impossible. Furthermore, reducing single-molecule information to counts per CpG-site would remove the read association of each individual methylation call, making it inaccessible for analyses such as ASM analysis, haplotype aware methylome segmentation, or methylation aided read phasing.

A single human methylome sequenced at 30x coverage can yield close to 850 million methylation calls [154, 188]. These need to be stored together with the read identifier, uncertainty estimate, and location, either read based (n^{th} base on the read) or reference based (chromosome and coordinate). For visualization or methylation analysis of specific genomic regions, it is imperative that methylation calls are stored in a fashion that allows random access with minimal disk I/O.

The original Nanopolish publication [180] stored methylation calls in tab-delimited files, where each line consisted of chromosome, start and end location, read name, strand, sequence context, likelihoods for methylated and unmethylated sequence, and log-likelihood ratio. While this format is easily human-readable, it results in a large amount of storage overhead, with duplicated information (e.g read name repeated for each methylation call) and inefficient data types (e.g. storing integers as text). Furthermore, subsetting data to genomic regions requires linear time unless the file is first indexed using for example tabix [189].

Remora, Guppy, and newer versions of Nanopolish have therefore moved on to storing methylation information leveraging the newly introduced MM and ML tags in the BAM format [190]. The BAM format was originally designed for the storage of reference aligned sequencing data. This MM tag stores methylation calls as an integer sequence, where each element represents the number of methylation-capable bases between methylated bases. Let $MM = (m_n)$ be the MM tag for cytosine methylation, where m_n is the n^{th}element in the sequence. Then, the n^{th} methylated cytosine is at the c^{th} cytosine in the read sequence, where $c = n + \sum_{i=1}^{n} m_i$. Since this only gives the coordinate of the methylation call in terms of number of cytosines, one then needs to reference the read sequence, to find the actual methylated coordinate on the read. In order to then map this to a reference coordinate, one needs to further perform a read-to-reference alignment, typically using the CIGAR string [190] in a BAM entry. The ML tag then contains a sequence of 8-bit integers which store the methylation call probability in the range of 0-256 and can be directly cross-referenced with the MM tag.

Since the MM tag stores methylated base coordinates as small integers, it makes very efficient use of storage space. However, reading the methylation information is a computationally expensive step. First, the entire BAM entry needs to be read and deflated. Then, the read-sequence needs to be loaded into memory together with the CIGAR string and the MM and ML tag, in order to resolve coordinates of methylated bases as described above.

Since the MM tag had originally been designed only for the storage of positive methylation calls, it was also impossible to store the certainty of negative methylation calls (that is, calls that indicated an unmethylated base). An additional flag to the MM tag had been proposed by the developers of Nanopolish, allowing the storage of negative methylation calls [180]. With the `?` flag set at the preamble of the MM tag, the elements of the MM tag are now interpreted as uncalled bases instead of unmethylated bases. While Remora, Guppy, and Nanopolish now write the

MM tag with this optional flag, the software library HTSLib [190], used for reading and writing of the BAM format has been adapted for this updated specification as of August 2022. The canonical python wrapper pysam [191], however, has not yet been updated to these changes.

2. MetH5 - efficent storage format for methylation calls from Nanopore

Methylation calls from long-read single-molecule sequencing, such as ONT or SMRT-seq, have very different storage requirements from methylation data acquired by short-read bulk or single-cell sequencing. Instead of storing methylated and unmethylated read-counts per genomic coordinate in a 1-dimensional fashion, such as in the Bismarck [106] file format or VCF [192] file format, each individual read's methylation calls need to be stored. The data can be presented as an ultra-sparse 2-dimensional matrix, with genomic coordinates on the x-axis and read identifier on the y-axis (Appendix Figure C.1). In other words, methylation calls from short read data require storage space in the scale $O(N)$, where N is the size of the reference genome. Methylation calls from long-read data, on the other hand, require storage space in the scale $O(DN)$, where D is the sequencing depth.

Furthermore, methylation calls come with methylation caller uncertainties. Much like quality strings in DNA sequencing, these need to be stored as well, making a methylation call not a binary data type but rather a floating point. Read names for each methylation call must be included, too, which are typically long character strings that should not be redundantly stored for each call. These increases in storage requirement not only result in a demand for higher capacity storage devices, but also put stress on access times. Disk reads, decompression, and querying of genomic regions become more expensive, thus impacting analysis times and usability.

One of the first file formats used for methylation storage for long-read data is the native output format written by Nanopolish (Chapter 1.3.2). This tab-delimited text-format contains one line per methylation call, resulting in redundant information, such as repeating read names for each of the read's calls. It is difficult to query, expensive to parse, and inefficient to store. Methylation storage in the existing BAM format is a more recent development (Chapter 1.3.2) and is used by modern methylation callers Remora and Guppy, as well as an alternative branch of Nanopolish. This method is optimized for low storage usage at a significant cost to access times (Chapter 2.4).

In order to address the requirement for an efficient data storage container for methylation calls from long-read sequencing, I designed and implemented **MetH5**. MetH5

is a file format based on hierarchical data format (HDF) version 5 [193], and addresses these unique requirements on two fronts: (1) by making efficient use of storage space, avoiding redundancies, and using efficient data types and compression, and (2) by designing an architecture and algorithms for rapid access which avoid unnecessary disk I/O and redundant operations. Furthermore, MetH5 has been implemented with parallelization in mind, allowing for even load distribution of concurrent computations using chunked data storage.

The MetH5 format is defined through a python API implemented and published as the open source python package `meth5`. This python package implements algorithms for efficient random access and aggregation methods operating on genomic windows or reads. It also includes a command line interface (CLI) for convenient creation, modification, and querying of MetH5 files. In order to allow for the analysis of multiple haplotyped samples, the MetH5 format also supports annotating reads with read-group assignments, using the term read-groups analogously to how it is used in the BAM format. This allows combining methylation calls from multiple samples, or the annotation of read phasing into haplotypes, directly in one MetH5 file. Read groups can then also be utilized in aggregation algorithms, e.g. computing the methylation rate of a region per read-group.

In this chapter, I present the design and implementation of MetH5 and evaluate its performance in comparison with the alternative of storing methylation calls in BAM files. The comparison shows that MetH5, albeit coming at a slightly increased storage space requirement, offers a massive improvement to access times, both for random as well as sequential access.

2.1. Requirements analysis

The first step for designing the MetH5 format consisted of identifying functional as well as non-functional requirements. The following list of requirements are then addressed in the design of the storage format for long-read single-molecule methylation data.

2.1.1. Functional requirements

This section includes the functional requirements to the format. That is, features and processes, including the content of the MetH5 container and the routines provided to users in order to interact with the format.

RQ1.1 Necessary data stored: The following data need to be stored for each methylation call emitted from the methylation caller: 1) genomic range consisting of chromosome (or contig), start coordiante, and end coordinate, 2) methylation LLR (Equation 1.7), and 3) read association in form of a locally (to this file) unique read identifier.

RQ1.2 Optional data stored: The following data can be stored if desired by the user, but are not mandatory for a complete MetH5 archive: 1) The original globally unique read name given by the ONT sequencing software. While read names produced by current versions of ONT software follow a specific scheme (160-bit identifier formatted as a hexadecimal number), the MetH5 container should support arbitrary alphanumeric read names. 2) An arbitrary number of read-group associations. For each read-group mapping, a read-group type needs to be stored as a string (e.g. "haplotype" or "sample"). Within each read-group mapping, each read is assigned to exactly one read-group.

RQ1.3 Optional access: Locally unique read ids as well as globally unique read names can be read separately from genomic ranges and methylation LLRs. Thus, methylation information can be accessed without read association if desired, for reduced disk I/O.

The following are requirements for the Python API:

RQ2.1 MetH5 creation from Nanopolish: The Python API shall allow for the creation of MetH5 files from multiple Nanopolish output files, or from similarly structured data where each methylation call consists of the three necessary data (RQ1.1).

RQ2.2 Random genomic access: The Python API shall allow for access to all methylation calls of a specific genomic range, defined by chromosome (contig), and start and end coordinate. This form of access is important for targeted analyses as well as for the purpose of visualization.

RQ2.3 Chunked access: Load-balancing on parallel systems requires an even distribution of data and operations across processes. Often, load balancing for genomic data is performed on genomic coordinates, such as fixed windows in the unit of basepairs. Parallel operations are then distributed over these windows. Since both read-coverage as well as CpG density varies across the genome, however, this does not result in evenly distributed workload. Particularly in highly repetitive regions, such as in pericentromeric regions which also contain CpG methylation [194, 195], a genomic window in reference space might contain disproportionally many methylation calls (Appendix Figure C.2). The Python API shall therefore also allow for access to data chunks containing a specified chunk-size, where the unit of the chunk-size is the number of methylation calls. The accessor may be expanded to include all methylation calls for the first and/or last coordinate in order to not split calls for one coordinate across chunks.

RQ2.4 Sparse matrix creation: The SciPy python package [196] implements operations over sparse matrices. The MetH5 Python API shall allow for construction of SciPy sparse matrices directly from data read from a MetH5 container.

RQ2.5 Aggregation per genomic coordinate: A standard operation in methylation analyses is to compute aggregations for all methylation calls of a certain genomic coordinate. Common examples include methylation rate per CpG-site or

overall read-coverage. Since methylation calls are not binary, the exact method with which a methylation rate is computed may depend on user preference. One possibility would be to drop uncertain calls, defined by an absolute LLR threshold, and then binarize the remaining calls before computing a methylation rate (Equation 2.1). Another may be to compute the sum over all LLR and then compute the sigmoid in order to compute an methylation probability (Equation 2.2).

$$\beta_t(Z) = \frac{|\{\text{LLR} \in Z : \text{LLR} > t\}|}{|\{\text{LLR} \in Z : |\text{LLR}| > t\}|} \tag{2.1}$$

$$P(met|Z) = \sigma\left(\sum_{\text{LLR}\in Z} \text{LLR}\right) \tag{2.2}$$

where Z refers to any set of LLRs. The diversity in different interpretations of methylation calls leads to the requirement that the Python API support custom aggregations over methylation calls, provided as user-implemented functions.

RQ2.6 Aggregation per read: Similarly to RQ2.5, the Python API shall support computing aggregates per read in order to allow computation of read statistics, such as read-level methylation rate or number of methylation calls per read.

RQ2.7 Read grouped aggregation: In a multi-sample or haplotyped experiment, it may be informative to compute statistics for each read-group. The aggregations supported by RQ2.5 and RQ2.6 therefore should therefore support computing on a read-grouped basis. In other words, a user should be able to directly query the methylation rate of a certain CpG site for sample A as well as for sample B.

RQ2.8 User interface: The Python API shall also implement a CLI for the creation of MetH5 files from Nanopolish output.

2.1.2. Nonfunctional requirements

The following non-functional requirements define quality attributes related to the performance of the MetH5 container and API:

NFRQ1.1 read names stored uniquely: In Nanopolish output format the globally unique character string read names are repeated for each methylation call. In a 30x human genome with 44-character read names and an average read length of 8 kilobases, that would result in approximately 37.4GB of data just for storing read names. If instead, a 4-byte integer is used as a locally unique read-id and read names are stored uniquely, the total storage requirement is reduced to 3.9GB. Therefore, read names should be stored uniquely and referred to by a locally unique numeric read id.

NFRQ1.2 Efficient random genomic access: Random access to genomic ranges should be efficient and should not require loading large blocks of unrelated data as when linearly searching through coordinates.

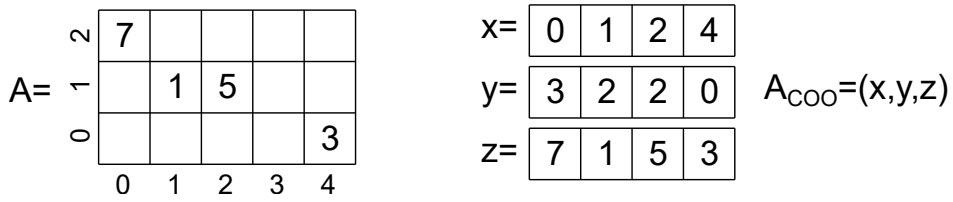

Figure 2.1.: Example of sparse matrix representation in coordinate (COO) format. Left: dense matrix representation with missing values with storage requirement $O(xy)$. Right: Sparse matrix representation in COO format with storage requirement $O(n)$ where n is the number of values not missing.

2.2. MetH5 implementation

Once the requirements were defined, I designed the MetH5 architecture as a container for chromosome organized sparse methylation matrices.

2.2.1. MetH5 as a coordinate format sparse matrix

Since methylation calls with genomic coordinate and read association can be presented as a sparse 2-dimensional matrix (Appendix Figure C.1), I decided to incorporate ideas from existing sparse matrix representations in the MetH5 design. Sparse matrices can be represented in a number of ways. For MetH5, the coordinate format was chosen, as it allows fast subsetting of content based on either or both dimensions (requirements RQ2.2, RQ2.6, and RQ2.7). Furthermore, the coordinate format is well supported in existing implementations, such as the python package SciPy [196], which allows for an efficient implementation of requirement RQ2.8.

In a coordinate format sparse matrix, content is stores as three vectors $x \in \mathbb{N}^n$, $y \in \mathbb{N}^n$, and $z \in \mathbb{R}^n$, where x_i and y_i store the coordinates of the value z_i (Figure 2.1). In MetH5 these three vectors are created for each chromosome, where x defines the genomic coordinates, y the read identifier, and z the log-likelihood ratio. Since MetH5 supports genomic ranges as coordinates (requirement RQ1.1), x is represented as sorted genomic ranges (with start and end coordinates).

2.2.2. HDF5 schema defining MetH5 format

I implemented the MetH5 format as an HDF5 container [193]. HDF5 is a file format for data management of multiple data matrices organized in a hierarchical directory-tree structure. HDF5 supports chunked data storage, making it particularly suitable for parallel processing, data compression, and cached reading and writing. As an industry standard file format, HDF5 is supported by a variety of languages and platforms. The highly optimized core HDF5 library is typically wrapped with language

specific APIs, such as the h5py package [197] for python, which is used in this work.

An HDF5 file consists of three major components. 1) *Datasets* are at the core of the HDF5 format and are multidimensional matrices for data storage. 2) *Groups* are essentially directories which are used to organize datasets and can be arranged hierarchically. A group can contain a number of datasets as well as a number of child-groups. 3) *Attributes* are dictionaries that store key-value pairs and are suitable for storage of meta-data. Each group and each dataset contains a single attributes dictionary.

The chunked storage and random access to HDF5 dataset elements without loading the entire dataset made HDF5 a natural choice in order to help fulfill requirements RQ2.2, RQ2.3, and NFRQ1.2.

The MetH5 specification foresees two top-level groups, named `chromosomes` and `reads`, designed for storing methylation calls organized in genomic coordinates and read information, respectively(Figure 2.2). The `chromosome` group can contain an arbitrary number of sub-groups, one for each chromosome or contig, with the group name representing the chromosome name. Chromosome group c contains all the necessary information defined in requirement RQ1.2. Let n_c be the number of methylation calls in chromosome c, then the following datasets are mandatory contents:

- `range`: Integer dataset of shape $(n_c, 2)$, storing the start and end locations in reference space on chromosome c of each methylation call, such that `range[i,0]` refers to the start and `range[i,1]` the end location of the i^{th}methylation call. If the methylation call refers to a single base, the two values are equal.
- `llr`: Float dataset of shape (n_c), storing methylation LLRs. Thus, `llr[i]` refers to the LLR of the i^{th}methylation call of chromosome c .
- `read_id`: Integer dataset of shape (n_c), storing the locally unique read ids, such that `read[i]` refers to the id of the read of which the i^{th}methylation call of chromosome c originated.
- `chunk_ranges`: Integer dataset of shape $(\lceil \frac{n_c}{C} \rceil, 2)$ where C refers to the chunk-size. This dataset stores for each chunk the genomic start and end locations on chromosome c and enables fast random access as in requirement RQ2.2 and NFRQ1.2. Therefore,`chunk_ranges[i, 0]` contains `range[(C*i),0]` and `chunk_ranges[i, 1]` contains `range[(C*(i+1)-1,1]`, the start and end of the first and last methylation call in chunk i, respectively. This allows access to the entire chunk index while loading only contiguous file-system blocks.

Therefore, a methylation call i is represented as the tuple (`range[i], llr[i], read_id[i]`). All n_c methylation calls within chromosome c are sorted by genomic coordinates, first by start position and second by end position, with undefined behavior in the case of ties.

The `reads` group is fully optional and contains read names and read-group assignments (requirement RQ1.2). It contains a single optional dataset:

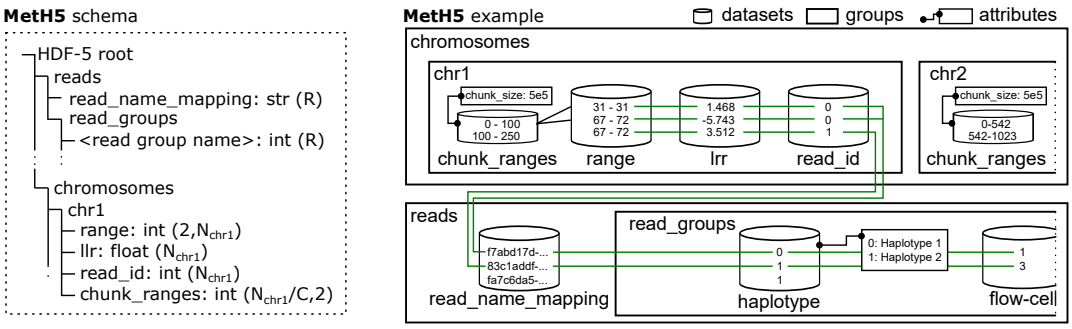

Figure 2.2.: Overview of the components of a MetH5 container. The two main HDF5 groups are `reads` and `chromosomes`. Within `chromosomes`, child-groups are named after chromosomes and contain `range`, `llr`, `read_id` and `chunk_ranges` datasets. The contents of `read_id` act as an index to the `read_names_mapping` dataset as well as to the read-groups datasets in the `reads` group. Read groups here are exemplified here by two read-groups: haplotype and flow-cell.

- `read_name_mapping`: String dataset of shape (r), where r refers to the total number of reads contained in the MetH5 container. This dataset stores the original globally unique read names. Each read name is stored exactly once in order to fulfill requirement NFRQ1.1, such that `read_name_mapping[read_id[i]]` where `read_id` came from chromosome c refers to the read name of the read of which the i^{th}methylation call of chromosome c originated.

Furthermore, the `reads` group contains an optional child-group named `read_groups`, which itself can contain an arbitrary number of datasets. Each dataset represents one read-group mapping, fulfilling requirement RQ1.2. For example, this can be used to store read-haplotype assignment by creating a dataset like:

- `haplotype`: Integer dataset of (r) storing a numeric read-group id for each read. If `read_id` is taken from chromosome c, then `haplotype[read_id[i]]` refers to the haplotype assignment of the read of which the i^{th}methylation call of chromosome c originated. Additionally, read-group datasets contain attributes which map the read-group id to a string read-group label, avoiding replicate storage of read-group labels. Read group id -1 is reserved to be interpreted as unassigned. For example, in this case unphased reads in haplotype phasing would be assigned read-group id -1.

2.3. MetH5 python API

I developed a Python API which acts as the canonical interface with MetH5 files. The `meth5` Python package consists of an API for Python programmers, as well a CLI

for general users (requirement RQ2.8). The CLI is described further in Appendix B. The following scientific Python packages are required as dependencies: NumPy [198], SciPy [196], pandas [199], h5py [200].

The Python API implements an algorithm for the iterative creation of MetH5 files from multiple Nanopolish output files or similarly structured data (RQ2.1). It further implements chunked access (RQ2.3) and fast random access (RQ2.2) algorithms, which take advantage of the `chunk_ranges` dataset. It implements aggregation functions (RQ2.5-2.7) including a number of standard aggregates, as well a function for direct conversion of methylation data from genomic regions or chunks to Scipy sparse matrices (RQ2.4).

2.3.1. Construction of a MetH5 file

In order to make MetH5 files mutable and expandable, all datasets are created without a maximum size, so that resize operations can be performed on the datasets. When creating a MetH5 file, a MetH5 object is created in write mode, after which individual methylation calls can be added in batches. Each batch consists of tuples of chromosome name, reference start and end, methylation LLR and read name. The MetH5 library will expand the HDF5 datasets by the exact number of added methylation calls and initially append the new data to the back of the datasets. In doing so, a "dirty" flag is set in the MetH5 object, which informs the Python API that the dataset needs re-sorting upon closing.

Globally unique read names for added methylation calls are first decoded into locally unique read ids, by looking up the already contained read names in the read name mapping dataset. If a read name is new and has not yet been written to this MetH5 container, the read name is added to the read name mapping dataset and a new read id is issued as the index of the read name in the read name mapping dataset.

When the "dirty" flag is set while the MetH5 object is being closed and resources are being freed, or when a read-access is performed, the Python API will first re-sort the MetH5 container. To do so, for each chromosome all genomic ranges are loaded into memory and sorted using the Mergesort [201] algorithm implemented in the Python package NumPy [198]. The sort order is then used to sort the LLR and read id datasets accordingly.

While the incremental/batched addition of methylation calls is designed to keep the memory profile consistent during MetH5 creation, the sorting step at the very end is the only step where memory requirement scales with the size of the file. However, the sorting is performed per-chromosome and only one index-array and one data-array are stored in memory. The memory requirement for these 4-byte arrays of dimension (n_c) is within acceptable ranges. For example, a 30x sequencing experiment of a human methylome will then, on the largest chromosome, require approximately 538MB of memory while re-sorting.

After re-sorting, the chunk ranges dataset is updated such that random access to genomic regions can take advantage of the indexing.

2.3.2. Random genomic access algorithm

For random access to genomic regions, I implemented a three step algorithm which takes advantage of the pre-computed index stored in the chunk ranges dataset. Given coordinates (c, s, e) where c refers to the chromosome and s, e to the start and end position, the first step is to find the chunks containing positions s and e. This only requires reading the chunk ranges dataset, which is small enough to remain in cache and does not need to be repeatedly loaded from disk for successive accesses. Next, the range dataset for chromosome c is accessed at only the chunks containing s and e. A binary search is then performed using the implementation in the NumPy [198] library, in order to determine the exact start and end location in the range dataset within the respective chunks. Let $s_{\texttt{range}}, e_{\texttt{range}}$ be the indices in the range dataset for coordinates s and e, respectively, the Python API will then return an accessor object which contains these indices. This accessor object can then be used to subset the LLR or read id datasets in order to directly read these data optionally and independently in $O(e_{\texttt{range}} - s_{\texttt{range}})$ time (constant with regards to n, the number of methylation calls in the container).

Algorithm 1: MetH5 indexed access to methylation calls in a given genomic range performs in $O\left(\frac{n}{C}\right)$ time

Data: HDF5 datasets llr, range, read_id, chunk_ranges and chunk size C for chromosome c

Input: Genomic cordinates s (start) and e (end), both integer, for chromosome c

Result: `llr[`I`]`, `range[`I`]` and `read_id[`I`]` contain LLR, genomic ranges, and local read ids for all methylation calls between s and e, respectively

// implemented as linear search in $O\left(\frac{n}{C}\right)$ time

$s_{\text{chunk\_ranges}} \leftarrow \arg\min_i$ chunk_ranges$[i,0]$

$e_{\text{chunk\_ranges}} \leftarrow \arg\max_i$ chunk_ranges$[i,1]$

// implemented as binary search in $O(\log C)$ time

$s_{\text{range}} \leftarrow \min\{i : 0 \leq i < C,\ \text{range}[C * s_{\text{chunk\_ranges}} + i,\ 0] = s\}$

$e_{\text{range}} \leftarrow \max\{i : 0 \leq i < C,\ \text{range}[C * e_{\text{chunk\_ranges}} + i,\ 0] = e\}$

// accessor object defines range in datasets

$I \leftarrow (s_{\text{range}} : e_{\text{range}})$

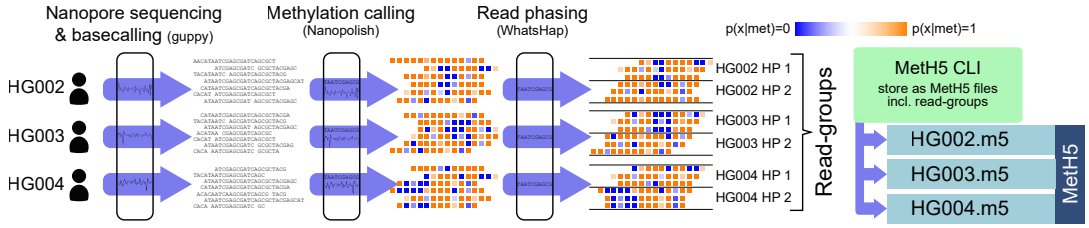

Figure 2.3.: Preprocessing of Genome in a Bottle (GIAB) [204] benchmark data for evaluation of MetH5 and pycoMeth. Figure adapted from the pycoMeth publication [207].

2.4. Evaluation

In order to evaluate the MetH5 implementation, I set up a benchmark scenario where I compared MetH5 with read-anchored methylation call storage in the BAM format [190]. The BAM format is designed for the storage of read alignment to reference and has been extended with a wide variety of tags for the annotation of read mappings. In parallel to my development of MetH5, the BAM format had been extended by two additional tags for the storage of methylation calls. The MM tag stores the location of methylation calls as a read-anchored methylation motif index, whereas the ML tag stores methylation probability for each methylation call. At the time of my benchmark, the API which implements reading and writing of BAM files, htslib [202], had only limited support for MM and ML tags, and the python implementation pysam [191] had not yet implemented it. Therefore, I used the python package modbampy [203], implemented by ONT, in order to interface with methylation calls in BAM files.

I produced benchmark data from publicly available raw Nanopore sequencing data produced by the Genome in a Bottle (GIAB) consortium [204], specifically the Ashkenazim trio consisting of samples HG002 (son), HG003 (father), and HG004 (mother). From these samples, I downloaded raw Nanopore data for four flow-cells from HG002, and three Nanopore flow-cells for HG003 and HG004 each, which yields a sequencing depth of roughly 20x to 30x coverage per sample. I basecalled the raw data using guppy version 5.0.11 with the high-accuracy model trained with modified bases (methylation-aware basecaller model). Then, I aligned the data to reference genome GRCh38 [205] using minimap2 [206] and used Nanopolish [180] for methylation calling. The benchmarking of the MetH5 format only uses the HG003 sample, but the HG002 and HG004 samples were pre-processed in the same fashion to be used in the evaluation of pycoMeth (Figure 2.3).

In order to produce both a MetH5 file and a BAM file with methylation data, I ran two different versions of Nanopolish on HG003. The MetH5 file was produced by first running the latest stable version of Nanopolish (release 0.13.3) to produce methylation data in the tab delimited format, which I then converted to MetH5

using the MetH5 CLI. The BAM file was produced using the at the time experimental implementation in the "methylation_bam" branch of Nanopolish (commit 9B01ad7). BAM files were also compressed to CRAM files using `samtools view -c` and MetH5 files were stored using two different dataset compression options: LZF [208] and gzip [209].

For the comparison of storage space requirement, I consider the storage requirement of the MM and ML tag in the BAM files. To do so, I computed the difference between the size of BAM files with methylation data stored in MM and ML tag (output by Nanopolish methylation calling) and BAM files without MM and ML tag (before methylation calling). Here I find that BAM storage is more efficient than MetH5 storage, as MetH5 requires 2.16GB using LZF compression and 1.47GB using gzip compression. The methylation called BAM file only requires 1.31GB and the CRAM file 0.77GB in addition to the original non-methylation called files (which were 83GB and 53GB, respectively). Partly this can be explained by BAM storage taking advantage of already existing information, such as read names. Read names are stored in MetH5 files and contribute to the total size, but are not part of the additional space required by the MM and ML tag in BAM files. Another factor, however, is also that the MM tag in BAM files stores small read-level indices which can be stored with fewer bits than the genomic coordinates saved for each methylation call in the MetH5 format (Figure 2.4).

For access times benchmarks, I implemented two comparisons, one for sequential access and one for random access. The sequential access benchmark reads methylation values for one entire chromosome in order to compute a global methylation rate. With MetH5, only the LLR datasets needed to be read, as it is independent of read assignment or genomic coordinates. With BAM files the entire BAM entry still needs to be deflated, giving MetH5 a major advantage. The sequential test took 10 and 56 seconds on the LZF and gzip compressed MetH5 files, respectively. On the BAM files using modbampy it took 245 seconds and on CRAM files 277 seconds.

Next, I tested random access which relies on the indexing mechanisms of both file formats. To do so, I implemented a random coordinate generator which generates random 1,000bp regions across all chromosomes. I then generated 100 sets of 100 segments (10,000 segments in total) and compute the total loading times for each set. For both MetH5 and BAM files the file objects are opened only once in order to not multiply BAM header parsing. The average reading time measured was 1.6 and 11.2 seconds for LFZ and gzip compressed MetH5, respectively. In BAM files the random test also performed slower, with 53.5 and 143.0 seconds for uncompressed BAM and CRAM files, respectively. Comparing LZF compressed MetH5 files and uncompressed BAM files, this corresponds to a 24.5 fold speed improvement in sequential reading and a 33 fold improvement in random access (Figure 2.4).

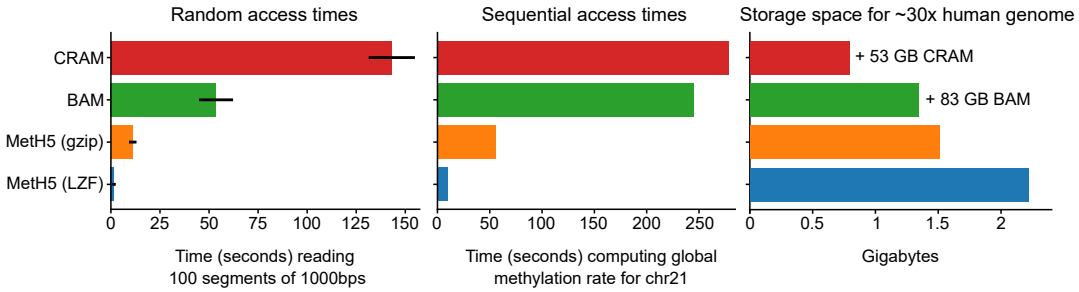

Figure 2.4.: Benchmarking the MetH5 format against methylation storage in BAM/CRAM files. Random access times are measured as the time required to read 100 random segments of 1,000bps length. Error bars represent one standard deviation after 100 trials. Sequential access time refers to the time required to simply load all methylation calls from a single chromosome. For the storage space comparison, only the difference between BAM/CRAM with and without MM and ML tag are presented, with the number behind the plus sign showing the total size of the BAM/CRAM file without the tags. MetH5 shows massively improved access times compared to BAM files, at a low cost to storage space. Figure adapted from the pycoMeth publication [207].

2.5. Code availability

The meth5 python package [210] is available on GitHub, and distributed via package managers pypi and anaconda.

- GitHub: `https://github.com/PMBio/MetH5Format`
- pypi: `https://pypi.org/project/meth5/`
- anaconda: `https://anaconda.org/snajder-r/meth5`

The benchmarking code used to generate runtime comparison and figures is also available on GitHub under:

`https://github.com/snajder-r/benchmark_meth5`

2.6. Conclusion

With MetH5, I implemented an efficient container for long-read single-molecule methylation sequencing data, complete with a useful python API and CLI. MetH5 storage offers rapid random access to genomic coordinates, which is important for targeted calculations as well as visualization. While BAM storage is an established file format which is well supported, it exists as a monolithic container inseparably storing a large number of data modalities. The BAM file used in the benchmark described above were 84GB (BAM) and 54GB (CRAM) in size, with methylation

data making up less than 2% of the file size. A smaller file format such as MetH5, which is dedicated to a single data modality, can be advantageous for data sharing or data transfer to different compute environments for further analysis or visualization. Another difference worth mentioning is that MetH5 stores methylation data in reference anchored coordinates, whereas MM tags in BAM files are read-anchored. The consequence of this can be that basecalling errors which could have been corrected in reference alignment may cause missing or misleading methylation calls.

Most importantly, however, MetH5 outperforms methylation storage in BAM when it comes to access times, with about 33 fold increase in random access and 24.5 fold increase in sequential access speed. Furthermore, MetH5 is explicitly designed for parallel computing. These advantages make MetH5 an excellent storage format for downstream methylation analyses and visualization.

3. pycoMeth - differential methylation analysis toolbox

Case-control studies on methylation data are used for the identification of disease- and treatment relevant markers [211–213]. Differential methylation can be computed for specific CpG sites or over genomic regions and software tools for computations based on methylation rates from WGBS are available [162, 166]. Existing tools for WGBS can be repurposed for ONT methylation calls. To do so, data must first be converted to a count per CpG format, thus emulating data from a WGBS experiment (pseudo-bulk). In addition to losing read specific information, this reduction also requires binariziation of methylation LLR, leading to the loss of uncertainty information. Dedicated tools for long-read methylation data which allow for the consideration of methylation call uncertainties and read-level information are currently lacking.

With these aspects in mind, pycoMeth was developed as a specialized differential methylation calling and reporting tool.

A first prototype of pycoMeth was developed by Adrien Leger at the European Bioinformatics Institute (EMBL-EBI), Hinxton, UK. This prototype relied on tab-delimited Nanopolish output files for its input. It could be used to perform differential methylation computation on pre-defined genomic regions, followed by effect-size filtering to remove low effect-size results. Statistical testing was performed using a non-parametric test on LLRs of 2 or more samples using the SciPy [196] Python package. Correction for multiple testing was provided by the statsmodels [214] Python package. Identified DMRs were reported in easily accessible HTML reports generated using plotly [215]. Additionally, it implemented a CGI-detector which would operate on the reference genome to compute putative CGIs.

This prototype came with a number of limitations, which I addressed in the course of my PhD project. The CGI-detector was implemented in order to allow DMR detection with limited *a priori* information about methylation organization. Instead of requiring user-defined regions, the putative CGIs predicted from the reference genome could be used for differential methylation testing. This strategy is based on the assumption that DMRs would be found within CGIs. Literature, however, suggest that aberrant methylation in disease frequently occurs outside of CGIs [41, 43].

The non-parametric test for differential methylation performed in the prototype

compares all LLRs in a region. Such a test draws power from both coverage and segment size, and may therefore be biased towards larger segments (Appendix Figure C.4). Finally, the prototype version performed effect-size filtering before testing for significance, which also creates false biases and can lead to overconfident predictions [216, 217].

As for computational performance, the pycoMeth prototype was primarily constrained by the reliance on tab-delimited input files produced by Nanopolish. With this input format, all methylation calls have to be read from disk before being able to subset to specific genomic regions. Since input files had to be read sequentially (line by line), the degree of parallelization was limited as well, making it difficult to take advantage of parallel hardware.

In order to address these limitations, I designed and implemented pycoMeth version 2 (Figure 3.1). The first major change in pycoMeth version 2 is reading input from the MetH5 format. Using MetH5 as a data structure improves overall runtime performance of pycoMeth significantly. It also also allowed me to implement multiprocessing in order to take advantage of multi-core / multi-CPU hardware for an added performance boost. The second major change is the implementation of a Bayesian methylome segmentation algorithm, which will be covered in detail in this chapter. This segmentation method allows for *de novo* segmentation based on methylation calls directly. The model explicitly accounts for methylation caller uncertainties and considers read-group association (such as sample or haplotype) stored in the MetH5 container. Finally, major changes in the differential methylation testing module of pycoMeth allow for a variety of tests with better calibration than the test implemented in the prototype.

3.1. Bayesian methylome segmentation algorithm

In order to allow for *de novo* discovery of differentially methylated regions in all sequence contexts, whether CGIs, shores, shelves, or open-sea, I developed a Bayesian methylome segmentation algorithm. I first identified the following design criteria: (i) The segmentation method must produce a single consensus segmentation from multiple methylation profiles. In order to test for differential methylation between multiple samples (whether these are biological samples or haplotypes of the same biological sample), a single segmentation valid for both samples is required. (ii) While a maximum number of segments may be set, the number of segments inferred should be dynamic and decided *de novo*. (iii) The segmentation must be independent of differential methylation. While the same samples on which differential methylation will be computed may also be used to inform segmentation, it is important that the segmentation is indifferent to differential methylation. If this requirement is not satisfied, it could create a false bias towards differential methylation, leading to overconfident DMR prediction. (iv) Methylation caller uncertainties should be propagated in the model. Uncertainties emitted by methylation callers, such as the

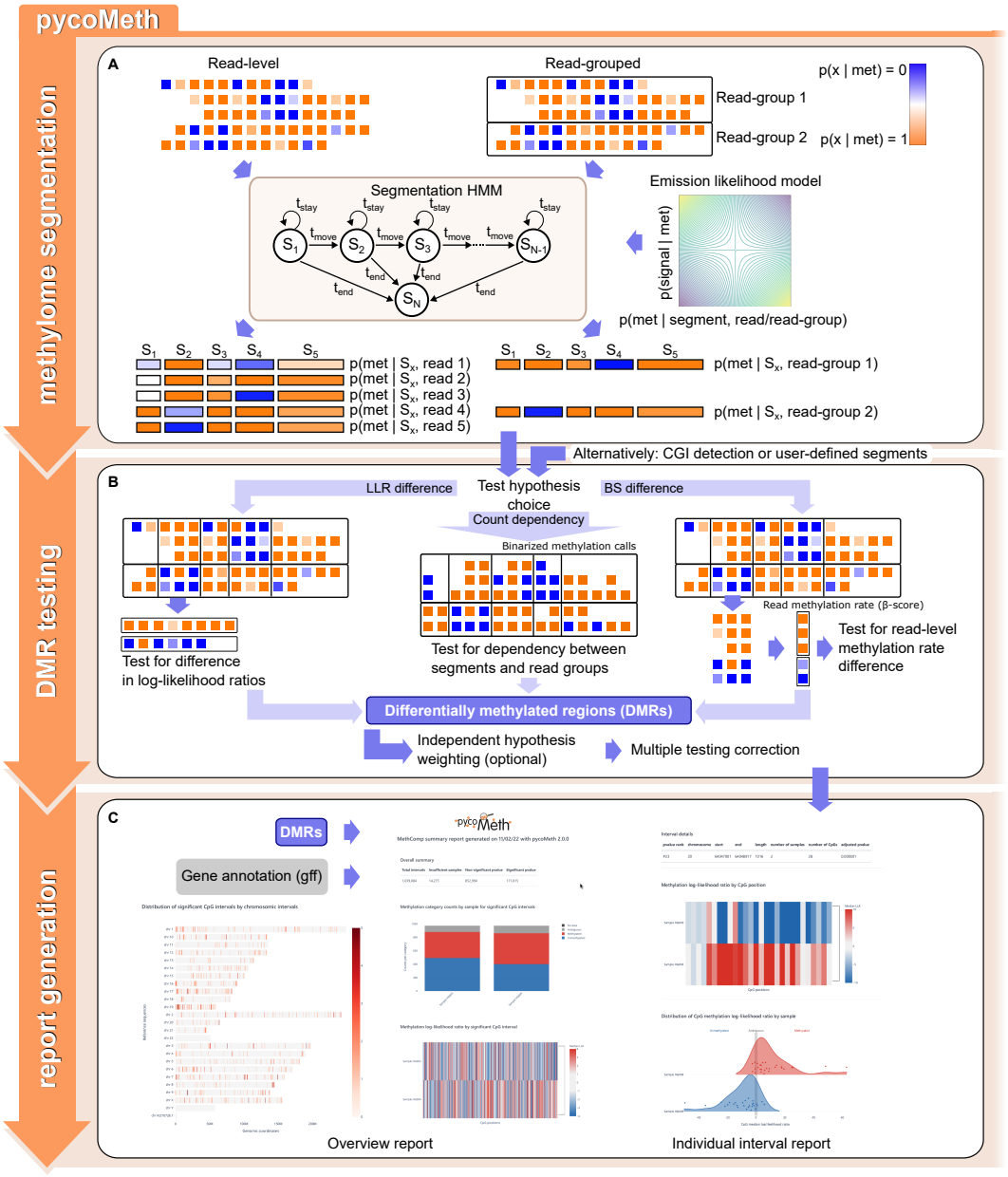

Figure 3.1.: Sketch of the pycoMeth software suite and workflow, figure adapted from the pycoMeth publication [207]. **A)** The Bayesian methylation segmentation algorithm I implemented for pycoMeth version 2. Designed as a changepoint-detection HMM, methylation calling uncertainties are modeled in emission likelihoods. Methylation calls are modeled as Bernoulli distributed events with a methylation rate for each read (or read-group) and segment. **B)** The differential methylation testing module. In addition to the original LLR difference test, I implemented two additional tests: one based on read-level methylation β-scores, and another based on a contingency matrix of methylated CpG-sites. Furthermore, I implemented independent hypothesis weighting (IHW) [217] as part of multiple testing correction. **C)** The reporting module generates HTML reports for DMRs.

LLRs emitted by Nanopolish, must be considered in the emission likelihoods and properly propagated in a Bayesian fashion.

The segmentation algorithm is implemented as a changepoint detection HMM [170]. Notable novelties in this implementation include that methylation call uncertainties are explicitly accounted for as part of the emission likelihoods, and methylation rates are modeled as a property of segment and read. That is, the methylation rate parameters $\mu_{s,r}$, where s represents the segment and r the read, are learned as part of the segmentation. Furthermore, pycoMeth implements a read-group mode, in which read-groups stored in the MetH5 container can be used to inform the segmentation. A read-group can then represent a biological sample or a haplotype, but it is flexible to allow for any arbitrary grouping. When used in read-group mode, all reads of the same read-group are assumed to share the same methylation profile, allowing the capture of sample or allele-specific methylation.

It is important to note that due to the memory requirement of the forward and backward algorithms used to learn parameters of an HMM, the methylome segmentation in pycoMeth is performed over genomic windows. The size of a window is defined as N, the number of unique CpG sites in the window. The memory requirement of the segmentation algorithm scales with complexity $O(NS)$. Since S scales linearly in dependence of N (Chapter 3.1.2) a simplified expression of complexity would be $O(N^2)$ (quadratic with window size). Therefore, a complete segmentation of an entire chromosome in a single call of the segmentation HMM is inefficient, and a windowed approach is performed instead.

3.1.1. Segmentation HMM

The segmentation HMM is specified by three components. First, a set of states $\mathcal{S}$, which represent the segments and are parameterized with a methylation rate which is learned from the data. Secondly, the transition probabilities T, which define the probability of moving through the states and to the end state, thus connecting states. Third, the emission likelihood function L, which describes the likelihood of observing methylation calls for a given segment.

The following sections describe these three components in more detail.

States $\mathcal{S}$

The set of states $\mathcal{S}$ represents the sequence of segments, with state s representing the s^{th} segment. The number of states $S = |\mathcal{S}|$ therefore defines the maximum number of segments in the window. Each state is parameterized with methylation rates $\mu_{s,r}$, where s refers to the state and r the read.

In read-group mode pycoMeth can be provided with a read-group identifier (such as "sample" or "haplotype") which it will then find in the MetH5 input file (Chapter 2).

A read-group can refer to any read annotation stored in the MetH5 file, and would typically correspond to a biological sample or a haplotype. If a read-group identifier is provided, pycoMeth assumes homogeneous methylation patterns for all reads in the same read-group. This is modeled by interpreting $\mu_{s,r}$ as the methylation rate for read-group r and all reads within a read-group share the same parameters.

In order to simplify the language, the following sections refer to r in the context of $\mu_{s,r}$ as the read r (w.l.o.g.).

Transition probabilities T

Transition probabilities are defined such that a state transition can only occur one state forward or to the end state. The three transition probabilities t_{stay}, t_{move}, and t_{end} are hyperparameters of the model such that $1 = t_{\text{stay}} + t_{\text{move}} + t_{\text{end}}$.

$$t_{i,j} = \begin{cases} t_{stay} & \text{if } i = j < S \\ t_{move} & \text{if } i = j - 1 < S \\ t_{end} & \text{if } i < j - 1 = S - 1 \\ t_{move} + t_{end} & \text{if } i = j - 1 = S - 1 \\ 1 & \text{if } i = j = S \\ 0 & \text{else} \end{cases} \tag{3.1}$$

An example transition table where $S = 4$ would look like

$$T = \begin{bmatrix} t_{stay} & t_{move} & 0 & t_{end} \\ 0 & t_{stay} & t_{move} & t_{end} \\ 0 & 0 & t_{stay} & t_{move} + t_{end} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

Emission likelihoods L

Let $\text{In}_{i,s}$ be the logical predicate that CpG-site i belongs to segment s, and let $M_{i,r}$ be the logical predicate that CpG-site i on read r is truly methylated. I then model methylation rates within a segment as Bernoulli distributed with $\mu_{s,r}$ being the methylation rate of a read in a segment:

$$P(M_{i,r}|In_{i,s}) = \text{Bernoulli}(M_{i,r}|\mu_{s,r})$$

Next, to include methylation calling uncertainties, we require a well calibrated posterior methylation probability $P(M_{i,r}|X_{i,r})$, where that $X_{i,r}$ is the measured Nanopore

raw signal for CpG-site i on read r. Deep learning based methylation callers emit this probability directly, but may not be calibrated well. The Bayesian methylation caller Nanopolish emits well calibrated LLRs (Equation 1.7), from which the posterior may then be derived as such:

$$\begin{aligned}\mathrm{LLR}_{i,r} = \log \frac{P(X_{i,r}|M_{i,r})}{P(X_{i,r}|\neg M_{i,r})} &= \log \frac{P(M_{i,r}|X_{i,r})P(M_{i,r})P(X_{i,r})}{P(\neg M_{i,r}|X_{i,r})P(\neg M_{i,r})P(X_{i,r})} \\ &= \log \frac{P(M_{i,r}|X_{i,r})}{1-P(M_{i,r}|X_{i,r})} + \log \frac{P(M_{i,r})}{1-P(M_{i,r})}\end{aligned}$$

where the second term is the prior methylation probability. Since Nanopolish assumes an uninformative prior, the second term evaluates to 0

$$= \log \frac{P(M_{i,r}|X_{i,r})}{1-P(M_{i,r}|X_{i,r})} \tag{3.3}$$

which is the logit function, also known as the inverse sigmoid function

$$= \sigma^{-1}(P(M_{i,r}|X_{i,r}))$$

and therefore

$$\begin{aligned}P(M_{i,r}|X_{i,r}) &= \sigma(\mathrm{LLR}_{i,r}) \\ &= \frac{1}{1+\exp(-\mathrm{LLR}_{i,r})}\end{aligned} \tag{3.4}$$

The emission likelihoods in the HMM are then the likelihood of observing Nanopore raw signal $X_{i,r}$ given that $\mathrm{In}_{i,s}$:

$$\begin{aligned}P(X_{i,r}|\mathrm{In}_{i,s}) &= P(X_{i,r}|M_{i,r})P(M_{i,r}|\mathrm{In}_{i,s}) + P(X|\neg M_{i,r})P(\neg M_{i,r}|\mathrm{In}_{i,s}) \\ &= P(X_{i,r}|M_{i,r})\mu_{s,r} + P(X_{i,r}|\neg M_{i,r})(1-\mu_{s,r}) \\ &= \frac{P(M_{i,r}|X_{i,r})P(X_{i,r})\mu_{s,r}}{P(M_{i,r})} + \frac{P(\neg M_{i,r}|X_{i,r})P(X_{i,r})(1-\mu_{s,r})}{P(\neg M_{i,r})} \\ &= P(X_{i,r})\left(\frac{\sigma(\mathrm{LLR}_{i,r})\mu_{s,r}}{P(M_{i,r})} + \frac{(1-\sigma(\mathrm{LLR}_{i,r}))(1-\mu_{s,r})}{1-P(M_{i,r})}\right)\end{aligned}$$

if a uniform prior, that is $P(M_{i,r}) = \frac{1}{2}$ is chosen, this can be further simplified to

$$= 2P(X_{i,r})(\sigma(\mathrm{LLR}_{i,r})\mu_{s,r} + (1-\sigma(\mathrm{LLR}_{i,r}))(1-\mu_{s,r}))$$

For the purpose of likelihood maximization in the HMM, terms that are independent of segment assignment can be discarded.

$$L(X_{i,r}|\mathrm{In}_{i,s}) \sim \sigma(\mathrm{LLR}_{i,r})\mu_{s,r} + (1-\sigma(\mathrm{LLR}_{i,r}))(1-\mu_{s,r})$$

Segmentation

Segmentation is then performed using an optimized implementation of the Baum-Welch algorithm (Chapter 1.2.1). First, the posterior probabilities $\log P(\mathrm{In}_{i,s}|X, \mu^{(j)})$ of CpG site i being in segment s using segment parameters μ in iteration j are computed. This requires the computation of the intermediates of both the forward and the backward algorithms(Equation 1.4). This computation is performed while taking advantage of the sparse transition matrix, simplifying Equation 1.3 from

$$f_{n+1,p} := \sum_{p' \in S} f_{n,p'} t_{p,p'} L(x_{n+1}|p)$$

to

$$f_{n+1,p} := L(x_{n+1}|p)(f_{n,p} t_{\mathrm{stay}} + f_{n,p+1} t_{\mathrm{move}} + f_{n,N} t_{\mathrm{end}})$$

thus reducing the computational complexity from $O(NS^2)$ to $O(NS)$.

Having computed $P(\mathrm{In}_{i,s}|X, \mu^{(j)})$ using the forward and backward algorithms, methylation rate parameters are then updated using the maximum likelihood estimator.

$$\mu_{s,r}^{(j+1)} = \arg\max_{\mu_{s,r}} \sum_i L(X_{i,r}|\mathrm{In}_{i,s}, \mu_{s,r}) P(\mathrm{In}_{i,s}|X, \mu^{(j)})$$

which does not have a closed form solution and is therefore maxmimized numerically using a sequential least squares programming algorithm implemented in SciPy [196].

All computations are performed in log space in order to improve performance and increase numerical stability, taking advantage of highly optimized implementations of $\mathrm{lse}(x, y) = \log(e^x + e^y)$ and $\mathrm{log1m}(x) = \log(1 - e^x)$ in NumPy [198].

When segmenting multiple windows, pycoMeth with produce artificial segment breaks between windows. The output file will then contain an additional column, informing whether a segment is the start or end segment of a window, such that downstream processing may decide how to handle this boundary condition.

3.1.2. Hyperparameters

The segmentation model comes with a number of hyperparameters which require closer consideration. The hyperparameters are:

- Transition probabilities t_{stay}, t_{move}, t_{end}
- Window size N
- Maximum number of segments S

All hyperparameters are related to segmentation granularity. They can be categorized into two groups: the ratio $\frac{N}{S}$ defines the upper bound to the segmentation

granularity, while the transition probabilities t affect the ability to go below the upper bound. The latter is primarily controlled by t_{end}, which describes the prior probability of skipping all following segments and choosing to move to the last segment. The difference between t_{stay} and t_{move}, if taken independent of the other hyperparameters, theoretically describe the prior belief of segmentation granularity. However, when conditioned on $\frac{N}{S}$ and t_{end} it mainly affects the magnitude of likelihoods in the Baum-Welch algorithm. Therefore, the most important hyperparameters to tune are N and S. Memory requirement in the Baum-Welch algorithm scales at $O(NS)$, as NS is the dimensionality of the forward matrix F and backward matrix B (Chapter 1.2.1). Let $r = \frac{N}{S}$ be the desired segmentation granularity, then it is recommended to select $S = \frac{N}{r}$ and choose window size N depending on the amount of memory available.

Since the model is allowed to go below segmentation granularity r, and since an oversegmentation is typically preferred to an undersegmentation, it is generally preferable to err on the side of choosing a higher value for r. In experiments I find the following defaults reasonable: $N = 600$, $M = 20$. Default values in pycoMeth are set to a smaller value in order to be more conservative about memory usage: $N = 300$, $M = 10$, which yields the same granularity r but computed over smaller windows. When possible, however, that larger window sizes are preferred, in order to reduce the number of additional segments produced by the boundary condition.

Defaults for transition probability were tuned manually on random sections of a human methylome, using the evaluation dataset described in Chapter 3.2.3 on different chromosomes than the ones presented in the benchmark. The values were set to $t_{move} = 0.8$, $t_{stay} = 0.1$, $t_{end} = 0.1$, which are hardcoded in pycoMeth and used in all following experiments.

3.1.3. Evaluation

I evaluated pycoMeth on two different datasets, a simulated dataset and a real Nanopore methylation dataset. Since real methylation data does not contain ground truth segments, I first describe the evaluation on simulated data. Chapter 3.2.3 then covers the benchmarking of the entire differential methylation calling pipeline on real data and also illuminates the effect of segmentation on DMR testing.

Simulated evaluation dataset

I created a simulated 2-sample dataset based on simulated WBGS data produced from OmicsSIMLA [218]. OmicsSIMLA is a software package for multi-omics simulation and includes a module for the simulation of WGBS experiments. This module comes with a variety of methylation profiles learned from experimental data of different tissue types. Methylation profiles can then be randomly perturbed in order to simulate case and control samples with differential methylation. Both the case

and the control profiles can then be used to produce concrete methylation count matrices. OmicsSIMLA will then randomly drawing methylation counts for each CpG site in a segment, based on the segment's methylation rate.

Using OmicsSIMLA, I was able to simulate a methylome for a single chromosome learned from human liver tissue. For the control sample, OmicsSIMLA first generates segments learned in training, with each segment having a methylation rate assigned. For the case sample, I used OmicsSIMLA to generate a perturbed methylation profile, with up to 10% of segments differentially methylated with varying effect-sizes (0.15, 0.20, ..., 0.6). These two methylomes formed the ground truth for the benchmark. Next, I used OmicsSIMLA to generate count matrices from both methylation profiles, with about 30x coverage for each sample, thus creating a simulated WGBS experiment.

In order to arrive at a simulated Nanopore experiment, with simulated Nanopolish methylation calls, I then developed a Nanopolish methylation call simulator. This simulator consists of two parts: read simulation, and LLR simulation. The read simulation component draws a genomic coordinate from a uniform distribution and then a log10 read length from a Gaussian mixture model. The parameters of the Gaussian mixture model were estimated from a real Nanopore experiment with an N50 read length of about 50,000. The three kernels were learned as (in log10 space) $\mu = (2.88, 3.8, 4.75), \sigma^2 = (0.38, 0.48, 0.29)$ and weights as $w = (0.24, 0.63, 0.13)$.

The LLRs were simulated from the same real Nanopore data, by first converting LLRs to a posterior probability of methylation (Equation 3.4) and then to methylation calling certainty $C = 2|\sigma(\text{LLR}) - 0.5|$. I then fit a beta distribution in the observed methylation call certainties, which resulted in distribution $\alpha = 0.64031, \beta = 0.2088$ (Figure 3.2).

Using these parameters I then created two simulated datasets, a high coverage example with approximately 30x coverage in both samples and a low coverage example with approximately 15x coverage.

Preparing MethCP and methylKit for benchmark

Since pycoMeth is the first methylation segmentation and differential methylation testing software suite for Nanopore methylation calls, I decided to perform a benchmark which compares to existing software for methylation calls from WGBS. This benchmark therefore evaluates the performance of pycoMeth compared to a pipeline where Nanopore methylation calls are binarized and reduced to counts per CpG sites as they would be assayed in a short-read experiment.

Furthermore, recall that two important features of pycoMeth segmentations are (i) that it performs a consensus segmentation over multiple methylomes and (ii) that the segmentation is not biased towards differential methylation and will detect changepoints regardless of the presence of DMRs. Since I was not able to find any

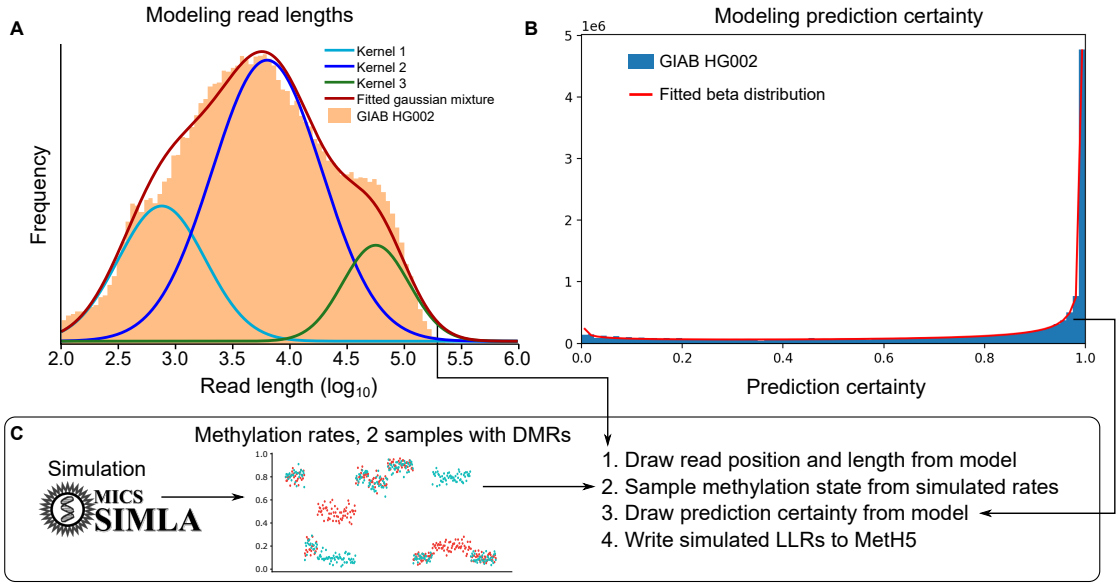

Figure 3.2.: The steps of creating a simulated 2-samples Nanopore methylation dataset with differential methylation. **A)** Modeling of read length distribution in $\log_{10}$ space using a Gaussian mixture model. **B)** Modeling of Nanopolish methylation call certainty distribution as a beta distribution. **C)** Simulation of a 2-sample Nanopore methylation experiment by simulating methylation segments using OmicsSIMLA [218]. Simulated methylation calls are then drawn from methylation rates, read length model, and Nanopolish prediction certainty distribution.

other software for WGBS which fulfills both of these features, I chose to benchmark against two widely used tools which each fulfills one of them.

The first tool is methylKit [165], an R [164] library for the analysis of methylation data from WGBS. The second tool is MethCP [162], also implemented in R, which performs a 2-sample consensus segmentation on the basis of differential methylation. Both tools are described in detail in Chapter 1.2.3.

In order to create pseudo-WGBS from Nanopore data for the use in those tools, I first binarize all methylation calls by thresholding LLRs using the threshold 2.0 as recommended by the Nanopolish developers (Figure 3.3A)

$$\mathrm{T(LLR)} = \begin{cases} 1 & \text{, if LLR} > 2.0 \\ 0 & \text{, else} \end{cases} \tag{3.5}$$

where $T(\mathrm{LLR}) = 1$ refers to a methylated site, $T(-\mathrm{LLR}) = 1$ refers to an unmethylated site and $T(\mathrm{LLR}) = T(-\mathrm{LLR}) = 0$ refers to a discarded call due to high uncertainty.

For methylKit, the total number of methylated and unmethylated reads from both the case and control samples for each CpG are added up and stored as a single methylation profile. The data is then loaded following the methylKit documentation and the function `methSeg` is called with parameters `maxInt=100, minSeg=10` as in the vignettes published with methylKit.

For MethCP, the methylated and unmethylated reads from the case and control sample are stored separately and are then loaded as two profiles following the MethCP documentation. The function `calcLociStat` then computes CpG-level differential methylation and `segmentMethCP` is then run to compute a segmentation.

Preparing pycoMeth segmentation for benchmark

For pycoMeth, I created two separate segmentations in order to compare the effect of two different sets of hyperparameters affecting segmentation granularity. The first segmentation, henceforth referred to as "pycoMeth", has been performed with window size $N = 300$ and maximum number of segments $M = 16$, thus a segmentation granularity of $r = 18.75$. The second segmentation, henceforth referred to as "pycoMeth coarse", has been performed with window size $N = 600$ and maximum number of segments $M = 16$. It presents a less fine-grained segmentation with segmentation granularity $r = 37.5$.

Evaluating segmentations on simulated data

The two main segmentation errors are oversegmentation and undersegmentation. Oversegmentation refers to placement of additional changepoints that do not corre-

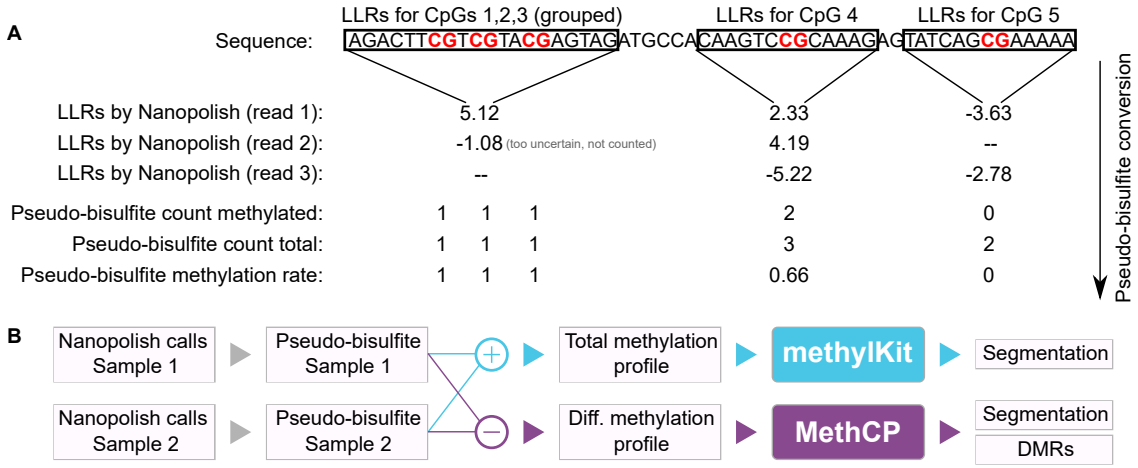

Figure 3.3.: Generating comparison data for segmentation benchmarking. **A)** Conversion of Nanopolish methylation calls to pseudo-bisulfite sequencing counts. Methylation calls are binarized (Equation 3.5) and then counted per CpG-sites. Grouped calls by Nanopolish are expanded, since bisulfite sequencing data contains CpG-level calls. **B)** Methylation profiles of two samples are first converted to pseudo-bisulfite data and then aggregated. For methylKit the aggregation is a sum of counts, whereas MethCP creates a differential profile (Chapter 1.2.3). Both tools then create an instance segmentation, while MethCP additionally calls DMRs (used in Chapter 3.2.3).

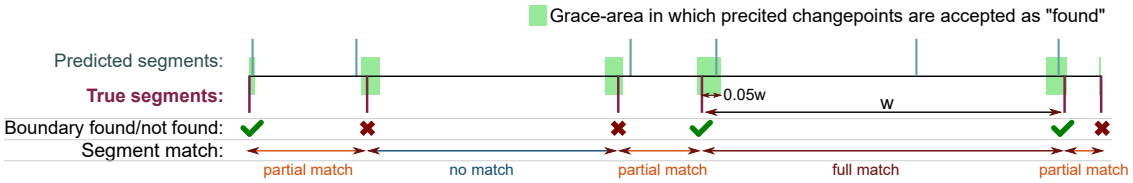

Figure 3.4.: Method used to evaluate undersegmentation. True segments are classified as no match, partial match, or full match, depending on whether start- and end-boundary are found. Whether or not a boundary is found is determined by whether or not a predicted boundary is within 5% of its containing segment's width from the true boundary. Also, note that all predicted changepoints not contributing to a match are reported as oversegmentation. In this example with 6 true changepoints, 3 matched changepoints, and 3 additional unmatched changepoints, it would be evaluated as 50% detection power and 50% oversegmentation. This is despite the fact that the total number of predicted changepoints matches the number of true changepoints.

spond to true segment boundaries. Undersegmentation refers to missing true segment boundaries, that is the merging of two neighboring segments. Both types of segmentation errors negatively affect DMR calling in different ways.

Oversegmentation results in more and smaller segments to be tested. As a result, test power suffers since fewer data-points per segment contribute to the test (smaller segments), and the increase in the number of tests results in a multiple testing burden. Oversegmentation can – to an extent – be fixed in post-processing, and a common strategy in computer vision is to oversegment and then merge [219]. Undersegmentation, on the other hand, leads to smaller effect-sizes, seeing how DMR effects are diluted. Furthermore, if the test assumes a certain distribution within a segment (e.g. a Normal distribution) this assumption may be violated if neighboring segments with different distribution parameters are combined. This can lead to DMRs being missed or filtered out for having too low observed effect-sizes. For these reasons, oversegmentation is typically preferred to undersegmentation, but the two types of errors must be balanced.

In this first segmentation evaluation, I focused on investigating the missed segment boundaries on the high coverage simulated data, in order to evaluate the degree of undersegmentation. To do so, I classified each true segment boundary as either found or missed by a particular method, depending on the distance of the nearest predicted segment boundary. The criteria I chose is that a segment boundary is considered found if the nearest predicted segment boundary is no more than 5% of the containing segment size. I then classify each true segment based on whether none, one, or both of its were boundaries found (no match, partial match, or full match, respectively, Figure 3.4).

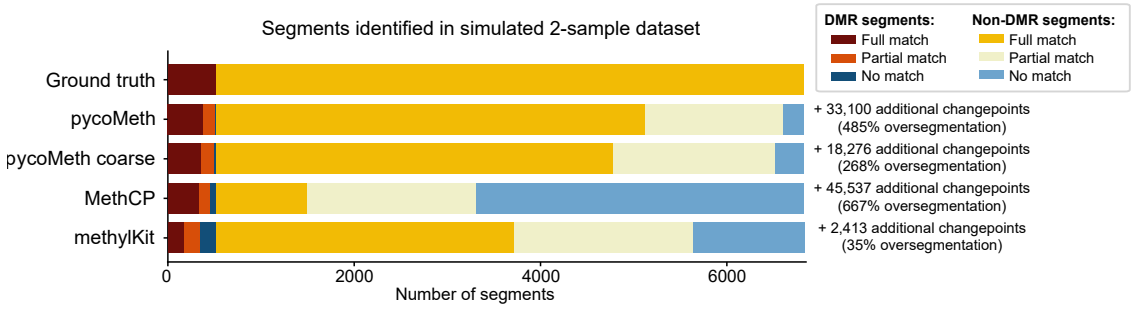

Figure 3.5.: Segmentation performance on high coverage simulated 2-sample data. For each tool, showing the number of segments discovered in full (full match - both start and endpoint discovered), partially (partial match - either start or endpoint discovered), or missed (no match - neither start nor endpoint discovered). pycoMeth coarse appears to provide a good compromise between oversegmentation and changepoint discovery power. Figure adapted from [207].

Figure 3.5 shows that the pycoMeth segmentation performs best in the discovery of both segments that are DMRs as well as non-DMR segments. More concretely, 72.2% and 73% of segments were fully matched for DMR and non-DMR segments, respectively, and 24.1% and 23.5% partially matched. Unsurprisingly, MethCP has difficulty detecting non-DMR segments, seeing how it operates on the differential methylation profile. A total of 55.8% of non-DMR segments yielded no match by MethCP, while 63.41% of DMR segments were fully matched and 22.4% of segments partially. Both pycoMeth and MethCP have a high degree of oversegmentation (485% and 667%, respectively). The methylKit segmentation, on the other hand, appears to be largely undersegmented, with 34.3% and 18.9% of DMR and non-DMR segments not matched, respectively, and with only 35% oversegmentation.

The pycoMeth coarse segmentation reduces oversegmentation to 268%, which is a 217 percentage point (pp) reduction compared to the pycoMeth segmentation. At the same time it reduces segment discovery by only 3.4pp, suggesting a better compromise between over- and undersegmentation. In order to evaluate whether segmentations perform better than random, I further permuted the order of segments and evaluated the distance from a discovered segment to the nearest ground truth segment (a measure of oversegmentation) and the distance from a ground truth segment to the nearest discovered segment (a measure of undersegmentation). All tools showed better performance than random in both categories, with the exception of MethCP, which unsurprisingly oversegments worse than random due to its inability to properly segment non-DMR regions (Appendix Figure C.5).

Finally, I repeated the benchmark on the low coverage example, showing that pycoMeth and pycoMeth coarse remained nearly unaffected (less than 1pp reduction), whereas MethCP shows a 1.5pp and methylKit a 6.8pp reduction in segment bound-

ary discovery(Appendix Figure C.3A).

3.2. Differential methylation calling

The differential methylation calling module in pycoMeth takes genomic segments as an input, together with methylation calls in MetH5 format, and then tests each segment for differential methylation between two or more read-groups. Instead of using the read-group annotation stored in a MetH5 container, pycoMeth may also be provided with multiple MetH5 files, in which case it will treat the reads of each file as a separate read-group. This functionality can also be used for the analysis of ASM, when haplotype assignment from a read phasing method is stored as read-groups.

In literature, the landscape of statistical tests used for differential methylation testing is diverse. The most commonly used test for WGBS data is probably the Fisher-Exact test [48, 165, 220, 221], but also logistic regression based tests [165] and binomial based tests [222–224] have been proposed. These individual tests come with different null hypotheses and assumptions.

In pycoMeth, I therefore decided to offer a variety of different statistical tests, based on a selection of test hypotheses. The selection of the test hypothesis can be performed using the command line argument `--hypothesis`. For example, the LLR based test has the null hypothesis that the mean sample LLR of all samples is identical. This test is accessible via the hypothesis `llr_diff`.

I further expanded pycoMeth by implementing two more test hypotheses. First, in order to support the widely adopted Fisher-Exact test, I implemented the `count_dependency` hypothesis, which performs a Fisher-Exact test in the 2-read-group case and a χ^2 test on a contingency matrix for more than 2 read-groups. Secondly, in order to provide a more conservative test, I implemented the `bs_diff` hypothesis. The null hypothesis for these tests is that the mean read-level methylation β-score per sample is identical.

Finally, I expanded the differential methylation testing module by implementing additional 1-vs-all *post hoc* tests. If the null hypothesis in an $n > 2$-read-group test can be rejected, a *post hoc* test is used to determine which, if any, one of the n read-groups is different from the other $n - 1$. Note that this does not affect the choice of regions reported as DMRs in the n-read-group test. Instead, it adds an additional column to the output file, containing the list of read-groups for which a *post hoc* test revealed significant difference from the rest. The statistical test used for the post-hoc test also follows the test hypothesis selection, using the appropriate 2-read-group version of the orignal $n > 2$-read-group test. For example, if the user chose the test hypothesis `count_dependency` a χ^2 test would be performed for the initial $n > 2$-read-group test, followed by a Fisher-Exact test for *post-hoc* testing.

3.2.1. Hypotheses and tests

The following sections detail the three test hypotheses currently implemented in pycoMeth, including the test selection and how the data for the test is computed. Each test produces a test statistic and a p-value, which is then used to determine whether a segment is a DMR.

LLR difference hypothesis

For a segment s to be tested, let $i \in s$ be the individual CpG sites in s. Let R be the set of read-groups and $r \in R$, $j \in r$ refer to the reads j in a read-group r. As previously used, let $\text{LLR}_{i,j}$ refer to the LLR of CpG-site i in read j. Let $\text{LLR}^{r,s} = \{\text{LLR}_{i,j} : i \in s, j \in r\}$ be the set of all LLRs in reads of read-group r and CpG-sites in segment s. As previously described, a Mann-Whitney-U test [225] on LLRs is used in the 2-read-group case and a Kruskal-Wallis [226] test for more than 2 read-groups.

Then the null-hypothesis for the `llr_diff` mode is

$$\mathcal{H}_0 := \forall_{r \in R, r' \in R}\ \text{Median}(\text{LLR}^{r,s}) \sim \text{Median}(\text{LLR}^{r',s})$$

β-score difference hypothesis

With the `bs_diff` option I implemented the test hypothesis that read-level methylation rates (β-score, Equation 1.5) in a segment differ between read-groups. Like with the `llr_diff` hypothesis, pycoMeth will choose a Mann-Whitney-U test [225] in the 2-read-group case and a Kruskal-Wallis [226] test for more than 2 read-groups. The test is then performed on the read-level β-scores for each read-group.

Let β_j^s be the read-level methylation β-score for read j in segment s

$$\beta_j^s = \frac{\sum_{i \in s} T(\text{LLR}_{i,j})}{\sum_{i \in s} T(\text{LLR}_{i,j}) + T(-\text{LLR}_{i,j})}$$

where T is defined in Equation 3.5. Let $\beta^{r,s} = \{\beta_j^s : j \in r\}$, the set of all read-level β-scores in a read-group, then the null-hypothesis for the `bs_diff` mode is

$$\mathcal{H}_0 := \forall_{r \in R, r' \in R}\ \text{Median}(\beta^{r,s}) \sim \text{Median}(\beta^{r',s})$$

Methylation rate dependency hypothesis

To generalize the commonly used Fisher-Exact test for a 2 or more read-group case, I implemented the `count_dependency` hypothesis. This implements the null-hypothesis that the probability of methylation for any CpG site in any read in one read-group is equal to the overall methylation probability across all read-groups. In order words, the null-hypothesis states that the methylation rate is independent

of the read-group. Recalling that $M_{i,j}$ was defined as the logical predicate that CpG-site i in read j is methylated, then the null-hypothesis can be defined as:

$$\mathcal{H}_0 := \forall_{r \in R, r' \in R}\ P(M_{i,j}|i \in s) = P(M_{i,j}|j \in r', i \in s)$$

To do so, pycoMeth computes a contingency matrix

$$Z = \begin{bmatrix} Z_0^M & \dots & Z_n^M \\ Z_0^U & \dots & Z_n^U \end{bmatrix}$$
$$Z_r^M = \sum_{i \in s, j \in r} T(LLR_{i,j})$$
$$Z_r^U = \sum_{i \in s, j \in r} T(-LLR_{i,j})$$

and then performs a Fisher-Exact [227] test on the contingency matrix in the 2-read-group case, or a χ^2-test [228] for more than 2 read-groups.

3.2.2. Adjustment for multiple testing

When performing multiple comparisons where a statistical test is subjected to a p-value threshold, each negative case (in our case, each non-DMR) has a chance to produce a false discovery if the null-hypothesis is rejected. When false positives are more likely than false negatives, particularly in an unbalanced dataset where the number of negative cases is larger than the number of positive cases, this imbalance will lead to an inflated false discovery rate (FDR). A commonly used remedy is the adjustment of p-values, and many schemes for p-value adjustment have been proposed [229]. Examples include Bonferroni [230], Sidak [231], Benjamini-Hochberg [232], and Storey's q-value [233]. In pycoMeth, multiple testing correction is performed after DMR testing, and raw p-value as well as adjusted p-value are stored in the output file. Methods for p-value adjustment are provided by the python statsmodels package [214].

To replace the effect-size filter in the pycoMeth prototype, I instead implemented an independent hypothesis weighting (IHW) scheme. This method allows to use a covariate independent of the p-value under the null-hypothesis, which is then used to weight p-values before adjustment [217]. In pycoMeth the covariate I chose was the pooled variance of all read-level β-scores in the segment. This covariate is informative of test power (a segment with low variance is less likely to contain methylation differences) yet independent of p-value (pooled variance can be confounded by between-sample or within-sample variance). Since weights must be positive and the average must be 1 [217], they are re-centered and then re-scaled.

$$
\begin{aligned}
\text{Average segment methylation rate: } \beta^s &= \frac{1}{\sum_{r \in R} |r|} \sum_{r \in R} \sum_{j \in r} \beta^{r,s} \\
\text{Unscaled weight: } v_s &= \frac{1}{\sum_{r \in R} |r|} \sum_{r \in R} \sum_{j \in R} (\beta^{r,s} - \beta^s)^2 \\
\text{Mean weight: } \bar{v} &= \frac{1}{S} \sum_s v_s \\
\text{Re-scaled and re-centered weight: } w_s &= 1 + \frac{v_s - \bar{v}}{\min_{s'}(v_{s'} - \bar{v})}
\end{aligned}
$$

This yields a weight w_s scaled to the range $[0, 2]$, where weight 0 corresponds to the highest variance and weight 2 to the lowest variance, with the arithmetic mean of all weights being $\bar{w} = 1$.

3.2.3. Evaluation

DMR testing, as well as the effect of segmentation on the DMR calling pipeline, was first evaluated on the simulated data described in Chapter 3.1.3. I performed pycoMeth DMR testing on all four segmentations (pycoMeth, pycoMeth coarse, methylKit, MethCP), as well as using all three test hypothesis, with and without IHW, for a total of 24 settings. Finally, I included DMRs called from the MethCP pipeline, bringing the number of settings to 25.

To correct for multiple testing within each setting I chose to use the Benjamini-Hochberg [232] procedure, with the exception of the MethCP setting, which uses a sliding linear model (SLIM) [234] instead (implemented within the software-suite).

DMR testing evaluation on simulated data

Seeing how each segmentation results in different regions to be tested, all comparisons were performed on a CpG-basis. To do so, I first classified each CpG site in the simulated data as either being a DMR CpG or a non-DMR CpG based on whether it is contained in a ground truth DMR segment. For each DMR calling setting, I repeated the same CpG classification, which allowed me to compute quality metrics such as precision, recall, and F1-score.

Let P be the number of DMR CpGs in ground truth, and TP be the number of true positive predicted DMR CpGs (CpGs that are both DMRs in the ground truth and the test), and let FP be the number of false positive predicted DMR CpGs (CpGs

that are predicted as DMRs in the test, but not the ground truth). Then

$$\begin{aligned} \text{Recall} &= \frac{TP}{P} \\ \text{Precision} &= \frac{TP}{TP + FP} \\ \text{F1-score} &= \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \end{aligned} \tag{3.6}$$

where recall is a metric of detection power, precision a metric of false discovery (precision = 1−FDR), and F1-score the harmonic mean of recall and precision, used to measure the balance between these two metrics.

When comparing different test hypotheses while leaving the test fixed, it appears the the test hypothesis does not have a great effect on DMR testing performance in this scenario Figure 3.6A. The overall best performing test was the `bs_difference` hypothesis without IHW (0.868 F1-score).

When comparing the effect of different segmentations on DMR-testing performance, while keeping the test hypothesis fixed, pycoMeth coarse yields the best overall performance (Figure 3.6B), while MethCP has best precision but reduced recall, yielding a lower F1-score than pycoMeth. Using Fisher-exact test as a comparison, as this is the same test that is underlying the MethCP segmentation, pycoMeth coarse has an F1-score of 0.865, whereas for MethCP an F1-score of 0.856 is obtained. As with the segmentation evaluation, I repeated the experiment with the low coverage simulated data, showing that all methods show reduced recall in lower coverage, while all but MethCP also show reduced precision. Still, pycoMeth coarse performed best overall with an F1-score of 0.85 (reduction of 0.015) while MethCP dropped to an F1-score of 0.84 (reduction of 0.016)(Appendix Figure C.3A).

DMR testing evaluation on real data

I performed a second benchmark on real Nanopore data obtained from the GIAB consortium [204], described in Chapter 2.4, in two scenarios: differential methylation between the two parents (HG003 and HG004) and ASM calling in the son (HG002), both evaluated on chromosome 20. Since all samples contain methylation calls with approximately 30x coverage, the parent comparison compares two 30x coverage samples as in the high coverage simulation benchmark, while the ASM calling compares two haplotypes with approximately 15x coverage, as in the low coverage simulation benchmark.

Across both comparisons, pycoMeth shows greater power in detecting low effect-size DMRs. Using the Fisher-Exact test with IHW for the comparison, the total number of DMR CpGs discovered is 115% greater in pycoMeth coarse compared to MethCP, with the majority being of effect-sizes 0.1-0.2 β-score difference (Figure 3.7).

Since there is no ground-truth available in these real world comparison, neither for

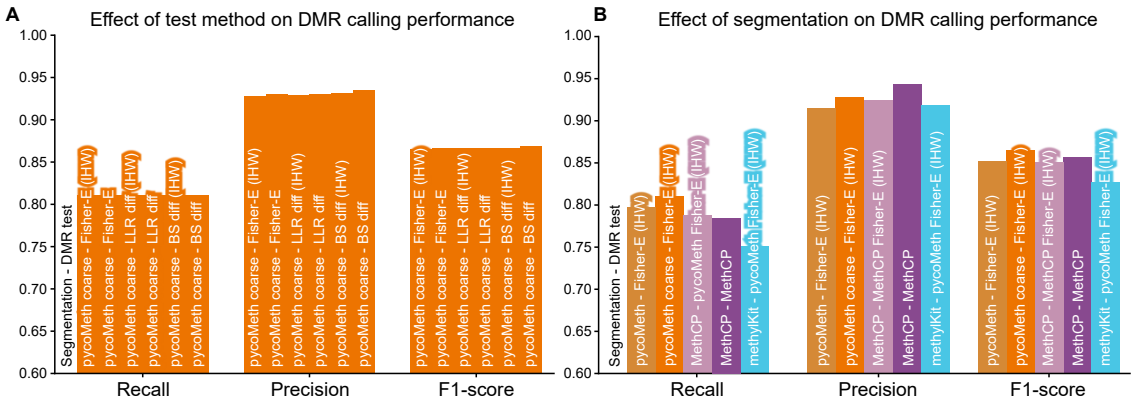

Figure 3.6.: Recall, precision and F1-score on simulated data based on CpG-classification (DMR/non-DMR, FDR < 0.05). All settings include Benjamini-Hochberg [232] multiple testing correction, with the exception of MethCP, which implements SLIM [234] . **A)** Comparison of different test hypothesis with a fixed segmentation (pycoMeth coarse). **B)** Comparison of different segmentations with a fixed test(Fisher-Exact with IHW). Figure adapted from [207].

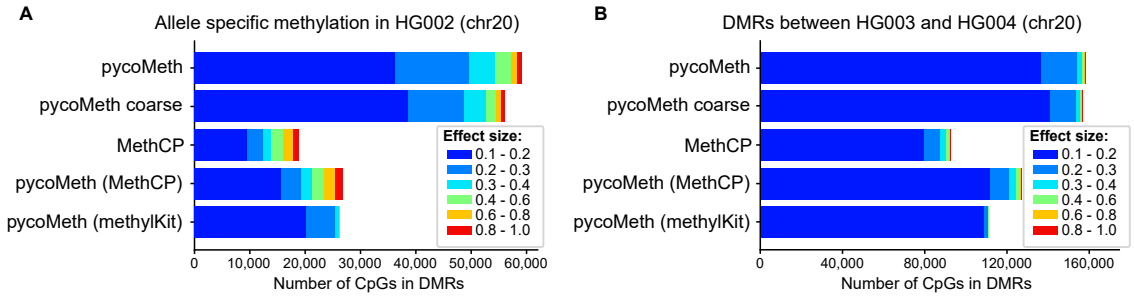

Figure 3.7.: Number of CpG sites in DMRs (FDR< 0.05) broken down by the containing segment's effect-size (β-score difference). All settings except for MethCP use the Fisher-Exact test with IHW and Benjamini-Hochberg [232] for multiple testing correction. MethCP implements a Fisher-Exact test with SLIM [234] **A)** DMR calling between parents HG003 and HG004. **B)** ASM calling son (HG002). Figure adapted from [207].

segmentation boundaries nor for DMRs, we cannot conclude whether the added power of pycoMeth in low effect-size DMR discovery is detecting real DMRs. An investigation of precision and recall in low effect-sizes in the simulated benchmark, however, suggests that pycoMeth coarse performs best or comparable even in low effect-size predictions (Appendix Figure C.6).

To get a better idea of the validity of our methods and results in the real world setting, I performed a permutation experiment to validate our observations. I created two randomized samples HG003_rand and HG004_rand, by randomizing the order of the elements in the `llr` dataset in the MetH5 container. This retained the same coverage and global methylation distribution per sample, yet making methylation entirely independent of read and CpG-site. I then performed segmentation and DMR calling using pycoMeth between the two samples and used CpGs inside the produced DMR segments as false discoveries in order to estimate an FDR. This compares the number of false discoveries with the number of real discoveries between the unperturbed samples. In order to account for global methylation differences between the two samples, an effect-size filter is required. Therefore, I vary the minimum effect-size threshold for accepting DMRs and then investigate FDR for different test settings. The `count_dependency` and `bs_diff` hypothesis both reach an FDR of 0.05 by thresholding effect-sizes to a minimum of 0.1, while the `llr_diff` hypothesis requires a more stringent threshold of 0.15. With increasing stringency to the effect-size threshold, the `bs_diff` hypothesis shows continously decreasing FDR, while the other two fluctuate, but remain in the range between 0.05 and 0.2 FDR. This indicates, that an effect-size threshold of 0.1 is sufficient to eliminate global effects in the HG003 vs HG004 comparison, giving confidence to the high number of low effect-size DMRs detected by pycoMeth.

Agreement in simulated and real data

In order to investigate agreement between prediction tools, I computed positive percent agreement (PPA) between pycoMeth, pycoMeth coarse, MethCP, and methylKit segmentation with pycoMeth DMR testing. If DMR_{tool} is the set of DMR CpGs identified by tool *tool*, then PPA between two tools is computed as:

$$PPA_{a,b} = 100\frac{DMR_a \cap DMR_b}{DMR_a \cup DMR_b}$$

I find that DMRs identified differ greatly between tools, with positive agreement between the two pycoMeth segmentations being higher than other pairs (79% in simulated data, 66% in GIAB data, Figure 3.9). Agreement in real data is overall lower than in simulated data, which can be explained by the overabundance of difficult to detect low effect-size DMRs. Overall, this analysis suggests high complementary of the methods and implies that different segmentation approaches may detect different kinds of DMRs.

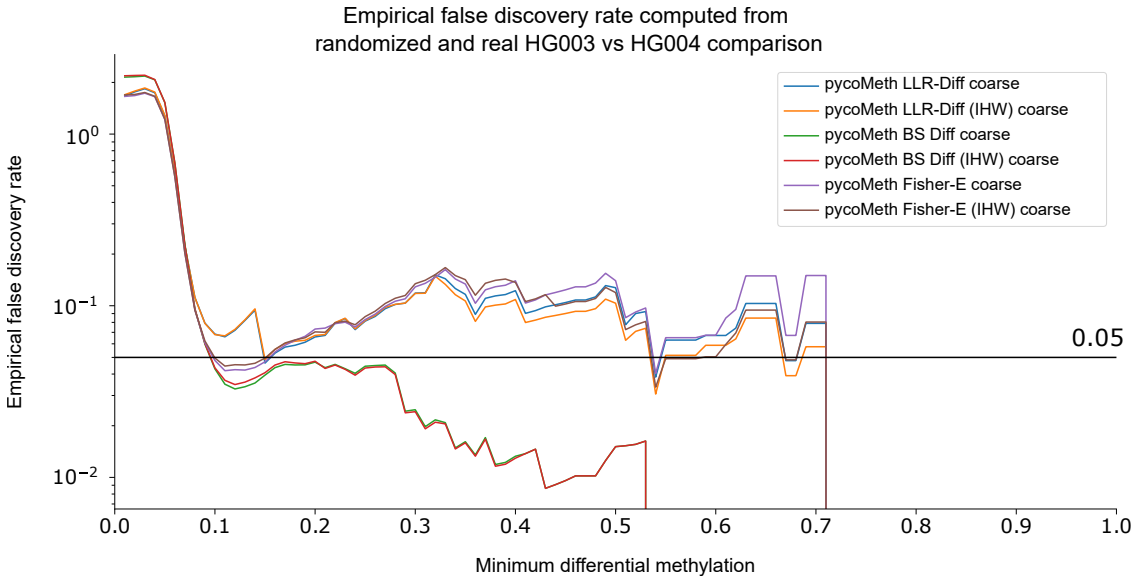

Figure 3.8.: FDR estimation from real and randomized GIAB parent comparison. Global differences between samples may produce low effect-size false discoveries, which is why an effect-size filter after DMR testing is necessary. FDR here is presented as ratio between predicted DMR CpGs in real and randomized experiment, with minimum differential methylation (β-score difference) used as a threshold. Figure adapted from the pycoMeth publication [207].

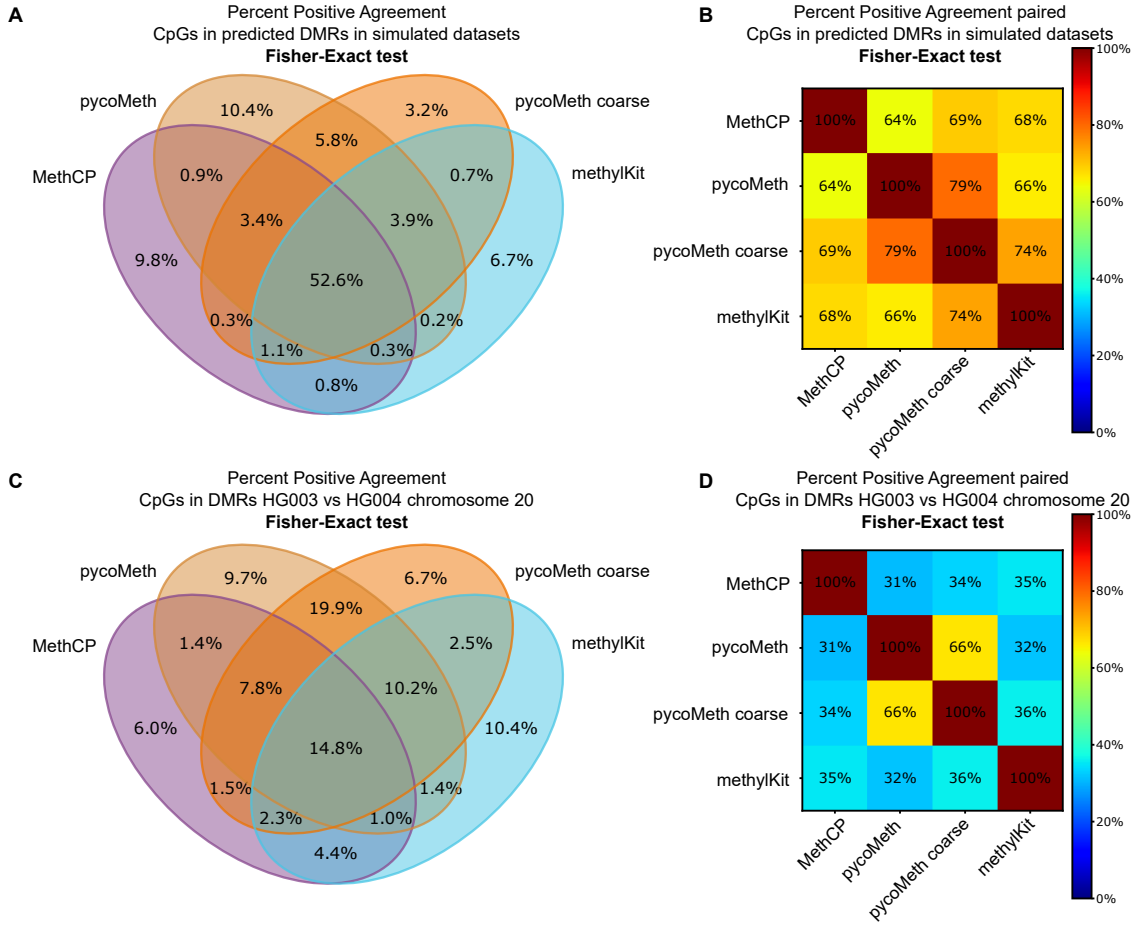

Figure 3.9.: Positive percent agreement between DMR test settings (FDR< 0.05). **A)** Agreement relative to the union of all 4 settings in the high coverage simulated dataset. **B)** Pairwise agreement in the high coverage simulated dataset. **C-D)** Positive agreement in the real data, GIAB parent comparison. Figure adapted from the pycoMeth publication [207].

3.3. Code availability

The pycoMeth python package [235] is under version control at GitHub, and distributed via package managers pypi and anaconda.

- GitHub: `https://github.com/PMBio/pycoMeth`
- pypi: `https://pypi.org/project/pycoMeth`
- anaconda: `https://anaconda.org/snajder-r/pycoMeth`

The segmentation and DMR testing benchmarking code is also available on GitHub under:

`https://github.com/snajder-r/benchmark_meth5`

3.4. Conclusion

With pycoMeth version 2, I implemented a versatile toolbox for methylome segmentation and differential methylation calling, specifically designed for Nanopore sequencing experiments. The segmentation algorithm in pycoMeth implements a Bayesian changepoint detection method, which considers methylation calling uncertainties. It allows for the consensus segmentation of multiple methylation profiles independent of the presence of differential methylation. The differential methylation testing module implements a variety of tests and chooses the right test depending on a user chosen test hypothesis and the number of read-groups. The pycoMeth toolbox takes advantage of the MetH5 format in order to perform operations on methylation data swiftly and in a parallelized fashion. MetH5 allows the distribution of segmentation tasks across multiple compute nodes, and pycoMeth implements CPU-level parallelization, two forms of concurrency that can also be combined.

The benchmark on simulated data shows that pycoMeth performs better than, or at least comparable to, existing tools designed for WGBS experiments. In real data, pycoMeth shows particularly strong power in the detection of low effect-size DMRs. Validation experiments verify that even in low-effect size DMRs pycoMeth performs with higher precision than other tools.

Segmentation granularity is an important factor and true granularity of methylation organization is difficult to estimate. This benchmark, suggests hyperparameters for segmentation granularity based on the performance on both simulated and real methylation data. Analysis of prediction agreement shows that changes to segmentation granularity do have an effect on the DMRs detected, but pycoMeth segmentations of varying DMRs still have stronger agreement than with other tools.

In conclusion, pycoMeth implements a user-friendly, efficient, and well-calibrated solution for Nanopore methylation data analysis.

4. Application on Medulloblastoma study

This chapter describes the application of pycoMeth segmentation and differential methylation calling on Nanopore sequencing data from a single patient with chromothriptic sonic hedgehog (SHH) medulloblastoma [187]. Medulloblastoma describes a malignant brain tumor located in the cerebellum, which primarily occurs in infants and young children [236]. The disease is typically categorized into four molecular subtypes, depending on histology, driver mutations, and pathway activations. These subtypes are associated with different treatment options and prognoses [236]. The WNT and SHH subtypes are named after the WNT and SHH signaling pathways activated in these medulloblastoma subtypes, while the Group 3 and Group 4 subtypes are of less clearly defined molecular composition [237, 238].

In this study, I analyzed the methylome of a pediatric SHH medulloblastoma patient from three samples: blood, primary tumor before treatment, and relapse tumor after treatment. The tumor genome of this patient had undergone massive genomic rearrangements, thought to have stemmed from one single catastrophic event. This type of rearrangement is referred to as chromothripsis or chromosome-shattering [239] and results in a highly complex genomic profile. This study shows the application of pycoMeth in a cancer setting, as well as the general utility of measuring DNA methylation on long reads in the presence of such genomic rearrangements.

4.1. Contributions

The study was designed by Aurélie Ernst (DKFZ), Oliver Stegle (DKFZ), Ewan Birney (EMBL-EBI), and Jan Korbel (EMBL). The samples were provided by Aurélie Ernst (DKFZ) and sequenced by Ewan Birney (EMBL-EBI). Genomic analyses, including the detection of germline and somatic variants, structural variants, read-phasing, and the discovery and assembly of chromothriptic events, were performed by Tobias Rausch (EMBL). Allele-specific expression (ASE) analysis was performed by Marc Jan Bonder (DKFZ). The contributions listed in this paragraph are limited to the work contributing to the results of this thesis. For the full author contributions to the medulloblastoma study, please refer to the author contribution section of the published manuscript [187].

4.2. Study design

In this study [187], three samples, blood, primary tumor, and relapse tumor, had been sequenced on a variety of platforms. Illumina short read sequencing data had been produced and used to infer germline (blood) and somatic (tumor minus blood) variants. Nanopore long-read sequencing data had been produced with a coverage of approximately 30x in primary tumor an 15x for the blood and relapse samples. Other data includes 450k methylation array data and Illumina RNA sequencing data for gene expression analysis.

Since this exploratory study is based on only a single patient, it is difficult to draw significant biological conclusions. However, it serves as a case study for cancer genome, epigenome, and transcriptome analyses enabled by long-read sequencing. One of the most striking discoveries in the tumor samples is the presence of a 1.55Mbp long chromothriptic structure composed of segments from chromosome 11 and chromosome 17, here called CS11-17. The other observation was a potentially novel type of genomic rearrangement, which was termed "templated insertion (TI) threads" over the course of this study. TI-threads are described as insertions copied from distal genomic templates [240], which were concatenated repeatedly into long repetitive threads. TI-threads were discovered and reconstructed in this sample using long Nanopore reads, and were subsequently also identified in publicly available short-read data from other cancer types.

To demonstrate the utility of Nanopore methylation analysis, I performed DNA methylation analyses using pycoMeth followed by further integrative downstream analyses. These analyses include (i) DMR calling between timepoints (tumor before and after relapse), (ii) ASM analysis in the main tumor, (iii) integration of ASE with ASM, as well as (iv) methylation analyses of chromothriptic structures CS11-17 and TI-threads.

4.2.1. Data preparation

For the pre-processing of Nanopore data, I implemented a Snakemake [241] pipeline [242] which performs the following steps: basecalling using guppy (version 6.1.7), filtering of basecalled reads based on read quality threshold [243], alignment to reference genome GRCh38 [205] using minimap2 [206], read phasing into haplotypes using WhatsHap [244], and methylation calling using Nanopolish [180]. The resulting methylation calls are then stored in MetH5 containers (one file per sample) and haplotype assignment from WhatsHap is then stored as read-groups in the MetH5 containers. Nanopore data from all three samples (blood, primary tumor, relapse tumor) were then processed using this pipeline, resulting in a total of 1.44 billion methylation calls over approximately 19 million reads.

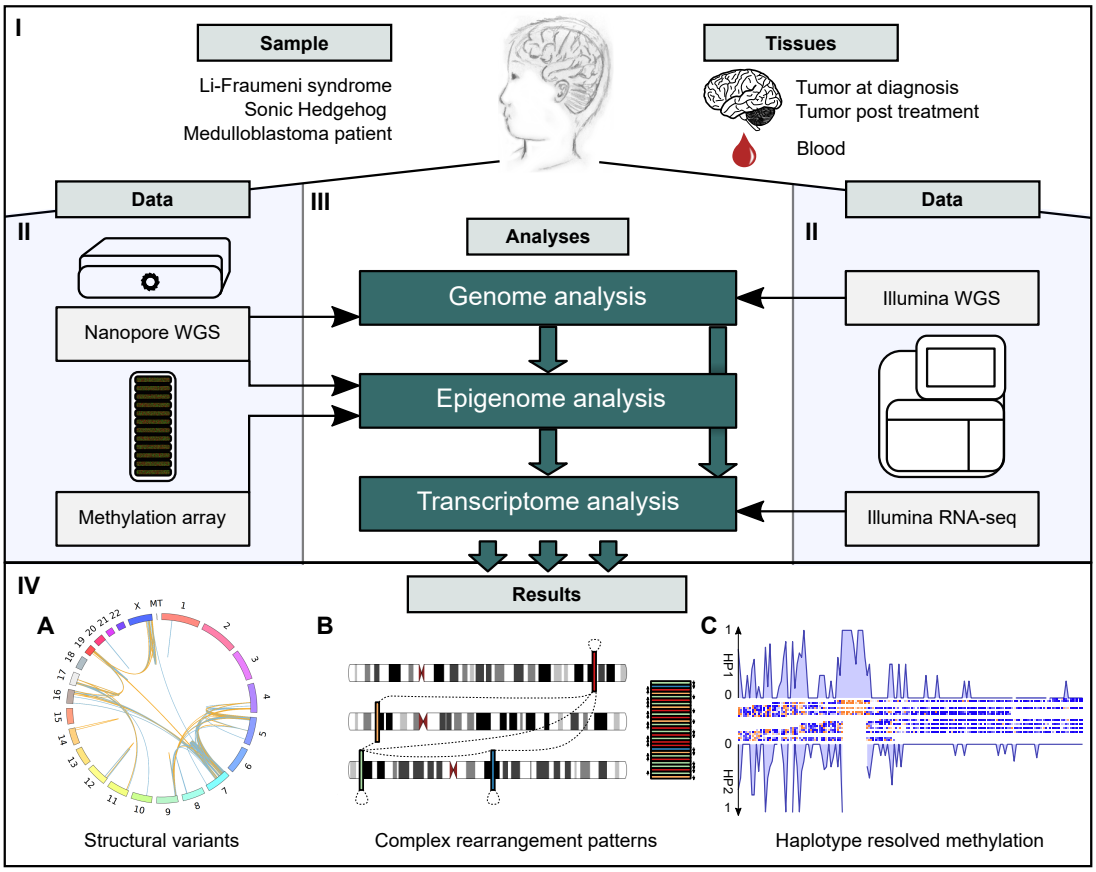

Figure 4.1.: Medulloblastoma study design. Three samples (blood, primary tumor, relapse tumor) from a single case of chromothriptic SHH medulloblastoma have been Nanopore (WGS), and Illumina (WGS & RNA-seq) sequenced and methylation array data was produced. A combined multi-omics analysis identified structural variants, complex rearrangements, and a comprehensive haplotype-resolved methylation analysis using methylation calls from Nanopore. Figure adapted from the medulloblastoma publication [187].

Segmentation and DMR calling using pycoMeth

In order to arrive at a single consistent segmentation for all tumor analyses, I took advantage of pycoMeth's ability to create a consensus segmentation from multiple methylation profiles. To do so, I provided pycoMeth with the MetH5 files from both the primary tumor as well as the relapse sample, including the haplotype assignment stored as read-groups in the MetH5 files. Unphased reads were treated as a third haplotype, therefore resulting in a consensus segmentation from 6 methylation profiles (2 samples with 3 haplotypes, each). The hyperparameters chosen were window size $N = 600$ and maximum number of segments per window $M = 16$, same as the "pycoMeth coarse" segmentation used in the pycoMeth benchmark (Chapter 3.1.3).

Since pycoMeth's Bayesian methylation segmentation algorithm takes methylation calling uncertainty into account, prefiltering of LLRs was not necessary. This allowed pycoMeth to also benefit from the 385 million methylation calls (36% of all methylation calls in the two tumor samples) in the LLR range $(-2.0, 2.0)$, which would have been filtered out during quality control in experiments which binarize methylation calls. Over all chromosomes, the segmentation yielded 443,244 segments over 22,887,157 unique CpG sites, resulting in an effective segmentation granularity of $\hat{r} = 51.6$ CpG sites per segment. This is coarser than the specified granularity $r = \frac{600}{16} = 37.5$, showing pycoMeth's ability to choose a number of segments lower than the defined maximum.

In addition to the *de novo* segmentation, I also applied pycoMeth's CGI finder to detect CGIs in the reference genome. For both the *de novo* segmentation as well as the putative CGIs, I then performed DMR testing using pycoMeth in two settings: differential methylation between the two tumor samples (between-sample variation), and ASM in each tumor sample (within-sample variation). In both settings, pycoMeth Meth_Comp was called with the `bs_diff` hypothesis. Benjamini-Hochberg for p-value adjustment for multiple testing was used together with IHW, and an adjusted p-value (FDR) threshold of 0.05 was used to determine statistical significance. DMRs were then further filtered based on effect-size. In order to be especially conservative in this single-patient study, only strong effects were considered, using an effect-size filter of 0.5 minimum β-score difference.

4.3. Differential DNA methylation between tumor samples detected by pycoMeth

The comparison of primary tumor before treatment and relapse tumor yielded a total of 1,785 segments as DMRs with effect-size greater than 0.5. These segments contained a total of 23,576 CpG sites (0.1% of sites tested). In order to gauge the benefit of using Nanopore sequencing compared to WGBS, I use the number of of CpG sites in low-complexity regions as a marker for discoverability in short-read sequencing experiments. Of all between-tumor DMR CpG-sites, over 34% fall in

such low-complexity regions indicated by being aligned to regions which are soft-masked in the reference genome.

I then assigned DMRs to genes by intersecting DMR regions with putative gene promoter regions. Putative gene promoter regions were defined as regions 2,000bps upstream to 500bps downstream of TSS annotated in the Ensembl [245] release 99. Differential methylation effect-size (β-score difference) was computed for each gene. For genes with multiple affected transcripts the DMR with the highest absolute effect-size was considered.

From the 23,576 DMR CpG-sites, a total of 2,921 (12.4%) were within gene promoter regions of 366 genes. Three of the 51 previously identified main medulloblastoma driver genes used as reference [238] were among the genes with promoter DMRs, namely PTCH1 (methylated in relapse), and SMARCA4 and GFI1B (both methylated in primary tumor). Upon manual inspection, the DMR identified in PTCH1 also perfectly intersected with a heterozygous deletion in both tumor samples, with the wild-type allele showing the differential methylation (Figure 4.2D). Another noteworthy example is a large DMR in the promoter and enhancer of NRN1, a gene previously associated with treatment resistance medulloblastoma [246] and which has previously been identified as a methylation marker of tumor growth in esophageal cancer [247]. Here, the NRN1 promoter appears to be methylated only in the pre-treatment primary tumor sample in both alleles (Figure 4.2C).

4.3.1. Functional analysis of DMRs

Since gene expression is typically negatively correlated with promoter methylation (Chapter 1.1.1), I then cross-referenced promoter DMRs with differential expression from Illumina RNA-seq data. Of the 366 genes with promoter DMRs, 321 were expressed in both samples. Of these, 41 also exhibited strong differential expression (absolute log fold change > 2). 33 (80.5%) of the 41 genes exhibited negative correlation between promoter methylation and gene expression, with a significant rank based correlation (Spearman R: -0.30, pvalue: 0.053, Appendix Figure C.7).

I then estimated copy number differences between primary tumor and relapse sample based on the Illumina short-read WGS data, in order to evaluate the effect of copy number variation on differential expression. I detected 74 differentially expressed genes with normalized copy number difference of at least 1. Within this set of 74 genes, I was able to observe a ranked based correlation comparable to that in the methylation analysis (Spearman R: 0.31, pvalue:0.0065). When considering all three factors (copy-number, methylation, and expression) partial correlation between differential methylation and differential expression is slightly improved (partial Spearman R: -0.33, pvalue: 0.037).

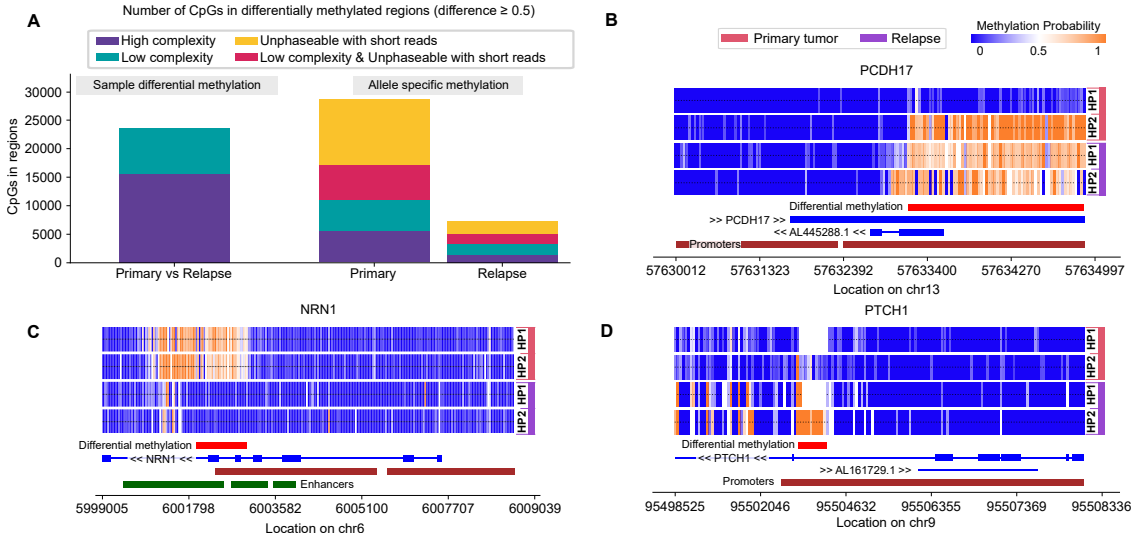

Figure 4.2.: Overview and examples of DMRs identified in medulloblastoma data. In **B-C** each rectangle refers to the a single CpG site and represents the methylation rate of that CpG site in the read-group represented by the rows. **A)** DMR CpGs identified in tumor comparison (primary tumor vs relapse) as well as ASM identified within tumor samples. *High complexity* refers to CpGs in regions which could also be mapped with short reads, whereas *low complexity* refers to CpGs in regions which are soft-masked in reference and would potentially not be resolved correctly in a WGBS experiment. *Unphaseable with short read* refers to CpGs which are more than 150bps from a heterozygous variant other than C>T and therefore requires long reads to be phased into haplotypes in order to identify ASM. **B)** ASM identified in the promoter region of gene PCDH17, a tumor suppressor gene [248]. **C)** Methylation in promoter of gene NRN1 in pre-treatment primary tumor. NRN1 was previously associated with treatment resistance in medulloblastoma [246]. **D)** Sample specific methylation in promoter of PTCH1, a driver gene in SHH medulloblastoma [249]. Haplotype 1 in both samples is deleted, whereas haplotype 2 is methylated in the relapse tumor and unmethylated in the pre-treatment primary tumor. Figure adapted from [187].

4.4. Allele-specific DNA methylation detected by pycoMeth

Using the same consensus segmentation as for the primary tumor vs relapse tumor DMR calling, I called ASM in both tumor samples. To do so, I pointed pycoMeth to the haplotype annotation in the MetH5 container and excluded unphased reads. In the primary tumor sample, 1,068 segments were classified as DMRs with an effect-size greater than 0.5 β-score difference, spanning a total of 28,803 CpG sites. An example is displayed in Figure 4.2B, which shows demetyhlation of haplotype 1 in the tumor sample at the promoter region of tumor suppressor gene PCDH17 [248]. The relapse sample appears to be unaffected, as both haplotypes in relapse are methylated.

In the relapse tumor sample, the number of ASM DMRs were much lower, with only 146 segments identified as DMRs, spanning 7,262 CpG sites (Figure 4.2A). Interestingly, when manually testing the 1,068 primary tumor ASM segments in the relapse sample, 370 (34.64%) also exhibited ASM effects in relapse, indicating that the low number of relapse ASM is potentially due to lack of test power from coverage. The `bs_diff` hypothesis in pycoMeth draws test power from coverage, and the 15x sequenced relapse sample only provides approximately 7.5x coverage per haplotype.

Similar to in the between-sample comparison, I further investigated the discoverability of ASM in short-read experiments. In addition to classifying CpGs into high and low complexity regions, I further broke them down into CpGs that could reasonably be expected to be phaseable in short-read data. A typical short-read Illumina sequencing platform produces reads in the length of 150bps (although longer reads exist). Therefore, in order to phase methylation calls, the presence of a heterozygous SNV within 150bps of the CpG-site is required. Furthermore, C>T SNVs are excluded from this selection, as such SNVs would be indistinguishable from cytosine methylation in a WGBS experiment. With these considerations, only 19% of CpGs in ASM DMRs fall in the category that is in high complexity regions and close enough to a phaseable SNV to be detected as ASM in a short-read WGBS experiment (Figure 4.2A).

4.4.1. Functional analysis of ASM

With ASE analysis performed (Chapter 4.1), I then attempted to correlate promoter ASM with ASE in the primary tumor sample. From 896 genes with significant ASE effects (p-value < 0.05), only 18 genes were among the ASM genes identified through pycoMeth. Of these 18, however, promoter methylation was strongly negatively correlated with expression (Pearson R: -0.59, p-value: 0.005). When accounting for allele-specific copy number, the partial correlation improves slightly (partial Pearson R: -0.60, p-value: 0.01, (Figure 4.3)).

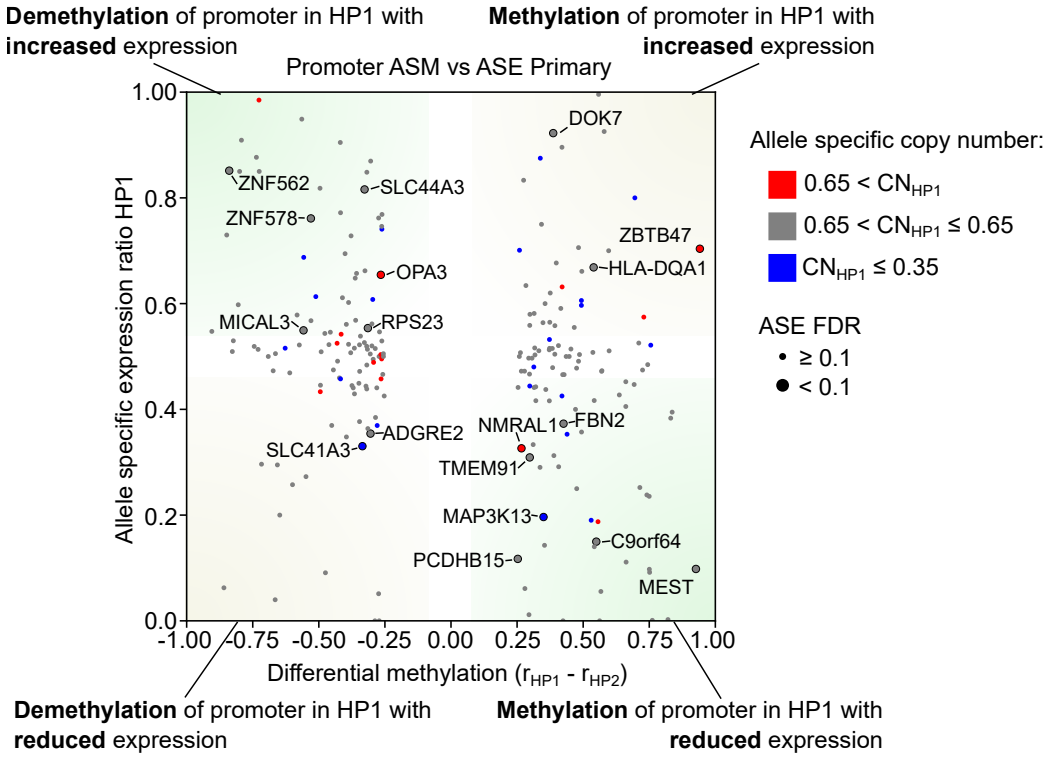

Figure 4.3.: Correlation between ASM and ASE in the primary tumor sample. Plotted are all genes with ASE regardless of p-value and ASM. Named genes are genes with significant FDR (Benjamini-Hochberg) after subsetting to genes which also show promoter ASM. Color indicates allelic copy number ratio, with red genes having higher copy number in haplotype 1 while blue genes have lower copy number in haplotype 1. Note that the scale of the x-axis is β-score difference ($\beta_{HP1} - \beta_{HP2}$), whereas the y-axis shows ASE ratio of haplotype 1 $\frac{\#reads_{HP1}}{\#reads_{HP1}+\#reads_{HP2}}$. Figure adapted from the medulloblastoma publication [187].

4.5. DNA Methylation patterns on structural variations

The CS11-17 structure present in this sample was assembled into two contigs (Chapter 4.1), contig 1 which consists of heavily rearranged segments of chromosome 11 and contig 2 which consists of a nearly 2Mbp continuous segment from chromosome 11. This structure appears to have been derived from a single haplotype, from which it was also deleted. I was therefore able to phase methylation calls from Nanopore reads into wildtype haplotype and chromothriptic (CS11-17) haplotype, allowing the interrogation of SV-specific methylation patterns. Methylation of contig 1 showed little ASM, with only two genes with containing promoter ASM regions. Contig 2 on the other hand appeared globally hypomethylated compared to the wildtype allele(Figure 4.4A). This hypomethylation covered the entire 2Mbp stretch that is contig 2 of CS11-17, which included the gene body of the two genes STK33 and TRIM66.

The other chromothriptic structure, the TI-threads, consists of short repetitive regions and is therefore harder to compare to wildtype. Long-read mappings of TI-threads to the reference genome result in complex chimeric alignments consisting of short primary mappings associated with many supplementary alignments. In order to ensure that Nanopolish was not influenced by the short alignment length, I instead performed an alignment against the polished assembly (Chapter 4.1) of the two largest TI-threads. First, I classified reads mapping into template regions on the reference genome as either belonging to a TI-thread or belonging to a wildtype haplotype. Then, I re-aligned the TI-thread reads to the TI-thread assembly, followed by running Nanopolish in order to produce methylation predictions. I then compared methylation called from TI-thread reads aligned to the TI-thread assembly with methylation of wildtype reads aligned to reference. Both TI-threads showed decreased methylation rate compared to wildtype, by about 0.18 and 0.28 β-score difference in TI-thread 1 and 2, respectively(Figure 4.4B-C).

4.6. Code availability

The Snakemake pipeline used to basecall, filter, align, and methylation call Nanopore data is available on GitHub under:

`https://github.com/snajder-r/nanopore`

The analysis code used to analyze results is published as part of the GitHub repository archiving all analysis scripts in the Medulloblastoma project:

`https://github.com/PMBio/mb-nanopore-2022`

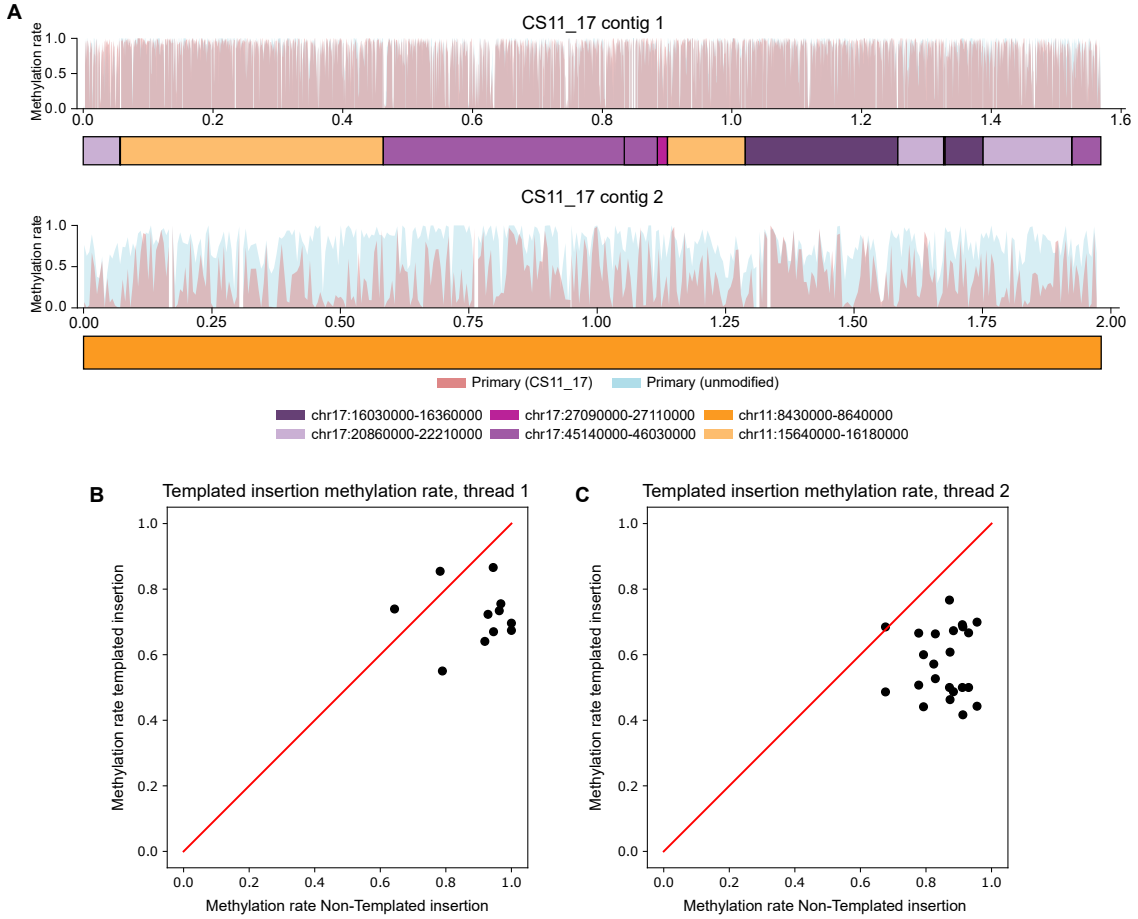

Figure 4.4.: Methylation analysis on chomothriptic SVs in medulloblastoma primary tumor. **A)** Contig 1 (top) and contig 2 (bottom) schematic representation of compositions and origin of sequences (chromosome and coordinates), together with the methylation rate in the chromothriptic haplotype (red) and the wildtype haplotype (blue). Methylation rate on contig 1 is typical for human genome (around 80% methylated) with few differences between alleles. Contig 2, on the other hand, is globally demethylated on the chromothriptic haplotype. **B-C)** Methylation rate of TI-threads compared to methylation of template regions in wildtype haplotype. Each point refers to one template region which is concatenated multiple times in a TI-thread. Figure adapted from the medulloblastoma publication [187].

4.7. Conclusion

In this exploratory study, I show the utility of pycoMeth and MetH5 in a cancer study of a single case of SHH medulloblastoma. Using pycoMeth, I was able to identify a large number of DMRs between before and after treatment tumor tissue, as well as highlighted ASM in the primary tumor sample, including aberrant DNA methylation on several medulloblastoma driver genes. A large portion of DMRs (34%) were in regions of low-complexity, and the majority (81%) of ASM DMRs in primary tumor were either in low-complexity regions or too distant from phaseable SNVs to be resolvable with short-read sequencing technologies.

Furthermore, taking advantage of methylation data stored in MetH5 format with haplotype annotation, I performed further integrative analyses of the methylome on structural variations, revealing hypomethylation in both chromothriptic structures CS11-17 and TI-threads. TI-threads have since been identified in a number of other cancer types [187] and further study of methylation patterns on these samples may be of interest.

Despite pycoMeth excelling at the detection of low-effect-size DMRs, this single-patient study warranted a more conservative reporting, as sources for potential validation are lacking. Therefore, only high effect-size DMRs were reported. Furthermore, DMRs were reduced to gene-level promoter methylation after DMR calling, in order to create a more easily digestable functional interpretation. Future studies may want to include low-effect-size DMRs in gene bodies or intergenic regions.

In conclusion, this study demonstrated the application of pycoMeth on a heavily perturbed cancer genome, as well as the benefits of long-read sequencing for DNA methylation profiling overall.

5. High resolution chromatin accessibility prediction

DNA methylation profiling has been used for the measurement of chromatin accessibility by *in vitro* methylating accessible chromatin using DNA methyltransferases, and then interpreting methylated DNA as accessible and unmethylated DNA as inaccessible. The NOMe-seq protocol uses M.CviPI in order to methylate accessible GpC sites, and then uses WGBS to read methylated cytosine [28, 123]. Cytosine in GpC context is then used to infer DNA accessibility, while cytosine in CpG context is interpreted as endogenous CpG methylation. This allows for the profiling of CpG methylation and DNA accessibility simultaneously. This method, however, is subject to the limitations of WGBS, such as short reads, which are difficult to phase into haplotypes or infer cis-regulatory interactions from, and the requirement to perform base conversion via bisulfite treatment or antibody pulldown.

The following chapter explores the application of Nanopore sequencing and methylation calling for the purpose of high resolution chromatin accessibility profiling in long reads, using multiple methyltransferases for multi-motif *in vitro* methylation of accessible chromatin. The ultimate goal of this study is to design a methylation caller that can be used for the study of transcription factor binding (Chapter 1.1.4). Such analyses have been previously performed using short read sequencing [95, 124]. Increased resolution and long reads may allow the study of more transcription factor binding motifs and co-binding behavior of nearby sites.

Experiments were performed on data obtained from *Drosophila melanogaster* (*D.mel*), an organism without endogenous cytosine or adenine methylation. The three methyltransferases applied here include CpG methyltransferase M.SssI, named after *E.coli* strain *Spiroplasma sp. strain MQ1* from which it was originally cloned [250], GpC methyltransferase M.CviPI, and context-free adenine methyltransferase EcoGII. Since context free adenine targets about 1 in 4 bases, this method would allow access to additional transcription factor binding motifs. Furthermore, it would lead to an overall increase in resolution to nearly 25%, compared to GpC methylation which only yields a resolution of 4.56%. Alternatively, it is possible to combine CpG and GpC methylation in organisms like *D.mel* without endogenous CpG methylation, yielding a resolution of 7.65%. Combining all three methyltransferases can yield a resolution of 36.50% and would allow simultaneous profiling of transcription factor binding sites with motifs containing all three methlyation motifs (Table 5.1).

Motif:	CpG	GpC	A	CpG+GpC	CpG+GpC+A
Resolution:	4.14%	4.56%	28.86%	7.65%	36.50%
Avg. distance:	24.16 bps	21.94 bps	3.46 bps	13.08 bps	2.74 bps

Table 5.1.: Resolutions available by individual or combinations of methylation motifs for chromatin accessibility profiling, represented as percentage of resolution and average distance between motifs. Numbers are computed based on overall motif distribution in reference genome *D.mel6*

Since existing tools such as Nanopolish are designed to call only one type of methylation in one particular context, this chapter includes the development of a specialized methylation caller. This caller is implemented as a modification to Nanopolish and designed to predict chromatin accessibility status from mixed methylation profiles. These methods can then be used for the discovery of transcription factor interaction (co-binding) inferred from co-accessibility on the same long reads.

5.1. Contributions

The study was designed in collaboration with Arnaud Krebs (EMBL) and Mathias Boulanger (EMBL). All Nanopore sequencing data used in this chapter were produced Mathias Boulanger (EMBL) and Rozemarijn Kleinendorst (EMBL).

5.2. Multi-modification Bayesian methylation caller

Nanopolish implements a Bayesian methylation caller which computes likelihood of observing Nanopore raw signal given that a subsequence is methylated and another given that it is unmethylated (Chapter 1.3.2). The difference between these (log-) likelihoods is then reported as the LLR of methylation (Equation 1.7). The model is parameterized by the mean and standard deviation of the Nanopore raw signal for methylated and unmethylated k-mers (Equation 1.6). Since Nanopolish assumes that nearby motifs share a methylation state, motifs closer than 10 bases are grouped together.

This grouping, as well as a number of further limitations, make it difficult to apply Nanopolish to the presented chromatin accessibility profiling setting. (i) Nanopolish is designed for a single modification type, and published models include a CpG model, a GpC model, and a dam model. Combined motifs, such as CpG+GpC are not supported by the implementation. (ii) Nanopolish assumes methylation occurs in self complementary context. This is a reasonable assumption for endogenous DNA methylation, as most methyltransferases indeed bind in self complementary sequences. With context-free adenine methylation, however, this is not given. CpG

methylation and GpC methylation on the 3'-5' strand can be collapsed to coordinates on the 5'-3' strand and still point at an appropriate motif. Yet, this is not the case with context free adenine methylation. (iii) Grouping of nearby methylation motifs is not viable in high resolution context, and would actually reduce the resolution of chromatin accessibility prediction. (iv) Finally, the tab-delimited file format with one line per methylation call is not suitable, particularly when interested in co-methylation on single reads.

To address (i-ii) I made minor changes to the Nanopolish training and prediction code. These changes allow for multiple modification types from mixed canonical bases (e.g. adenine and cytosine) regardless of whether the binding motif is self-complementary. Addressing (iii), I implemented an additional subcommand in Nanopolish, called `call-accessibility`, which is tailored for near-basepair resolution chromatin accessibility calling. Lastly, the output of the `call-accessibility` subcommand efficiently writes dense methylation calls on a read-level (iv). The following section describes the modifications I made to Nanopolish as well as information on the training strategy.

Parallel to the development of our specialized Nanopolish version, ONT released a new methylation caller Remora (Chapter 1.3.2). Remora produces single-basepair resolution methylation calls and implements a deep learning architecture, yet shares some of the limitations of Nanopolish (specifically i-ii from above). To place my specialized Nanopolish version into context, I also modified Remora accordingly and trained it on the same data as the modified Nanopolish version presented here. The evaluation section (Chapter 5.2.3) shows the performance of both the Nanopolish and Remora models on hold-out data not used in training.

5.2.1. Methylation caller design

Original Nanopolish searches for methylation motifs such as CpG or GpC and then groups nearby motifs together to produce a subsequence on which to call methylation. For my modified Nanopolish command, I instead implemented a sliding window approach (Figure 5.1). Since sequence context is necessary in order to compensate for basecalling errors or sequence-to-signal alignment errors, a methylation call must still be performed on a sequence window. For window size 16, the sequence context used for methylation calling on base R_n is then represented as $S = [R_{n-8}, ..., R_n, ..R_{n+7}]$, where R is the reference genome sequence the read had been aligned to and S the sequence context for methylation calling.

Standard Nanopolish would then compute a LLR based on sequence context as $\text{LLR} = \log L(X|S_M) - \log L(X|S_R)$ (Equation 1.7). This assumes that the entire 16bp sequence is either methylated or unmethylated which would fail to accurately portray changepoints in chromatin accessibility. While this limitation applies to endogenous methylation calling as well, it is especially problematic for study of transcription factor binding, where footprints are measured in 5bps bins [95]. Since

model parameters are based on 6-mers, and are either fully methylated or unmethylated, it is impossible to completely avoid this discrepancy, but in this modified implementation I attempted to improve resolution without reducing window size. To do so, I implement marginalization of methylation likelihood for flanking 6bps regions.

In a first step, the sequence context S is split into three 6bp long parts, $S^{Pr} = [S_0, ..., S_5]$, $S^C = [S_6, ..., S_{11}]$, and $S^{Po} = [S_{12}, ..., S_{16}]$, the prefix, center, and postfix sequence, respectively. Likelihood can be split into

$$\log L(X|S_M) = \log L(X^{Pr}|S_M^{Pr}) + \log L(X^C|S_M^C) + \log L(X^{Po}|S_M^{Po}) \tag{5.1}$$

The goal is then to compute LLR based only on the center sequence, that is

$$\text{LLR} = \log L(X^C|S_M^C) - \log L(X^C|S_R^C) \tag{5.2}$$

First, I derive the marginal likelihood of the postfix and prefix sequence independent of methylation status

$$L(X^{Px}) = L(X^{Px}|S_M^{Px}) + L(X^{Px}|S_R^{Px})$$

where Px stands for either Po or Pr.

I can then expand LLR to

$$\begin{aligned}\text{LLR} = &\log L(X^{Pr}) + \log L(X^C|S_M^C) + \log L(X^{Po}) \\ &- \log L(X^{Pr}) - \log L(X^C|S_R^C) - \log L(X^{Po})\end{aligned}$$

However, likelihoods still need to be computed on the entire 16bps window in order to allow sequence-to-signal alignment to perform reliably. Therefore, I implemented likelihood computation on the full 3-part sequences, for all combinations of methylation states in the flanking regions. Let S_{ijk} refer to a query sequence with methylation status i in the prefix, j in the center and k in the postfix. For example, S_{MMR} is methylated in the prefix and center and unmethylated in the postfix.

Using Equation 5.1, given such a mixed-methylated sequence, the likelihood can then be described as the product

$$L(X|S_{ijk}) = L(X^{Pr}|S_i^{Pr})L(X^C|S_j^C)L(X^{Po}|S_k^{Po}) \tag{5.3}$$

I then implement likelihood calculation as

$$\text{LLR} = \log \sum_{i\in\{M,R\}} \sum_{k\in\{M,R\}} L(X|S_{iMk}) - \log \sum_{i\in\{M,R\}} \sum_{k\in\{M,R\}} L(X|S_{iRk}) \tag{5.4}$$

To derive how this equality holds, let's define some shorthands. Let $Y^Z = L(X^Z|S_Y^Z)$ where $Z \in \{Pr, C, Po\}$ is the segment and $Y \in \{M, R\}$ the methylation state.

Then the first sum evaluates to

$$\begin{aligned}
&\log \sum_{i\in\{M,R\}} \sum_{k\in\{M,R\}} L(X|S_{iMk}) = log \sum_{i\in\{M,R\}} \sum_{k\in\{M,R\}} i^{Pr} M^C k^{Po} \\
&= \log(M^{Pr}M^C M^{Po} + M^{Pr}M^C R^{Po} + R^{Pr}M^C M^{Po} + R^{Pr}M^C R^{Po}) \\
&= \log(M^C(M^{Pr}M^{Po} + M^{Pr}R^{Po} + R^{Pr}M^{Po} + R^{Pr}R^{Po})) \\
&= \log(M^C(M^{Pr} + R^{Pr})(M^{Po} + R^{Po})) \\
&= \log M^C + \log(M^{Pr} + R^{Pr}) + \log(M^{Po} + R^{Po})
\end{aligned}$$

and analogously the second sum evaluates to

$$\log \sum_{i\in\{M,R\}} \sum_{k\in\{M,R\}} = logR^C + \log(M^{Pr} + R^{Pr}) + \log(M^{Po} + R^{Po})$$

substituting these in equation Equation 5.4 yields

$$\begin{aligned}
\text{LLR} = \quad & \log M^C + \log(M^{Pr} + R^{Pr}) + \log(M^{Po} + R^{Po}) \\
& - \log R^C - \log(M^{Pr} + R^{Pr}) - \log(M^{Po} + R^{Po}) \\
= \quad & \log M^C - \log R^C \\
= \quad & \log L(X^C|S_M^C) - \log L(X^C|S_R^C)
\end{aligned}$$

which is what was defined as the goal in Equation 5.2. Note that LLRs in regions without a methylation motif in the center segment will simply be returned as 0, since M^C and R^C will be the same. This is desired, as a LLR of 0 corresponds to a methylation probability, and thus probability of chromatin accessibility, of 50%, which means complete uncertainty.

One more issue that needs considering is that mixed methylation states at the segment boundaries can result in mixed-methylation k-mers, where the start of a k-mer is methylated and the end is not. For this purpose, the model needs to be extended with all combinations of mixed methylated 6mers such that first $n < 6$ are methylated and the remaining $n - 6$ bases are not, or vice versa. Depending on the number of methylation motifs in the model, this can lead to an increase from 2 (fully methylated/unmethylated) to 7 k-mer variations. Since training data does not contain such mixed-methylated k-mers, however, these mixed k-mers simply receive dummy parameters, which contribute equally on both sides of the LLR calculation and therefore do not influence the LLR computation.

Output format

High resolution accessibility calls require an efficient storage solution, as many more LLRs are produced per read. Since the purpose of this caller is to be used for transcription factor co-binding on single reads, the read-dimension is more important

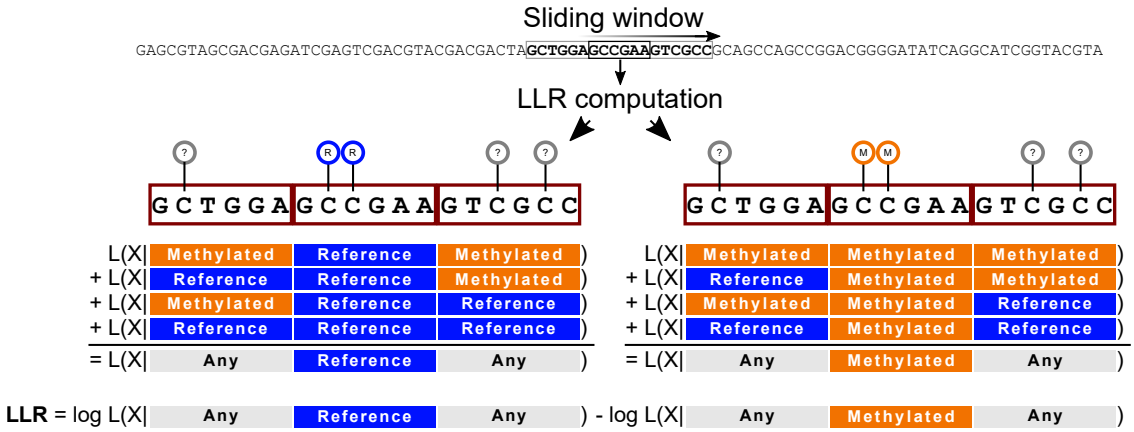

Figure 5.1.: Specialized methylation caller design for high resolution calling, based on a modified Nanopolish implementation. Methylation calls are produced on 16bps windows using a sliding-window fashion. Windows are then split into three 6bps segments, the center segment for which the LLR should be computed, and the two flanking segments (prefix and postfix). Methylation status of the flanking segments is then marginalized out to produce a methylation call for at most 6bps context.

than the genomic coordinates. MetH5 is designed for rapid access to genomic coordinates and read-level aggregation, but not optimized for such a high density of methylation calls, as genomic coordinates would be stored for each call.

For this reason, and also because I was modifying Nanopolish, I decided to implement a variation of the original Nanopolish tab-delimited text output format, yet with one line per read instead of one line per methylation call. As a compromise between accurate storage of LLR and efficient usage of datatypes, I decided to store LLRs for an entire read as an ASCII string, not unlike how quality strings are stored in FASTA files. Each line in the output format consists of the start coordinates at which the read mapped (chromosome and position on the reference genome), the mapping direction/strand, the read name, and an ASCII string containing all LLRs computed for this string, including zero-LLRs for regions with no methylation motif. Long sequences of zero-LLRs can be easily compressed (e.g. using bzip compression) Including all LLRs reduces the dependency on other forms of methylation call alignment, such as the distance based method implemented in the MM tag in the BAM format, or coordinate based as in MetH5.

The transformation from LLR to ASCII character is performed as:

$$f(\text{LLR}) = \max(\min(\lfloor 79 + \text{sign}(\text{LLR}) \cdot 15 \cdot \log(1 + |\text{LLR}|) \rfloor, 33), 125)$$

The resulting value corresponds to an ASCII character, where a LLR of 0 evaluates to 79 (ASCII `O`), LLRs less than -20 evaluate to 33 (ASCII `!`) and LLRs larger than 20 evaluate to 125 (ASCII `}`). The logarithmic scale makes it so that LLRs closer

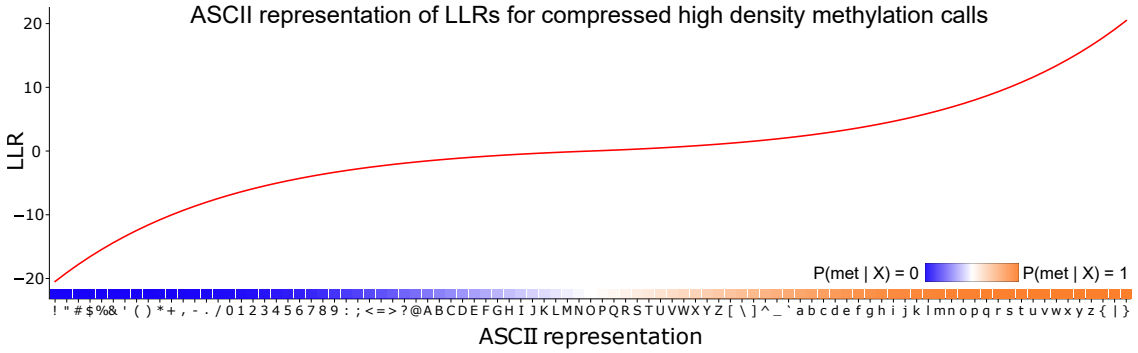

Figure 5.2.: Mapping of ASCII character to LLR in the output format of the modified Nanopolish version for high resolution / high density chromatin accessibility calling from methylation. LLR 0 maps to ASCII character 'O'. Colored squares represent LLR transformed to methylation posterior probability as visualized in pycoMeth plots.

to zero are resolved with more detail than LLRs with large magnitude(Figure 5.2). The inverse function used when reading the file format is then

$$\text{LLR} = f^{-1}(a) = \text{sign}(a - 79)\left(\exp\left(\frac{|a - 79|}{15}\right) - 1\right)$$

5.2.2. Training

Training data was produced from *D.mel* by *in vitro* methylation of DNA after nucleosome removal, yielding DNA that is completely methylated (to the degree of methyltransferase efficiency). For the training of unmethylated k-mers, completely unmethylated DNA was obtained as well. In order to investigate multiple combinations of methylation motifs, three different settings were explored: (i) single-enzyme methylation using the context-free adenine methyltransferase, (ii) double-enzyme enzyme methylation using the two cytosine methyltransferases (CpG and GpC), and (iii) triple-enzyme methylation combining both the two cytosine methyltransferases and the context-free adenine methyltransferase.

I then modified the training functions provided by Nanopolish to allow for multiple methylation motifs and to also allow for methylation of non self complementary motifs. Applying this modified training routine, I then train four Nanopolish models, one for each of the three methylated datasets, plus one on the unmethylated dataset in order to fine-tune parameters of unmethylated k-mers (Figure 5.3).

5.2.3. Evaluation

I evaluated models based on hold-out data from fully methylated and and unmethylated data which was not used in training. Furthermore, I also modified ONT's open

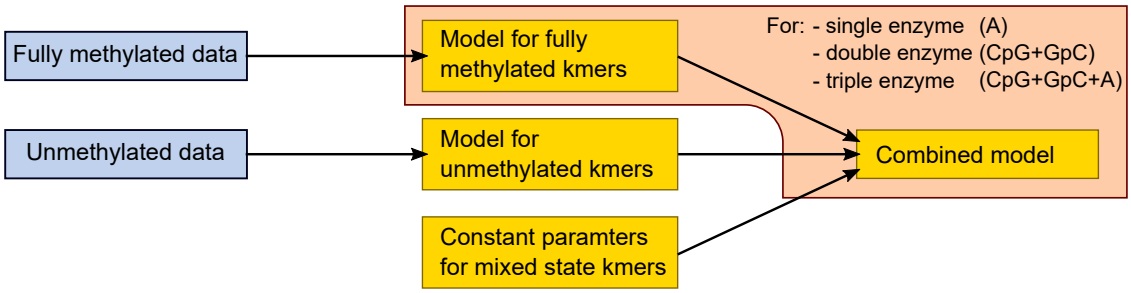

Figure 5.3.: Training of modified Nanopolish for chromatin accessibility. Three methylated models are trained from fully methylated data, with adenine, cytosine in CpG+GpC, or all three combined. One unmethylated model is trained. Constant parameters for mixed states are included to mitigate that no training data is available for mixed methylated k-mers(Chapter 5.2.1).

source methylation caller Remora to support multiple methylated bases and trained Remora on the same data, using the training routines provided by Remora.

Since both Nanopolish and Remora report uncertainties of methylation calls, in the case of Nanopolish as a Bayesian expression represented by the LLR and in the case of Remora as the activation of the final classifier layer, methylation calling results can be thresholded to include only calls of a minimum degree of certainty. A higher certainty threshold typically results in higher accuracy calls (assuming a well calibrated model), but also result in fewer usable calls, reducing coverage and resolution. When comparing different callers, it is therefore important to consider how they perform over varying different certainty thresholds, while certainty is defined as a threshold to absolute LLR. Activations from Remora are transformed to the same scale, by applying the logit function (Equation 3.3).

Comparing Nanopolish and Remora shows good performance in both methods from the double-enzyme model, with Nanopolish having higher precision with lower recall, but a better F1-score than Remora. When aiming for at least 80% of covered motifs called, Nanopolish achieves an F1-score of 0.9713, while Remora achieves 0.9636. With single- and triple-enzyme, where adenine methylation is called, Remora performs better than the Nanopolish model, however both models show poor performance. With the same target 80% calling rate, Nanopolish achieves F1-scores of 0.7616 and 0.6106 for single-enzyme and triple-enzyme, respectively, and Remora 0.7827 and 0.6644.

These rather discouraging results for adenine methylation calling could be the result of a number of factors. The simplest explanation may be that adenine methylation does not perturb the current signal as strongly as cytosine methylation, therefore leading to a higher signal-to-noise ratio for the problem of methylation calling. This problem could be improved with future advances in Nanopore chemistry, however

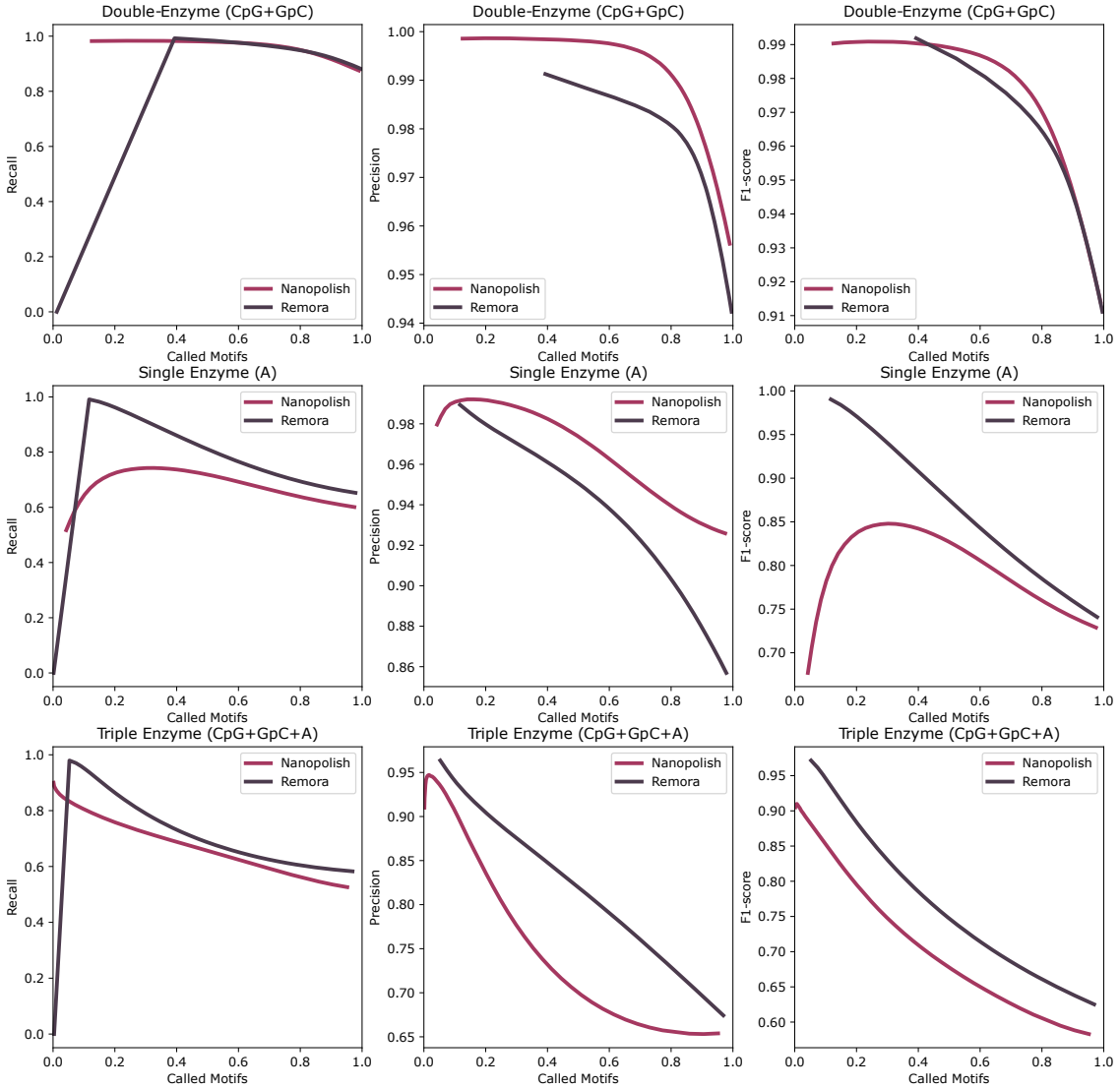

Figure 5.4.: Evaluation of chromatin accessibility model implemented from modified and retrained Nanopolish as well as alternative Remora model on the three data-types. X-axis maps the fraction of methylation motifs which could be confidently called, while lines are plotted by varying the threshold for confidence from 0.1 to 20 absolute LLR. The y-axis represents, from left to right, the recall (TPR), precision, and F1-score as a compromise between recall and precision(Equation 3.6).

the fact that Remora performs better on adenine methylation may be an indication that deeper models with more sequence context may be sufficient. Another possible explanation for reduced performance in adenine methylation could be biases against highly methylated molecules in sequencing, reducing to flawed training and testing data.

In any case, seeing how adenine methylation could not be reliably called, the remaining evaluations were performed on the double-enzyme model.

Viability for transcription factor co-binding study

Finally, I evaluated the viability of the model for the study of transcription factor co-binding in *D.mel*, assuming an absolute LLR threshold of 3.4, which results in 80% of motifs being called. While methylation calling accuracies using the proposed Nanopolish model are rather accurate, it is important to consider the potential accumulation of errors and drop-outs in the analysis. A bound transcription factor can be identified as short inaccessible regions flanked by short accessible regions, while an unbound transcription factor is either completely accessible or completely inaccessible if nucleosome occupied (Chapter 1.1.4).

Following the literature [95], I characterized a transcription factor as a three-bin region which then need to be classified as accessible (A) or inaccessible (I), where each bin is 20bps in size. A pattern of AIA would then correspond to a bound transcription factor, III an unbound transcription factor, and the remaining 6 combinations as nucleosome occupied. In order to analyze transcription factor binding, all three bins must contain a methylation motif and contain a confident (and correct) methylation call. Furthermore, for analysis of co-binding, the same must be true for both transcription factors in a read, hence extending the number of bins which must be called correctly to 6.

In order to estimate how drop-outs and methylation calling errors propagate in the analyiss, I first estimate the average number of methylation motifs (CpG or GpC) in each bin, using transcription factor annotations for *D.mel* from JASPAR 2020 [251]. I find that the average number of motifs in the flanking bins is 1.75, while it is 2.0 for the center bin. Using these averages, I then modeled the number of correct methylation calls for accessible and inaccessible bins as a binomial distribution and computed the expect number of bin-classifications. This was done under consideration of the two different error types (false positives and false negatives) in methylation calling. The resulting confusion matrix (Figure 5.5A) shows high accuracy, with 95% of bound and 93% of unbound transcription factors being accurately classified.

Further it is important to consider that lack of methylation motif in any of the three bins, or lack of methylation calls passing the threshold in one of the bins, results in that motif being unable to be called. To estimate the number of motifs that can be used in co-binding analyses, I first need to assume a binding fraction. Literature

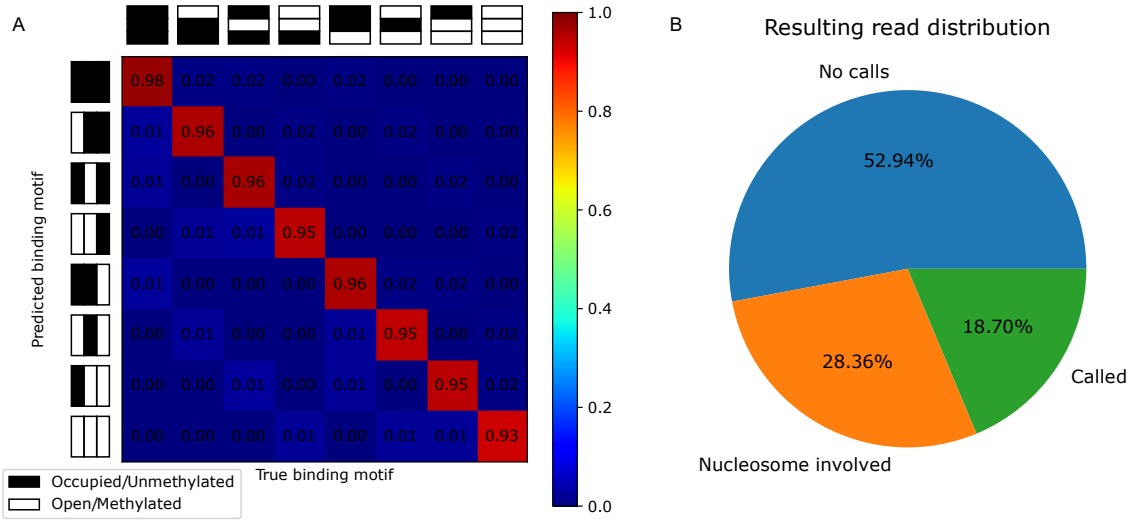

Figure 5.5.: Analysis of transcription factor binding site classification using the modified Nanopolish caller for chromatin accessibility profiling. **A)** Confusion matrix between transcription factor binding motifs represented as three bins. **B)** Breakdown of expected distribution of transcription factor classification results. "No calls" refers to sites in which at least one bin has no valid methylation call. "Called" refers to motifs which were classified as bound or unbound, while "Nucleosome involved" refers to all other classifications.

suggests an expected distribution of 15% of motifs being bound, 35% unbound and 50% nucleosome occupied [95]. I then find that using these assumptions and the evaluation statistics an expected 52.94% of motifs would not receive a classification, 27.36% would be classified as nucleosome occupied and 18.70% would be called as either bound or unbound and therefore usable in a co-binding analysis.

Finally, I performed a power analysis via simulation, determining the sequencing depth required to accurately identify co-bound transcription factors. In this simulation, I simulated 300 pairs of transcription factor binding sites, where 150 pairs (50%) were co-binding and the other 50% were independently binding. For each motif-pair, I simulated a number of reads matching the sequencing depth. Since even co-bound transcription factors do not always perfectly match their binding states, I estimated a co-binding rate of 50%. That means that a co-bound pair will match the binding state in 50% of reads and will have independent binding state (which may also be matching) in the other 50% of reads. For each read, I then drew classifications for both motifs from the confusion matrix, or with probability matching the drop-out rate decide that the motif cannot be classified. I then computed a contingency matrix with the number of reads bound and unbound in both motifs of the pair and performed a Fisher-Exact test to determine whether the pair is identified as co-binding (p-value < 0.05). I repeated this simulation for varying

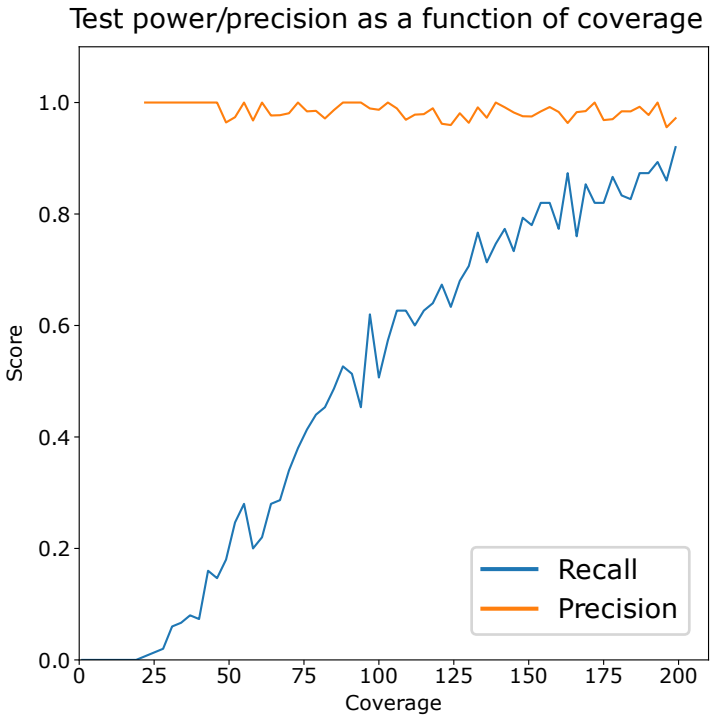

Figure 5.6.: Power analysis of transcription factor co-binding analysis using modified Nanopolish model. Simulating pairs of transcription factors, 50% co-binding and 50% independently binding, at varying sequencing depth, to analyze recall and precision of co-binding prediction.

sequencing depths, computing precision and recall for co-binding identification.

The result shows that high coverage is required to compensate for the high number of dropouts and nucleosome occupation. The number of false positives, however, is low at any coverage, while detection power increases with increasing sequencing depth. Using the assumption made above, I estimate that a coverage of 150x is sufficient to detect 80% of co-binding transcription factor pairs (Figure 5.6). Note, that this coverage describes the number of reads covering both binding sites, and actually sequencing depth needs to then be adjusted to consider for read length distribution and distance between binding sites to be studied.

5.3. Code availability

The modified Nanopolish and Remora versions have been forked from their original GitHub repositories. Remora modifications also required slight modifications to the open-source ONT basecaller bonito.

- Nanopolish: `https://github.com/snajder-r/nanopolish`
- Remora: `https://github.com/PMBio/remora`

- Bonito: `https://github.com/PMBio/bonito`

5.4. Conclusion

Methylation based profiling of chromatin accessibility on long reads shows good potential for the study of transcription factor interaction, given that read lengths and sequencing depth manage to cover pairs of transcription factor binding sites. The combined calling of CpG and GpC methylation increases the resolution compared to only GpC-based profiling from 21.94bps to 13.08bps on average, and the Nanopolish model I implemented performs better than the deep-learning model Remora when trained on the same data. Still, seeing how transcription factor footprinting requires accurate methylation calling on three (small) bins per binding site, drop-outs from lack of methylation covered motifs or from quality filtering of methylation calls reduce accuracy. This leads to a requirement of high sequencing depth to reach reasonable power for the detection of co-binding.

Chromatin accessibility profiling based on adenine methylation would massively increase the resolution (to 3.46bps) and allow profiling of transcription factors with binding sites not containing CpG or GpC motifs. Unfortunately, with the methylation calling accuracies observed from both the modified Nanopolish and Remora, the error is too great to perform transcription factor co-binding analysis. It is unclear whether this is an inherent issue with sequencing highly adenine methylated molecules, or whether signal of adenine methylation is more challenging to detect. Future studies may focus on training of deeper models and bespoke architectures for the detection of adenine methylation in Nanopore data. Future versions of Nanopore chemistry (pore, motor protein) could also be explored for improved methylation signal.

6. Summary

In this thesis I presented novel standards, software infrastructure, and machine learning methods for epigenetic analysis from Nanopore sequencing data. This includes **MetH5**, a high-performance, rapid random access, and parallelizable storage container optimized for analysis and visualization of methylation calls from Nanopore. MetH5 efficiently stores reference-anchored methylation calls including methylation call uncertainty. Furthermore, it allows for the storage of read information, including haplotype assignment or other read annotations. The MetH5 python API and CLI implement optimized routines for creation of MetH5 files, data querying, and aggregation of important quality statistics in a user friendly interface. When compared with the in parallel developed MM tag used in BAM files for methylation storage, MetH5 shows massively improved access times, both for random access as well as sequential access, at little cost to storage space usage.

With **pycoMeth** (version 2), I presented an improvement to the original pycoMeth prototype, where I implemented a Bayesian methylation segmentation suite in the form of a HMM changepoint detection algorithm. This segmentation algorithm allows for consensus segmentation of multiple methylation profiles, such as from multiple samples or haplotypes, and models methylation call uncertainties reported by the methylation caller in a Bayesian framework. Furthermore, in addition to implementing improved parallelization, I expanded pycoMeth's differential methylation testing suite by adding two more test modes and improved correction for multiple testing including IHW. I demonstrate the utility of pycoMeth both on a simulated dataset with ground truth segments and DMRs as well as a real dataset and show that pycoMeth performs equivalent or better than existing tools designed for WGBS.

Applying pycoMeth to a cancer study from a patient with chromothriptic medulloblastoma, I demonstrated the benefits of DNA methylation analysis on Nanopore data compared to WGBS and report methylation related to ASE and structural variations.

Finally, I explored the potential of methylation based **chromatin accessibility profiling** using Nanopore sequencing, and developed a specialized methylation caller for mixed methylation motifs. This caller was implemented as a modification to Nanopolish, which I then trained on fully methylated and unmethylated Nanopore data of varying methylation motifs. In parallel, I trained the deep learning based methylation caller Remora on the same training data, and compared the perfor-

mance. While both tools managed to produce good methylation calls for the mixed CpG+GpC motif, with the modified Nanopolish model performing slightly better, neither model was able to accurately learn adenine methylation from the given training data. In a power analysis, however, I show that CpG+GpC may be sufficient for the analysis of transcription factor co-binding utilizing long Nanopore reads, given that the transcription factors contain CpG or GpC motifs and sufficient sequencing depth can be generated

7. Future research

Long term read-level storage of methylation data will undoubtedly find a gold-standard in the BAM format, seeing how it has high adoption and is well supported in many visualization and analysis tools. While the **MetH5** format offers much higher performance, making it a good intermediate format for methylation analysis, continued maintenance and development is required to keep it relevant. This can be capitalized on by expanding the MetH5 toolkit with additional language APIs, such as for R, which is used in a plethora of standard methylation analysis workflows. Furthermore, with increasing interest in the study of oxidative derivatives of 5mC, MetH5 should be expanded to allow for multiple modification calls for different modification types, such as 5mC, 5hmC, 5fC, or 5caC on the same cytosine and read. While this is already possible in BAM format, the relational design of the MetH5 container is a natural choice for this, as the LLR dataset could be expanded with additional dimensions. Thereby, the ranges dataset could be shared for fast access and efficient storage, while the MM tag in the BAM format require repeating the (relative) coordinates for each call.

While the current **pycoMeth** segmentation model works well when read-groups (such as samples or haplotypes) can be assigned *a priori*, it may be possible to infer read-groups directly from methylation, or a combination of methylation and genomics, during the segmentation process. Recent studies have shown the potential of clustering reads into epigenetic states based on methylation signal [252]. Such a step could be implemented directly in the Bayesian model, and help to identify reads from different cell cycle states, cell- and tissue-types. Further potential improvements to the pycoMeth differential methylation testing module include additional tests, such as implementation and benchmarking of a beta-regression test for the β-score difference hypothesis, which may be better suited for comparing ratios than the currently implemented rank based test. Furthermore, the IHW implementation currently shows little improvement to testing power and future work may include implementation and benchmarking of different weights for IHW. As for computational performance, current segmentation algorithms are currently expensive to compute and scale poorly with increasing sample size. The current implementation of the Bayesian segmentation algorithm could be converted to a Bayesian neural network. Model compression could then be applied to improve performance and allow for hardware acceleration using GPU architecture.

The mixed-motif methylation calling for the purpose **chromatin accessibility** profiling, will require further research and development, especially for the m6A methy-

lation calling. Adenine methylation calling would allow for massively increased resolution and access to more transcription factor binding motifs. In the presented benchmark, both the Bayesian caller Nanopolish as well as the deep learning model Remora have difficulty predicting adenine methylation. As a first step, the quality of training data will need to be verified. At the moment it is unclear, whether Nanopore reads from the fully adenine methylated samples were actually methylated at a high degree, or whether biases in the Nanopore chemistry preferably sequenced unmethylated DNA. The degree of such a bias, if one exists, could be verified by sequencing methylated (5mC and/or 6mA) and unmethylated DNA together, followed by clustering of raw Nanopore signal, as was performed in early methylation callers [179]. Recent developments in methylation calling from Nanopore have shown potential of sequence based deep learning architectures, such as the transformer model [253]. Seeing how Remora performed better than Nanopolish in the calling of adenine methylation, such deep architectures may be explored for the problem of adenine or combined adenine and cytosine methylation calling.

Furthermore, with new sequencing kits being released by ONT, including new pore versions and modifications to the motor enzyme, models for methylation calling will require retraining to operate on signal retrieved from newer chemistry. This may trigger a re-evaluation of adenine methylation using newly trained deep learning models.

Appendix A.

Example uncertainty propagation

To give a toy example, assuming $m(x) \in \{met, \neg met\}$ is a binary classifier based on Bayesian inference deciding whether cytosine in a CpG is methylated based on measurement x, and $r(X)$ computes the estimated methylation rate of cytosines in region X, then the naïve approach would be to reduce $m(x)$ to a binary prediction for each x.

$$m(x) = \begin{cases} 1 & \text{if } P(met|x) > 0.5 \\ 0 & \text{otherwise} \end{cases}$$
$$r(X) = \frac{1}{|X|} \sum_{x \in X} m(x)$$

However, the methylation prediction may be easier in one sequence context than another, and the result of $m(x)$ may be more reliable in the easier context. In Bayesian terms, the posterior $P(met|x)$ directly scales with $P(x)$ (Equation 1.1), the term representing the information value of evidence x. Uncertainty propagation refers to forwarding this information to the downstream model. In this toy example, r could instead be modeled as the expectation value of r given x and m:

$$r(m(X)) = E\left[r|m, X\right] = \frac{1}{|X|} \sum P(met|x)$$

Appendix B.

MetH5 CLI

The MetH5 CLI is implemented in python and part of the meth5 python module. The purpose of the MetH5 CLI is to provide users with an interface from which they can perform basic operations on MetH5 file, such as the creation of MetH5 files, read-annotation, as well as extraction of methylation aggregates and quality statistics. This chapter describes the functionality implemented in the MetH5 CLI.

B.1. Writing MetH5 files

MetH5 creation

Two modes for MetH5 creation have been implemented in the CLI. The `create_m5` subcommand allows for the creation of a MetH5 container from one or multiple tab-delimited input files, such as the format created by the Nanopolish methylation caller. In order to keep the memory profile during creation constant, the MetH5 CLI will load the tab-delimited text files in fixed-size batches of methylation calls and incrementally build the MetH5 container. While this routine ins designed to be used with Nanopolish output files, the format is easily human readable and can be created by users from output of any other methylation caller.

The second mode for MetH5 creation is implemented in the `convert` subcommand. This mode will can read multiple BAM files with MM and ML tag and convert the methylation calls stored in BAM files into a single MetH5 container. Seeing how this mode depends on pysam, which at this time is not yet compatible with MM tags making use of the `?` modifier (Chapter 1.3.2), compatibility with BAM files created by Remora or Guppy is currently not given, but the command is ready to be deployed with an updated version of pysam once available, allowing for MetH5 creation from Remora or Guppy methylation calls.

Both methods of MetH5 creation allow users to choose compression options, chunk size, and a selection of chromosomes to be included in the MetH5 container.

Read annotation

Using the `annotate_reads` subcommand, read-group annotations can be applied to an existing MetH5 file. This function requires a tab-delimited input file with two columns: the read name and the read-group. The read-group key under which read-groups should be stored is provided when calling the subcommand. Read groups are then assigned unique integer representation, and stored as datasets in the MetH5 file, together with the metadata required to map integer read-groups back to the original text representation.

B.2. Reading MetH5 files

Listing contents

Since MetH5 is designed for parallel computing with load distribution based on chunked storage, it is important for users to be able to query the number of chunks per chromosome at a given chunk size. For this purpose, the MetH5 CLI implements the `list_chunks` subcommand, which will list the chromosomes and number of chunks per chromosomes for an existing MetH5 container.

BedGraph extraction

The BedGraph track format is a widespread file format used for visualization of 1-dimensional data on genomic coordinates [254]. The MetH5 CLI implements the `bedgraph` subcommand for the extraction of quality statistics over genomic coordinates. Users can subset regions or create BedGraph for the entire contents of the MetH5 file. Currently implemented statistics are *methylation rate*, which computes the methylation β-score for each region after thresholding LLRs, and *coverage*, which computes the number of reads covering a region after filtering out low confidence methylation calls.

Region statistics

The `region_stats` subcommand is designed to compute aggregate statistics for regions defined by the user and supplied as a second input file. For each provided region, the MetH5 CLI then computes the overall methylation β-score, the number of unique CpGs covered by the region, and the number of total methylation calls passing the confidence threshold from all reads and CpGs in the region. The output is returned as one line per region in a tab-delimited format.

Appendix C.

Figures

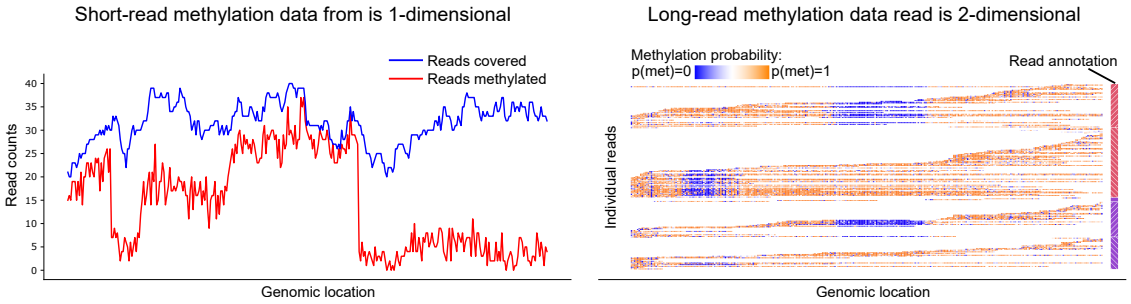

Figure C.1.: Difference between methylation call storage from short-read and long-read data. Left: short-read methylation calls are typically reduced to a 1-dimensional signal. For each CpG-site, the number of total reads and the number of methylated reads is stored. Right: long-read methylation calls can be seen as sparse 2-dimensional data.

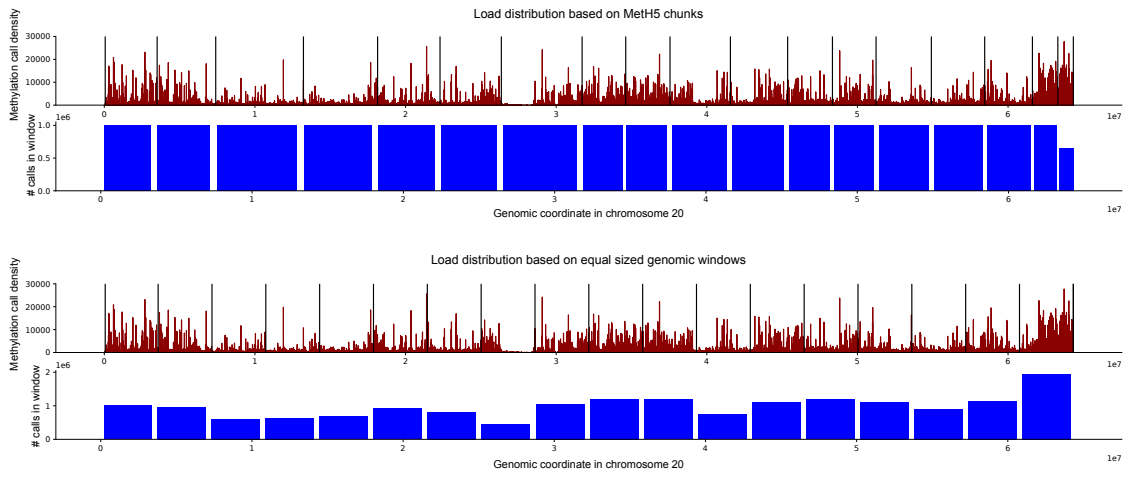

Figure C.2.: Top: Parallel operations over methylation-call based chunks lead to even load distribution. Bottom: Using evenly spaced genomic windows to define work packages results in uneven load distribution caused by variation in CpG density and mapping coverage. Figure adapted from the pycoMeth publication [207]

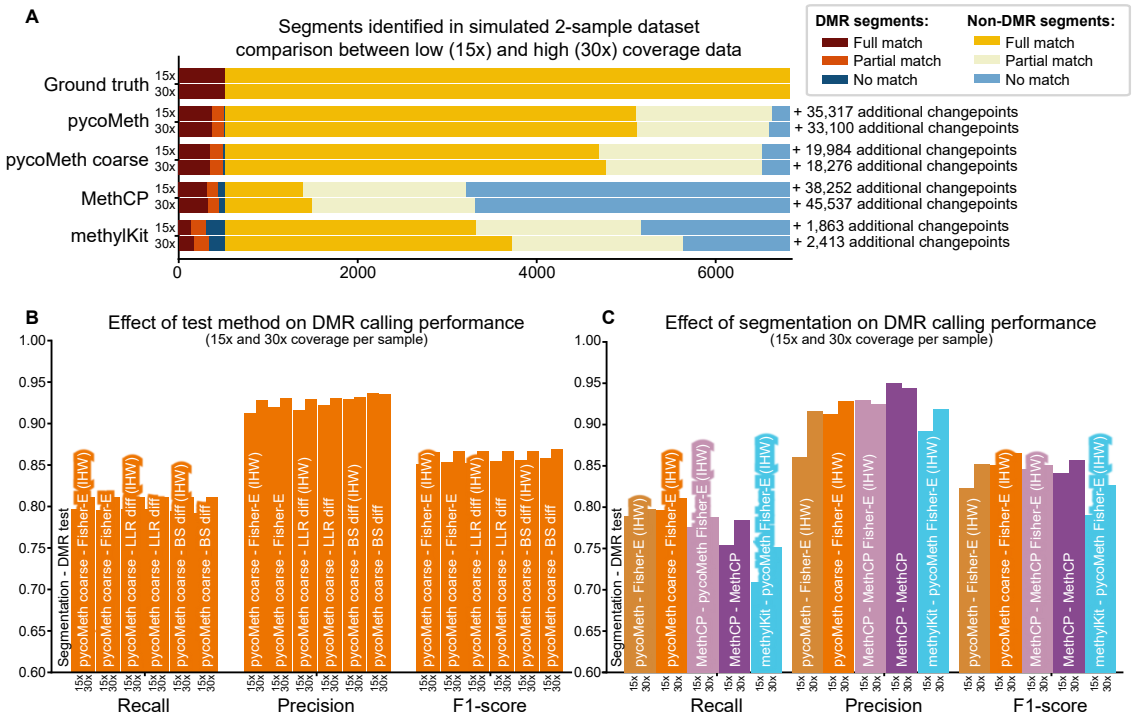

Figure C.3.: Comparison between pycoMeth performance on high coverage (30x) and low-coverage (15x) simulated data. Figure adapted from the pycoMeth publication [207] **A)** Segmentation evaluation as in Figure 3.5. **B-C)** DMR calling evaluation as in Figure 3.6

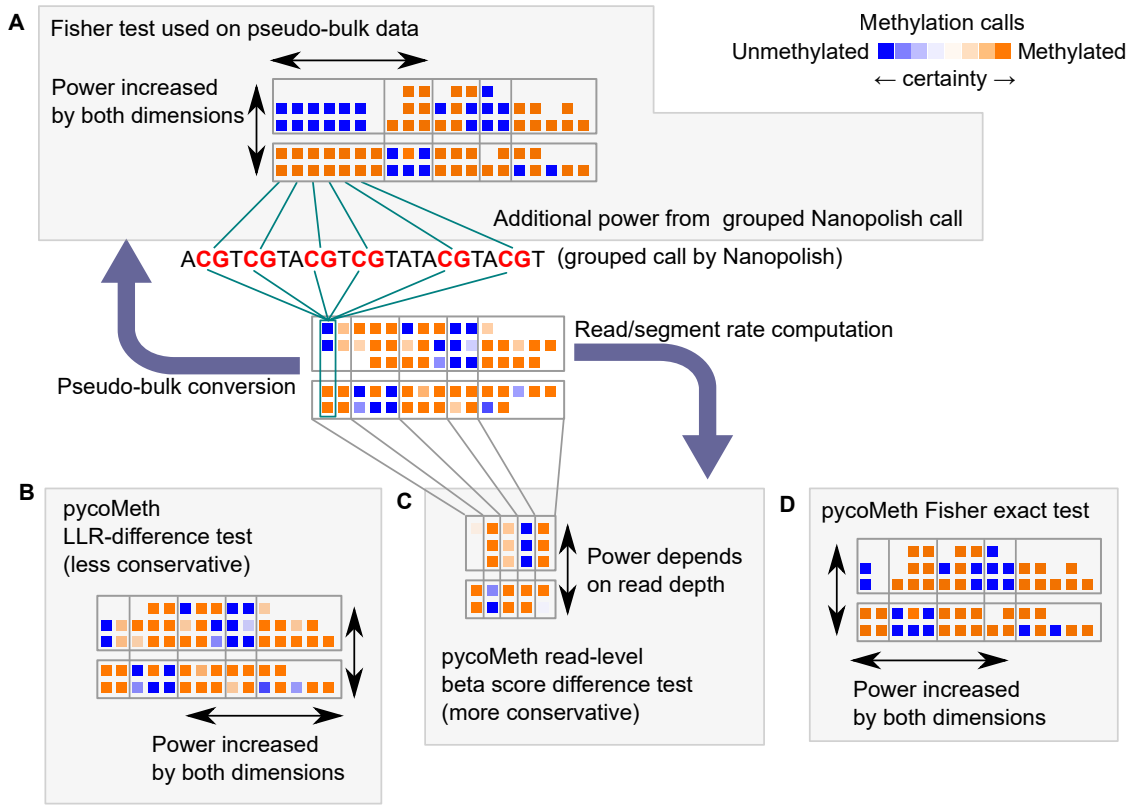

Figure C.4.: Confounders of test power in the DMR tests implemented in pycoMeth and in bisulfite sequencing. Figure adapted from the pycoMeth publication [207]. **A)** When Nanopore methylation calls are converted to pseudo-bulk such that tools designed for bisulfite sequencing can be applied, grouped Nanopolish calls may inflate testing power. The Fisher-Exact test also draws power from both read-depth and segment size. **B)** The LLR difference hypothesis implemented in pycoMeth version 1 and 2 draws power from both read-depth and segment size. **C)** The BS difference hypothesis implemented in pycoMeth version 2 draws power only from read-depth and therefore represents the most conservative test implemented in pycoMeth. **D)** The count dependency hypothesis implemented in pycoMeth performs a Fisher-Exact or χ^2 test and draws power from both dimensions as well.

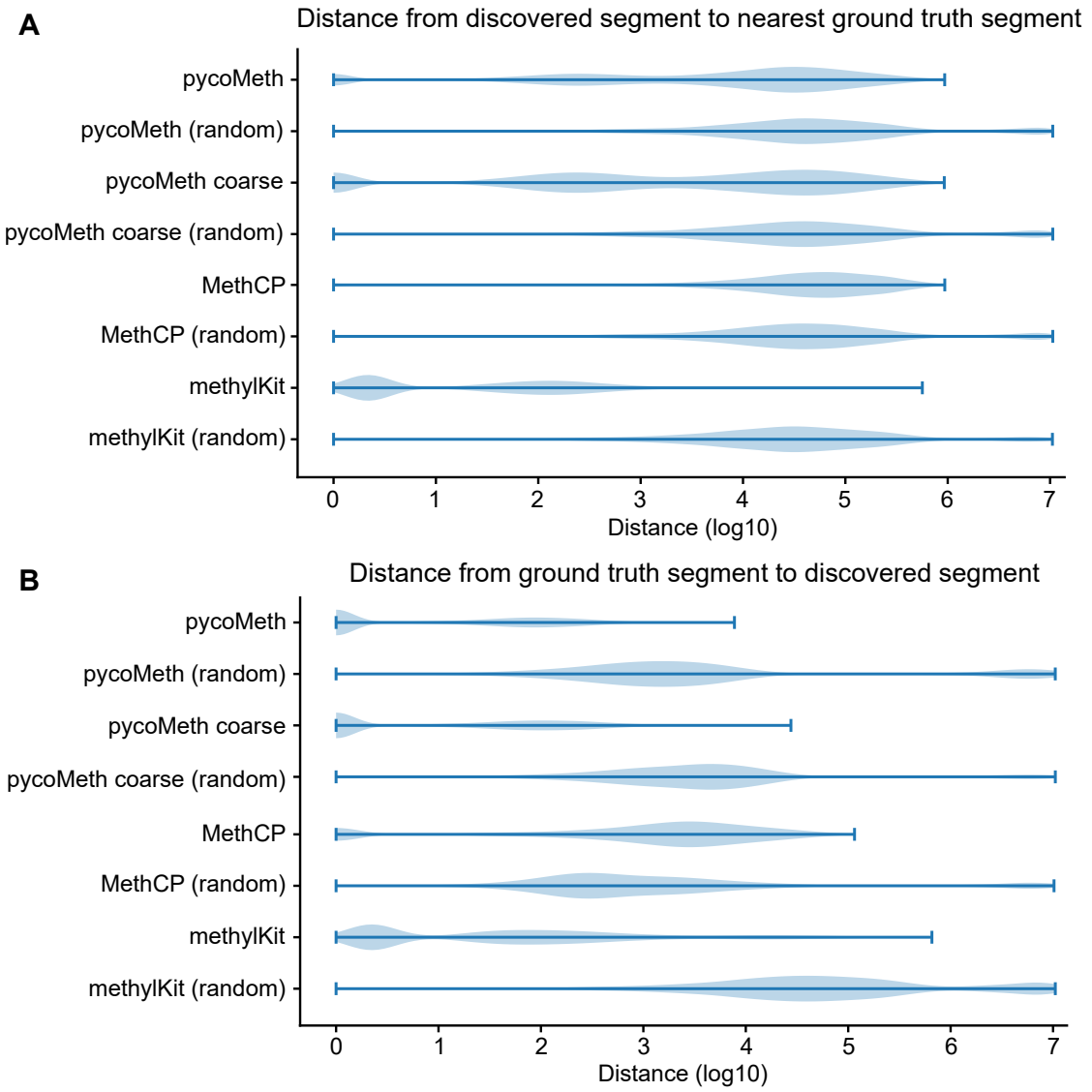

Figure C.5.: Comparing segmentations performed on simulated Nanopolish methylation calls with permuted segmentations. Rows with "(random)" represent segmentations in which the original segmentation had the order of segments randomized, while keeping number of segments and segment lengths the same. This figure shows that all segmentations perform better than random with the same distribution of segment lengths. Figure adapted from the pycoMeth publication [207]. **A)** Distance from discovered segment to nearest ground truth segment. **B)** Distance from ground truth segment to nearest discovered segment.

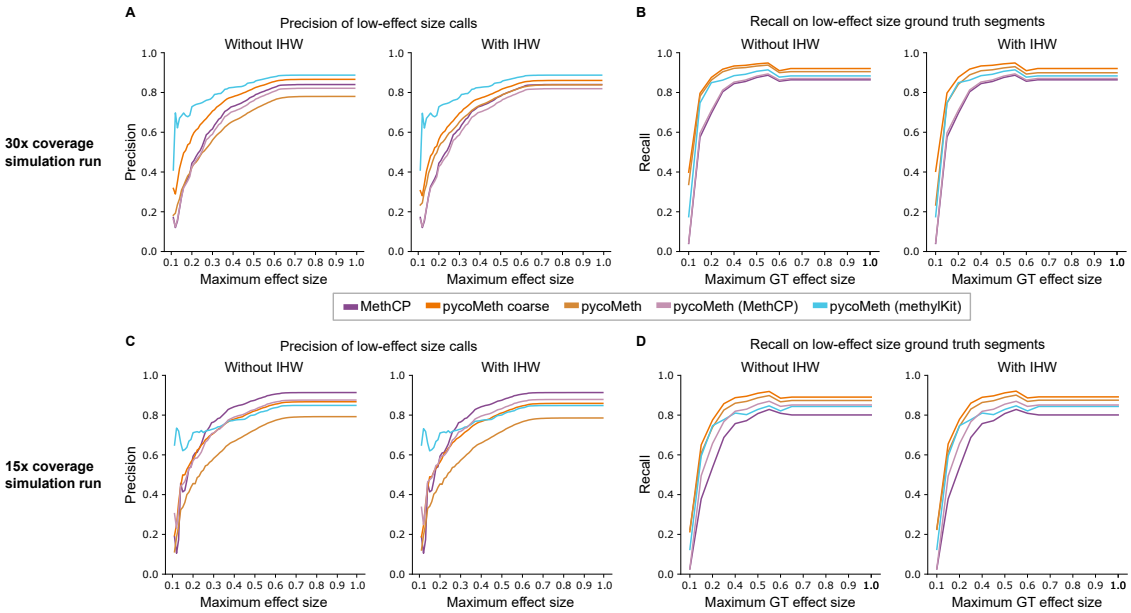

Figure C.6.: Investigating precision and recall in low-effect-size DMRs in simulated data. For the computation of recall, ground-truth DMRs were filtered based on the maximum effect-size on the X-axis, while for the computation of precision predicted DMRs were filtered. Figure adapted from the pycoMeth publication [207]

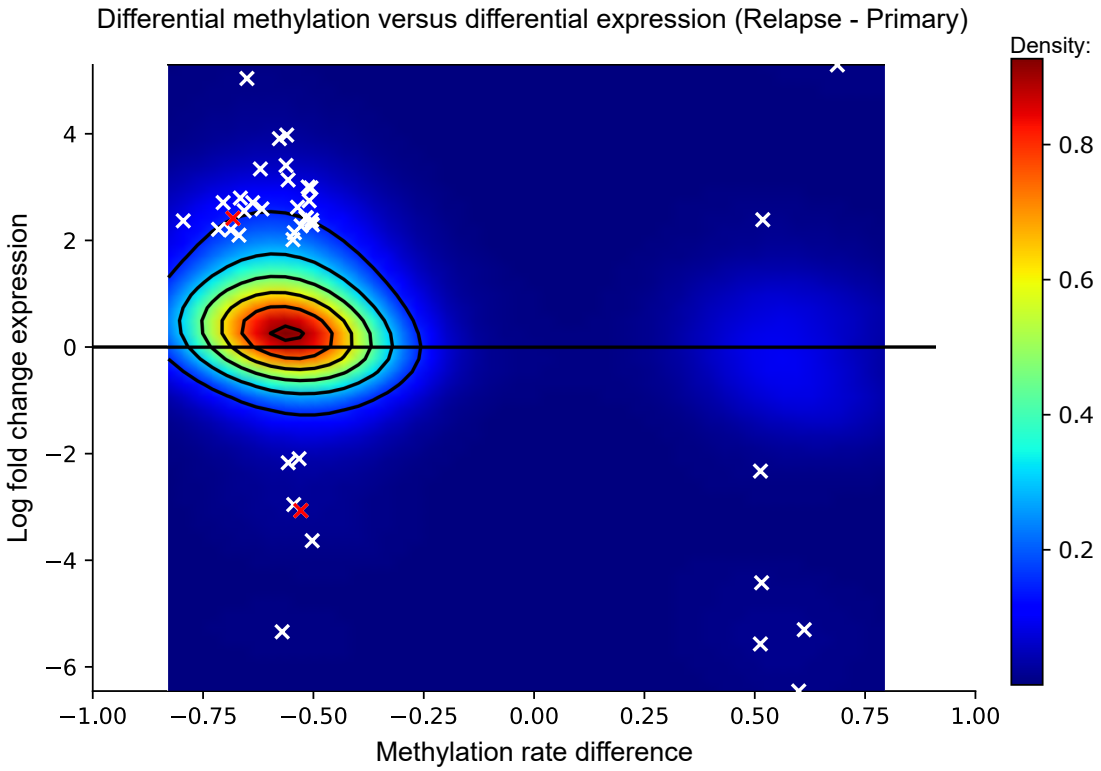

Figure C.7.: Figure adapted from [187]. Correlation between differential promoter methylation and differential expression, comparing primary tumor before treatment and relapse tumor. White crosses show genes with strong differential expression (absolute log-fold change > 2) while red crosses additionally exhibit increased copy-number in primary tumor. Figure adapted from the medulloblastoma publication [187]

Acknowledgements

My thanks go to my advisors, Marc Jan Bonder and Oliver Stegle, for their continued support and guidance throughout this work. Furthermore, I want to thank my other Thesis Advisory Committee members Lennart Hilbert, Daniel Hübschmann, and Matthias Schlesner. Many thanks also to all our collaboration partners, primarily to Mathias Boulanger, Aurélie Ernst, Arnaud Krebs, Adrien Leger, and Tobias Rausch. I am also very grateful to everyone in the Stegle group for fostering such a supportive and warm work environment. Special thanks also to Ines Reinartz for encouraging me to see this through to the end. Most of all, however, I want to thank my wife, Kay, for the continued support and encouragement.

Last, but not least, many thanks to all donors who have contributed their biological material for the pursuit of research.

Bibliography

[1] Quentin Gouil and Andrew Keniry. “Latest techniques to study DNA methylation”. en. In: *Essays Biochem.* 63.6 (Dec. 2019), pp. 639–648.

[2] C H Waddington. “The epigenotype. 1942”. en. In: *Int. J. Epidemiol.* 41.1 (Feb. 2012), pp. 10–13.

[3] Agustina D’Urso and Jason H Brickner. “Mechanisms of epigenetic memory”. en. In: *Trends Genet.* 30.6 (June 2014), pp. 230–236.

[4] Rosalind M John and Claire Rougeulle. “Developmental Epigenetics: Phenotype and the Flexible Epigenome”. en. In: *Front Cell Dev Biol* 6 (Oct. 2018), p. 130.

[5] Zuyun Liu and Yimin Zhu. “Epigenetic clock: a promising mirror of ageing”. en. In: *Lancet Healthy Longev* 2.6 (June 2021), e304–e305.

[6] Anders Berglund et al. “Epigenetic dysregulation of immune-related pathways in cancer: bioinformatics tools and visualization”. en. In: *Exp. Mol. Med.* 53.5 (May 2021), pp. 761–771.

[7] Douglas Hanahan. “Hallmarks of Cancer: New Dimensions”. en. In: *Cancer Discov.* 12.1 (Jan. 2022), pp. 31–46.

[8] Douglas Hanahan and Robert A Weinberg. “Hallmarks of cancer: the next generation”. en. In: *Cell* 144.5 (Mar. 2011), pp. 646–674.

[9] Nadine Darwiche. “Epigenetic mechanisms and the hallmarks of cancer: an intimate affair”. en. In: *Am. J. Cancer Res.* 10.7 (July 2020), pp. 1954–1978.

[10] Yuanjun Lu et al. “Epigenetic regulation in human cancer: the potential role of epi-drug in cancer therapy”. en. In: *Mol. Cancer* 19.1 (Apr. 2020), p. 79.

[11] En Li and Yi Zhang. “DNA methylation in mammals”. en. In: *Cold Spring Harb. Perspect. Biol.* 6.5 (May 2014), a019133.

[12] Treat B Johnson and Robert D Coghill. “RESEARCHES ON PYRIMIDINES. C111. THE DISCOVERY OF 5-METHYL-CYTOSINE IN TU-

BERCULINIC ACID, THE NUCLEIC ACID OF THE TUBERCLE BACILLUS1". In: *J. Am. Chem. Soc.* 47.11 (Nov. 1925), pp. 2838–2844.

[13] Andjela Rodic et al. "Understanding key features of bacterial restriction-modification systems through quantitative modeling". en. In: *BMC Syst. Biol.* 11.Suppl 1 (Feb. 2017), p. 377.

[14] Victor V Zinoviev et al. "Symmetry elements in DNA structure important for recognition/methylation by DNA [amino]-methyltransferases". en. In: *Nucleic Acids Res.* 32.13 (July 2004), pp. 3930–3934.

[15] Geoffrey G Wilson and Noreen E Murray. "RESTRICTION AND MODIFICATION SYSTEMS". en. In: (Nov. 2003).

[16] Wil A M Loenen and Elisabeth A Raleigh. "The other face of restriction: modification-dependent enzymes". en. In: *Nucleic Acids Res.* 42.1 (Jan. 2014), pp. 56–69.

[17] M G Marinus and A Løbner-Olesen. "DNA Methylation". en. In: *EcoSal Plus* 6.1 (May 2014).

[18] P Polaczek, K Kwan, and J L Campbell. "GATC motifs may alter the conformation of DNA depending on sequence context and N6-adenine methylation status: possible implications for DNA-protein recognition". en. In: *Mol. Gen. Genet.* 258.5 (June 1998), pp. 488–493.

[19] Diego Gonzalez et al. "The functions of DNA methylation by CcrM in Caulobacter crescentus: a global approach". en. In: *Nucleic Acids Res.* 42.6 (Apr. 2014), pp. 3720–3735.

[20] Jue D Wang and Petra A Levin. "Metabolism, cell growth and the bacterial cell cycle". en. In: *Nat. Rev. Microbiol.* 7.11 (Nov. 2009), pp. 822–827.

[21] Marek Drozdz et al. "Novel non-specific DNA adenine methyltransferases". en. In: *Nucleic Acids Res.* 40.5 (Mar. 2012), pp. 2119–2130.

[22] Iain A Murray et al. "The non-specific adenine DNA methyltransferase M.EcoGII". en. In: *Nucleic Acids Res.* 46.2 (Jan. 2018), pp. 840–848.

[23] Bi-Feng Yuan. "5-methylcytosine and its derivatives". en. In: *Adv. Clin. Chem.* 67 (Nov. 2014), pp. 151–187.

[24] Robert A Gaultney et al. "4-Methylcytosine DNA modification is critical for global epigenetic regulation and virulence in the human pathogen Leptospira interrogans". en. In: *Nucleic Acids Res.* 48.21 (Dec. 2020), pp. 12102–12115.

[25] Josep Casadesús and David Low. "Epigenetic gene regulation in the bacterial world". en. In: *Microbiol. Mol. Biol. Rev.* 70.3 (Sept. 2006), pp. 830–856.

[26] Kevin T Militello et al. "Conservation of Dcm-mediated cytosine DNA methylation in Escherichia coli". en. In: *FEMS Microbiol. Lett.* 328.1 (Mar. 2012), pp. 78–85.

[27] M Xu et al. "Cloning, characterization and expression of the gene coding for a cytosine-5-DNA methyltransferase recognizing GpC". en. In: *Nucleic Acids Res.* 26.17 (Sept. 1998), pp. 3961–3966.

[28] Theresa K Kelly et al. "Genome-wide mapping of nucleosome positioning and DNA methylation within individual DNA molecules". en. In: *Genome Res.* 22.12 (Dec. 2012), pp. 2497–2506.

[29] Monique G P van der Wijst et al. "Experimental mitochondria-targeted DNA methylation identifies GpC methylation, not CpG methylation, as potential regulator of mitochondrial gene expression". en. In: *Sci. Rep.* 7.1 (Mar. 2017), p. 177.

[30] M Okano et al. "DNA methyltransferases Dnmt3a and Dnmt3b are essential for de novo methylation and mammalian development". en. In: *Cell* 99.3 (Oct. 1999), pp. 247–257.

[31] Waddington. "The strategy of the genes. A discussion of some aspects of theoretical biology. With an appendix by H. Kacser". In: *The strategy of the genes. A discussion of some aspects of theoretical biology. With an appendix by H. Kacser.* ().

[32] T Naveh-Many and H Cedar. "Active gene sequences are undermethylated". en. In: *Proc. Natl. Acad. Sci. U. S. A.* 78.7 (July 1981), pp. 4246–4250.

[33] Atsushi Onodera et al. "Roles of TET and TDG in DNA demethylation in proliferating and non-proliferating immune cells". en. In: *Genome Biol.* 22.1 (June 2021), p. 186.

[34] Atanu Maiti and Alexander C Drohat. "Thymine DNA glycosylase can rapidly excise 5-formylcytosine and 5-carboxylcytosine: potential implications for active demethylation of CpG sites". en. In: *J. Biol. Chem.* 286.41 (Oct. 2011), pp. 35334–35338.

[35] Maria A Hahn, Piroska E Szabó, and Gerd P Pfeifer. "5-Hydroxymethylcytosine: a stable or transient DNA modification?" en. In: *Genomics* 104.5 (Nov. 2014), pp. 314–323.

[36] Chun-Xiao Song and Chuan He. "Potential functional roles of DNA demethylation intermediates". en. In: *Trends Biochem. Sci.* 38.10 (Oct. 2013), pp. 480–484.

[37] A Bird et al. “A fraction of the mouse genome that is derived from islands of nonmethylated, CpG-rich DNA”. en. In: *Cell* 40.1 (Jan. 1985), pp. 91–99.

[38] M McClelland and R Ivarie. “Asymmetrical distribution of CpG in an ’average’ mammalian gene”. en. In: *Nucleic Acids Res.* 10.23 (Dec. 1982), pp. 7865–7877.

[39] Francisco Antequera and Adrian Bird. “CpG Islands: A Historical Perspective”. In: *CpG Islands: Methods and Protocols*. Ed. by Tanya Vavouri and Miguel A Peinado. New York, NY: Springer New York, 2018, pp. 3–13.

[40] Karolina Janitz and Michal Janitz. “Chapter 12 - Assessing Epigenetic Information”. In: *Handbook of Epigenetics*. Ed. by Trygve Tollefsbol. San Diego: Academic Press, Jan. 2011, pp. 173–181.

[41] Rafael A Irizarry et al. “The human colon cancer methylome shows similar hypo- and hypermethylation at conserved tissue-specific CpG island shores”. en. In: *Nat. Genet.* 41.2 (Feb. 2009), pp. 178–186.

[42] M Gardiner-Garden and M Frommer. “CpG islands in vertebrate genomes”. en. In: *J. Mol. Biol.* 196.2 (July 1987), pp. 261–282.

[43] Juan Sandoval et al. “Validation of a DNA methylation microarray for 450,000 CpG sites in the human genome”. en. In: *Epigenetics* 6.6 (June 2011), pp. 692–702.

[44] María Del Carmen Turpín-Sevilla et al. “Global Methylome Scores Correlate with Histological Subtypes of Colorectal Carcinoma and Show Different Associations with Common Clinical and Molecular Features”. en. In: *Cancers* 13.20 (Oct. 2021).

[45] Haider M Hassan et al. “Regulation of Active DNA Demethylation through RAR-Mediated Recruitment of a TET/TDG Complex”. en. In: *Cell Rep.* 19.8 (May 2017), pp. 1685–1697.

[46] Barbara Panning. “X-chromosome inactivation: the molecular basis of silencing”. en. In: *J. Biol.* 7.8 (Oct. 2008), p. 30.

[47] J Boyes and A Bird. “Repression of genes by DNA methylation depends on CpG density and promoter strength: evidence for involvement of a methyl-CpG binding protein”. en. In: *EMBO J.* 11.1 (Jan. 1992), pp. 327–333.

[48] Ryan Lister et al. “Human DNA methylomes at base resolution show widespread epigenomic differences”. en. In: *Nature* 462.7271 (Nov. 2009), pp. 315–322.

[49] F Watt and P L Molloy. "Cytosine methylation prevents binding to DNA of a HeLa cell transcription factor required for optimal expression of the adenovirus major late promoter". en. In: *Genes Dev.* 2.9 (Sept. 1988), pp. 1136–1143.

[50] Mun-Kit Choy et al. "Genome-wide conserved consensus transcription factor binding motifs are hyper-methylated". en. In: *BMC Genomics* 11 (Sept. 2010), p. 519.

[51] Michela Curradi et al. "Molecular mechanisms of gene silencing mediated by DNA methylation". en. In: *Mol. Cell. Biol.* 22.9 (May 2002), pp. 3157–3173.

[52] Yimeng Yin et al. "Impact of cytosine methylation on DNA binding specificities of human transcription factors". en. In: *Science* 356.6337 (May 2017).

[53] Jun Wan et al. "Characterization of tissue-specific differential DNA methylation suggests distinct modes of positive and negative gene expression regulation". en. In: *BMC Genomics* 16 (Feb. 2015), p. 49.

[54] Ieva Rauluseviciute, Finn Drabløs, and Morten Beck Rye. "DNA hypermethylation associated with upregulated gene expression in prostate cancer demonstrates the diversity of epigenetic regulation". en. In: *BMC Med. Genomics* 13.1 (Jan. 2020), p. 6.

[55] Gwendal Dujardin et al. "How slow RNA polymerase II elongation favors alternative exon skipping". en. In: *Mol. Cell* 54.4 (May 2014), pp. 683–690.

[56] Galit Lev Maor, Ahuvi Yearim, and Gil Ast. "The alternative role of DNA methylation in splicing regulation". en. In: *Trends Genet.* 31.5 (May 2015), pp. 274–280.

[57] Kai Song, Li Li, and Guofan Zhang. "The association between DNA methylation and exon expression in the Pacific oyster Crassostrea gigas". en. In: *PLoS One* 12.9 (Sept. 2017), e0185224.

[58] Sanjeev Shukla et al. "CTCF-promoted RNA polymerase II pausing links DNA methylation to splicing". en. In: *Nature* 479.7371 (Nov. 2011), pp. 74–79.

[59] Lauren A Urban et al. "The impact of age-related hypomethylated DNA on immune signaling upon cellular demise". en. In: *Trends Immunol.* 42.6 (June 2021), pp. 464–468.

[60] Zhiqian Tong et al. "Age-related formaldehyde interferes with DNA methyltransferase function, causing memory loss in Alzheimer's disease". en. In: *Neurobiol. Aging* 36.1 (Jan. 2015), pp. 100–110.

[61] Fariha Angum et al. "The Prevalence of Autoimmune Disorders in Women: A Narrative Review". en. In: *Cureus* 12.5 (May 2020), e8094.

[62] Zhuo Sun, Jinbo Fan, and Yang Wang. "X-Chromosome Inactivation and Related Diseases". en. In: *Genet. Res.* 2022 (Mar. 2022), p. 1391807.

[63] Mohammad Javad Mousavi, Mahdi Mahmoudi, and Somayeh Ghotloo. "Escape from X chromosome inactivation and female bias of autoimmune diseases". en. In: *Mol. Med.* 26.1 (Dec. 2020), p. 127.

[64] Zelin Jin and Yun Liu. "DNA methylation in human diseases". en. In: *Genes Dis* 5.1 (Mar. 2018), pp. 1–8.

[65] Chai-Jin Lee et al. "Impact of mutations in DNA methylation modification genes on genome-wide methylation landscapes and downstream gene activations in pan-cancer". en. In: *BMC Med. Genomics* 13.Suppl 3 (Feb. 2020), p. 27.

[66] Warwick J Locke et al. "DNA Methylation Cancer Biomarkers: Translation to the Clinic". en. In: *Front. Genet.* 10 (Nov. 2019), p. 1150.

[67] Melanie Ehrlich. "DNA hypomethylation in cancer cells". en. In: *Epigenomics* 1.2 (Dec. 2009), pp. 239–259.

[68] Jason P Ross, Keith N Rand, and Peter L Molloy. "Hypomethylation of repeated DNA sequences in cancer". en. In: *Epigenomics* 2.2 (Apr. 2010), pp. 245–269.

[69] Geoffrey M Cooper. *Tumor Suppressor Genes*. Sinauer Associates, 2000.

[70] Karyn L Sheaffer, Ellen N Elliott, and Klaus H Kaestner. "DNA Hypomethylation Contributes to Genomic Instability and Intestinal Cancer Initiation". en. In: *Cancer Prev. Res.* 9.7 (July 2016), pp. 534–546.

[71] Alexandros Daskalos et al. "Hypomethylation of retrotransposable elements correlates with genomic instability in non-small cell lung cancer". en. In: *Int. J. Cancer* 124.1 (Jan. 2009), pp. 81–87.

[72] Rosa Visone et al. "DNA methylation of shelf, shore and open sea CpG positions distinguish high microsatellite instability from low or stable microsatellite status colon cancer stem cells". en. In: *Epigenomics* 11.6 (May 2019), pp. 587–604.

[73] Alex R D Delbridge, Liz J Valente, and Andreas Strasser. "The role of the apoptotic machinery in tumor suppression". en. In: *Cold Spring Harb. Perspect. Biol.* 4.11 (Nov. 2012).

[74] Jae-Won Cho et al. "The importance of enhancer methylation for epigenetic regulation of tumorigenesis in squamous lung cancer". en. In: *Exp. Mol. Med.* 54.1 (Jan. 2022), pp. 12–22.

[75] Rachel E Bell et al. "Enhancer methylation dynamics contribute to cancer plasticity and patient mortality". en. In: *Genome Res.* 26.5 (May 2016), pp. 601–611.

[76] Atsuya Nishiyama and Makoto Nakanishi. "Navigating the DNA methylation landscape of cancer". en. In: *Trends Genet.* 37.11 (Nov. 2021), pp. 1012–1027.

[77] Catherine Do et al. "Allele-specific DNA methylation is increased in cancers and its dense mapping in normal plus neoplastic cells increases the yield of disease-associated regulatory SNPs". en. In: *Genome Biol.* 21.1 (June 2020), p. 153.

[78] Miles C Benton et al. "Genome-wide allele-specific methylation is enriched at gene regulatory regions in a multi-generation pedigree from the Norfolk Island isolate". en. In: *Epigenetics Chromatin* 12.1 (Oct. 2019), p. 60.

[79] John N Hutchinson et al. "Allele-specific methylation occurs at genetic variants associated with complex disease". en. In: *PLoS One* 9.6 (June 2014), e98464.

[80] Moshe Szyf. "DNA methylation and cancer therapy". en. In: *Drug Resist. Updat.* 6.6 (Dec. 2003), pp. 341–353.

[81] Jean-Pierre J Issa. "DNA methylation as a therapeutic target in cancer". en. In: *Clin. Cancer Res.* 13.6 (Mar. 2007), pp. 1634–1637.

[82] Yuan Cheng et al. "Targeting epigenetic regulators for cancer therapy: mechanisms and advances in clinical trials". en. In: *Signal Transduct Target Ther* 4 (Dec. 2019), p. 62.

[83] Victor V Levenson. "DNA methylation as a universal biomarker". en. In: *Expert Rev. Mol. Diagn.* 10.4 (May 2010), pp. 481–488.

[84] Changshun Yang et al. "Molecular subtypes based on DNA methylation predict prognosis in colon adenocarcinoma patients". en. In: *Aging* 11.24 (Dec. 2019), pp. 11880–11892.

[85] Paul A Northcott et al. "Medulloblastoma comprises four distinct molecular variants". en. In: *J. Clin. Oncol.* 29.11 (Apr. 2011), pp. 1408–1414.

[86] Kimberly D Siegmund et al. "Inferring clonal expansion and cancer stem cell dynamics from DNA methylation patterns in colorectal cancers". en. In: *Proc. Natl. Acad. Sci. U. S. A.* 106.12 (Mar. 2009), pp. 4828–4833.

[87] E A Klein et al. "Clinical validation of a targeted methylation-based multi-cancer early detection test using an independent validation set". en. In: *Ann. Oncol.* 32.9 (Sept. 2021), pp. 1167–1177.

[88] Chunlei Zheng and Rong Xu. "Predicting cancer origins with a DNA methylation-based deep neural network model". en. In: *PLoS One* 15.5 (May 2020), e0226461.

[89] Huiyan Luo et al. "Liquid Biopsy of Methylation Biomarkers in Cell-Free DNA". en. In: *Trends Mol. Med.* 27.5 (May 2021), pp. 482–500.

[90] Peter B Becker and Jerry L Workman. "Nucleosome remodeling and epigenetics". en. In: *Cold Spring Harb. Perspect. Biol.* 5.9 (Sept. 2013).

[91] Olivia Morrison and Jitendra Thakur. "Molecular Complexes at Euchromatin, Heterochromatin and Centromeric Chromatin". en. In: *Int. J. Mol. Sci.* 22.13 (June 2021).

[92] Palak Gujral et al. "Histone acetylation and the role of histone deacetylases in normal cyclic endometrium". en. In: *Reprod. Biol. Endocrinol.* 18.1 (Aug. 2020), p. 84.

[93] Jaime L Miller and Patrick A Grant. "The role of DNA methylation and histone modifications in transcriptional regulation in humans". en. In: *Subcell. Biochem.* 61 (2013), pp. 289–317.

[94] Monali NandyMazumdar et al. "Looping of upstream cis-regulatory elements is required for CFTR expression in human airway epithelial cells". en. In: *Nucleic Acids Res.* 48.7 (Apr. 2020), pp. 3513–3524.

[95] Can Sönmezer et al. "Molecular Co-occupancy Identifies Transcription Factor Binding Cooperativity In Vivo". en. In: *Mol. Cell* 81.2 (Jan. 2021), 255–267.e6.

[96] Chia-Yu Guh, Yu-Hung Hsieh, and Hsueh-Ping Chu. "Functions and properties of nuclear lncRNAs-from systematically mapping the interactomes of lncRNAs". en. In: *J. Biomed. Sci.* 27.1 (Mar. 2020), p. 44.

[97] D W Russell and R K Hirata. "The detection of extremely rare DNA modifications. Methylation in dam- and hsd- Escherichia coli strains". en. In: *J. Biol. Chem.* 264.18 (June 1989), pp. 10787–10794.

[98] Eugene J H Wee, Thu Ha Ngo, and Matt Trau. "Colorimetric detection of both total genomic and loci-specific DNA methylation from limited DNA inputs". en. In: *Clin. Epigenetics* 7 (July 2015), p. 65.

[99] Hikoya Hayatsu. "Discovery of bisulfite-mediated cytosine conversion to uracil, the key reaction for DNA methylation analysis–a personal account". en. In: *Proc. Jpn. Acad. Ser. B Phys. Biol. Sci.* 84.8 (2008), pp. 321–330.

[100] Chang Shu et al. "Comparison of methylation capture sequencing and Infinium MethylationEPIC array in peripheral blood mononuclear cells". en. In: *Epigenetics Chromatin* 13.1 (Nov. 2020), p. 51.

[101] Carl W Fuller et al. "The challenges of sequencing by synthesis". en. In: *Nat. Biotechnol.* 27.11 (Nov. 2009), pp. 1013–1023.

[102] Mathias Ehrich et al. "A new method for accurate assessment of DNA quality after bisulfite treatment". en. In: *Nucleic Acids Res.* 35.5 (Jan. 2007), e29.

[103] Fumihito Miura et al. "Amplification-free whole-genome bisulfite sequencing by post-bisulfite adaptor tagging". en. In: *Nucleic Acids Res.* 40.17 (Sept. 2012), e136.

[104] Suhua Feng et al. "Efficient and accurate determination of genome-wide DNA methylation patterns in Arabidopsis thaliana with enzymatic methyl sequencing". en. In: *Epigenetics Chromatin* 13.1 (Oct. 2020), p. 42.

[105] Hikoya Hayatsu. "The bisulfite genomic sequencing used in the analysis of epigenetic states, a technique in the emerging environmental genotoxicology research". en. In: *Mutat. Res.* 659.1-2 (July 2008), pp. 77–82.

[106] Felix Krueger and Simon R Andrews. "Bismark: a flexible aligner and methylation caller for Bisulfite-Seq applications". en. In: *Bioinformatics* 27.11 (June 2011), pp. 1571–1572.

[107] Ecseq Bioinformatics. *Why does the per base sequence quality decrease over the read in Illumina?* `https://www.ecseq.com/support/ngs/why-does-the-sequence-quality-decrease-over-the-read-in-illumina`. Accessed: 2022-10-10.

[108] Ge Tan et al. "Long fragments achieve lower base quality in Illumina paired-end sequencing". en. In: *Sci. Rep.* 9.1 (Feb. 2019), p. 2856.

[109] Quanhu Sheng et al. "Multi-perspective quality control of Illumina RNA sequencing data analysis". en. In: *Brief. Funct. Genomics* 16.4 (July 2017), pp. 194–204.

[110] Alexander Payne et al. "BulkVis: a graphical viewer for Oxford nanopore bulk FAST5 files". en. In: *Bioinformatics* 35.13 (July 2019), pp. 2193–2198.

[111] Yang Liu et al. "DNA methylation-calling tools for Oxford Nanopore sequencing: a survey and human epigenome-wide evaluation". en. In: *Genome Biol.* 22.1 (Oct. 2021), p. 295.

[112] John Eid et al. "Real-time DNA sequencing from single polymerase molecules". en. In: *Science* 323.5910 (Jan. 2009), pp. 133–138.

[113] Benjamin A Flusberg et al. "Direct detection of DNA methylation during single-molecule, real-time sequencing". en. In: *Nat. Methods* 7.6 (June 2010), pp. 461–465.

[114] Liesbeth Minnoye et al. "Chromatin accessibility profiling methods". en. In: *Nature Reviews Methods Primers* 1.1 (Jan. 2021), pp. 1–24.

[115] Zhenhai Zhang and B Franklin Pugh. "High-resolution genome-wide mapping of the primary structure of chromatin". en. In: *Cell* 144.2 (Jan. 2011), pp. 175–186.

[116] Maria Tsompana and Michael J Buck. "Chromatin accessibility: a window into the genome". en. In: *Epigenetics Chromatin* 7.1 (Nov. 2014), p. 33.

[117] Steven M Johnson et al. "Flexibility and constraint in the nucleosome core landscape of Caenorhabditis elegans chromatin". en. In: *Genome Res.* 16.12 (Dec. 2006), pp. 1505–1516.

[118] Jorja G Henikoff et al. "Epigenome characterization at single base-pair resolution". en. In: *Proc. Natl. Acad. Sci. U. S. A.* 108.45 (Nov. 2011), pp. 18318–18323.

[119] Lingyun Song and Gregory E Crawford. "DNase-seq: a high-resolution technique for mapping active gene regulatory elements across the genome from mammalian cells". en. In: *Cold Spring Harb. Protoc.* 2010.2 (Feb. 2010), db.prot5384.

[120] Jason D Buenrostro et al. "Transposition of native chromatin for fast and sensitive epigenomic profiling of open chromatin, DNA-binding proteins and nucleosome position". en. In: *Nat. Methods* 10.12 (Dec. 2013), pp. 1213–1218.

[121] Hashem Koohy et al. "A comparison of peak callers used for DNase-Seq data". en. In: *PLoS One* 9.5 (May 2014), e96303.

[122] Evan D Tarbell and Tao Liu. "HMMRATAC: a Hidden Markov ModeleR for ATAC-seq". en. In: *Nucleic Acids Res.* 47.16 (Sept. 2019), e91.

[123] Fides D Lay, Theresa K Kelly, and Peter A Jones. "Nucleosome Occupancy and Methylome Sequencing (NOMe-seq)". en. In: *Methods Mol. Biol.* 1708 (2018), pp. 267–284.

[124] Arnaud R Krebs et al. "Genome-wide Single-Molecule Footprinting Reveals High RNA Polymerase II Turnover at Paused Promoters". en. In: *Mol. Cell* 67.3 (Aug. 2017), 411–422.e4.

[125] Isac Lee et al. "Simultaneous profiling of chromatin accessibility and methylation on human cell lines with nanopore sequencing". en. In: *Nat. Methods* 17.12 (Dec. 2020), pp. 1191–1199.

[126] Zohar Shipony et al. "Long-range single-molecule mapping of chromatin accessibility in eukaryotes". en. In: *Nat. Methods* 17.3 (Mar. 2020), pp. 319–327.

[127] Iqbal H Sarker. "Machine Learning: Algorithms, Real-World Applications and Research Directions". en. In: *SN Comput Sci* 2.3 (Mar. 2021), p. 160.

[128] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. en. MIT Press, Nov. 2016.

[129] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. "Variational Inference: A Review for Statisticians". In: *J. Am. Stat. Assoc.* 112.518 (Apr. 2017), pp. 859–877.

[130] Christian P Robert and Sylvia Richardson. "Markov Chain Monte Carlo Methods". In: *Discretization and MCMC Convergence Assessment*. Ed. by Christian P Robert. New York, NY: Springer New York, 1998, pp. 1–25.

[131] Rens van de Schoot et al. "Bayesian statistics and modelling". en. In: *Nature Reviews Methods Primers* 1.1 (Jan. 2021), pp. 1–26.

[132] Markov. "Rasprostranenie zakona bol'shih chisel na velichiny, zavisyaschie drug ot druga". In: *Izvestiya Fiziko-matematicheskogo obschestva pri* (1906).

[133] Leonard E Baum and Ted Petrie. "Statistical inference for probabilistic functions of finite state Markov chains". In: *Ann. Math. Stat.* 37.6 (Dec. 1966), pp. 1554–1563.

[134] Sean R Eddy. "What is a hidden Markov model?" en. In: *Nat. Biotechnol.* 22.10 (Oct. 2004), pp. 1315–1316.

[135] R L Stratonovich. "CONDITIONAL MARKOV PROCESSES**Teor. ver. i yeye primen., Akad. nauk SSSR., 5, No. 2 172 (1960). Translated by O. M. Blunn.††The part dealing with continuous time is not altogether convincing, but it has been allowed to stand in its original form in view of the great interest in the problems which are posed [Russian Editor]". In: *Non-Linear Transformations of Stochastic Processes*. Elsevier, 1965, pp. 427–453.

[136] "Viterbi Algorithm". In: *Encyclopedia of Machine Learning*. Ed. by Claude Sammut and Geoffrey I Webb. Boston, MA: Springer US, 2010, pp. 1025–1025.

[137] L R Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". In: *Proc. IEEE* 77.2 (Feb. 1989), pp. 257–286.

[138] T K Moon. "The expectation-maximization algorithm". In: *IEEE Signal Process. Mag.* 13.6 (Nov. 1996), pp. 47–60.

[139] Moloud Abdar et al. "A review of uncertainty quantification in deep learning: Techniques, applications and challenges". In: *Inf. Fusion* 76 (Dec. 2021), pp. 243–297.

[140] Ian Farrance and Robert Frenkel. "Uncertainty of Measurement: A Review of the Rules for Calculating Uncertainty Components through Functional Relationships". en. In: *Clin. Biochem. Rev.* 33.2 (May 2012), pp. 49–75.

[141] Martyn Pickersgill. "Epistemic Modesty, Ostentatiousness and the Uncertainties of Epigenetics: On the Knowledge Machinery of (Social) Science". In: *Sociol. Rev.* 64.1_suppl (Mar. 2016), pp. 186–202.

[142] Sascha Ranftl and Wolfgang von der Linden. "Bayesian Surrogate Analysis and Uncertainty Propagation". en. In: *Physical Sciences Forum* 3.1 (Nov. 2021), p. 6.

[143] F Rosenblatt. "The perceptron: a probabilistic model for information storage and organization in the brain". en. In: *Psychol. Rev.* 65.6 (Nov. 1958), pp. 386–408.

[144] Henry J Kelley. "Gradient theory of optimal flight paths". en. In: *ARS j.* 30.10 (Oct. 1960), pp. 947–954.

[145] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. "Learning representations by back-propagating errors". en. In: *Nature* 323.6088 (Oct. 1986), pp. 533–536.

[146] P J G Lisboa and S J Perantonis. "Complete solution of the local minima in the XOR problem". en. In: *Network* 2.1 (Feb. 1991), p. 119.

[147] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. "Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks". In: *Neural Netw.* 3.5 (Jan. 1990), pp. 551–560.

[148] Matthew E Peters et al. "Deep contextualized word representations". In: (Feb. 2018). arXiv: `1802.05365 [cs.CL]`.

[149] Zhen Shen, Wenzheng Bao, and De-Shuang Huang. "Recurrent Neural Network for Predicting Transcription Factor Binding Sites". en. In: *Sci. Rep.* 8.1 (Oct. 2018), p. 15270.

[150] Noopur Singh, Ravindra Nath, and Dev Bukhsh Singh. "Splice-site identification for exon prediction using bidirectional LSTM-RNN approach". en. In: *Biochem Biophys Rep* 30 (July 2022), p. 101285.

[151] Sunitha Basodi et al. "Gradient amplification: An efficient way to train deep neural networks". In: *Big Data Mining and Analytics* 3.3 (Sept. 2020), pp. 196–207.

[152] Ashish Vaswani et al. "Attention Is All You Need". In: (June 2017). arXiv: `1706.03762 [cs.CL]`.

[153] Jinhyuk Lee et al. "BioBERT: a pre-trained biomedical language representation model for biomedical text mining". In: (Jan. 2019). arXiv: `1901.08746 [cs.CL]`.

[154] Vladimir N Babenko, Irina V Chadaeva, and Yuriy L Orlov. "Genomic landscape of CpG rich elements in human". en. In: *BMC Evol. Biol.* 17.Suppl 1 (Feb. 2017), p. 19.

[155] Danuta M Jeziorska et al. "DNA methylation of intragenic CpG islands depends on their transcriptional activity during differentiation and disease". en. In: *Proc. Natl. Acad. Sci. U. S. A.* 114.36 (Sept. 2017), E7526–E7535.

[156] Bruce Alberts et al. *Chromosomal DNA and Its Packaging in the Chromatin Fiber*. Garland Science, 2002.

[157] Yan Zhang et al. "HHMD: the human histone modification database". en. In: *Nucleic Acids Res.* 38.Database issue (Jan. 2010), pp. D149–54.

[158] Steven L Salzberg. "Open questions: How many genes do we have?" en. In: *BMC Biol.* 16.1 (Aug. 2018), p. 94.

[159] Zaka Wing-Sze Yuen et al. "Systematic benchmarking of tools for CpG methylation detection from nanopore sequencing". en. In: *Nat. Commun.* 12.1 (June 2021), p. 3438.

[160] Peng Ni et al. "DNA 5-methylcytosine detection and methylation phasing using PacBio circular consensus sequencing". en. Mar. 2022.

[161] Shuying Sun and Xiaoqing Yu. "HMM-Fisher: identifying differential methylation using a hidden Markov model and Fisher's exact test". en. In: *Stat. Appl. Genet. Mol. Biol.* 15.1 (Mar. 2016), pp. 55–67.

[162] Boying Gong and Elizabeth Purdom. "MethCP: Differentially Methylated Region Detection with Change Point Models". en. In: *J. Comput. Biol.* 27.4 (Apr. 2020), pp. 458–471.

[163] Rajasekhar Kakumani, Omair Ahmad, and Vijay Devabhaktuni. "Identification of CpG islands in DNA sequences using statistically optimal null filters". en. In: *EURASIP J. Bioinform. Syst. Biol.* 2012.1 (Aug. 2012), p. 12.

[164] R Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria, 2021.

[165] Altuna Akalin et al. "methylKit: a comprehensive R package for the analysis of genome-wide DNA methylation profiles". en. In: *Genome Biol.* 13.10 (Oct. 2012), R87.

[166] *Recipes, scripts and genomics*. `http://zvfak.blogspot.com/2015/06/segmentation-of-methylation-profiles.html`. Accessed: 2022-10-11.

[167] Günter Klambauer et al. "cn.MOPS: mixture of Poissons for discovering copy number variations in next-generation sequencing data with a low false discovery rate". en. In: *Nucleic Acids Res.* 40.9 (May 2012), e69.

[168] P Baldi and A D Long. "A Bayesian framework for the analysis of microarray expression data: regularized t -test and statistical inferences of gene changes". en. In: *Bioinformatics* 17.6 (June 2001), pp. 509–519.

[169] Adam B Olshen et al. "Circular binary segmentation for the analysis of array-based DNA copy number data". en. In: *Biostatistics* 5.4 (Oct. 2004), pp. 557–572.

[170] The Minh Luong, Vittorio Perduca, and Gregory Nuel. "Hidden Markov Model Applications in Change-Point Analysis". In: (Dec. 2012). arXiv: `1212.1778 [stat.AP]`.

[171] Jason Ernst and Manolis Kellis. "Chromatin-state discovery and genome annotation with ChromHMM". en. In: *Nat. Protoc.* 12.12 (Dec. 2017), pp. 2478–2492.

[172] Medhat Mahmoud et al. "Structural variant calling: the long and the short of it". en. In: *Genome Biol.* 20.1 (Nov. 2019), p. 246.

[173] Ilanila Ilangumaran Ponmalar et al. "Pore Forming Protein Induced Biomembrane Reorganization and Dynamics: A Focused Review". en. In: *Front Mol Biosci* 8 (Sept. 2021), p. 737561.

[174] Ryan Wick. *Basecalling-comparison: A comparison of different Oxford Nanopore basecallers*. en.

[175] Alex Graves et al. "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks". In: *Proceedings of the 23rd international conference on Machine learning*. ICML '06. Pittsburgh, Pennsylvania, USA: Association for Computing Machinery, June 2006, pp. 369–376.

[176] Keiron O'Shea and Ryan Nash. "An Introduction to Convolutional Neural Networks". In: (Nov. 2015). arXiv: `1511.08458 [cs.NE]`.

[177] K Fukushima. "Neocognitron: a self organizing neural network model for a mechanism of pattern recognition unaffected by shift in position". en. In: *Biol. Cybern.* 36.4 (1980), pp. 193–202.

[178] *Accuracy*. en. `https://nanoporetech.com/accuracy`. Accessed: 2022-10-18.

[179] Marcus Stoiber et al. "De novo Identification of DNA Modifications Enabled by Genome-Guided Nanopore Signal Processing". en. Dec. 2016.

[180] Jared T Simpson et al. "Detecting DNA cytosine methylation using nanopore sequencing". en. In: *Nat. Methods* 14.4 (Apr. 2017), pp. 407–410.

[181] Peng Ni et al. "DeepSignal: detecting DNA methylation state from Nanopore sequencing reads using deep-learning". en. In: *Bioinformatics* (Apr. 2019).

[182] *tombo: Tombo is a suite of tools primarily for the identification of modified nucleotides from raw nanopore sequencing data*. en.

[183] Nicholas J Loman, Joshua Quick, and Jared T Simpson. "A complete bacterial genome assembled de novo using only nanopore sequencing data". en. In: *Nat. Methods* 12.8 (Aug. 2015), pp. 733–735.

[184] *megalodon: Megalodon is a research command line tool to extract high accuracy modified base and sequence variant calls from raw nanopore reads by anchoring the information rich basecalling neural network output to a reference genome/transriptome*. en.

[185] Andrey Malinin. "Uncertainty Estimation in Deep Learning with application to Spoken Language Assessment". In: (May 2019).

[186] *remora: Methylation/modified base calling separated from basecalling*. en.

[187] Tobias Rausch et al. "Long-read sequencing of diagnosis and post-therapy medulloblastoma reveals complex rearrangement patterns and epigenetic signatures". en. Feb. 2022.

[188] Rene Snajder et al. "pycoMeth: A toolbox for differential methylation testing from Nanopore methylation calls". en. Aug. 2022.

[189] Heng Li. "Tabix: fast retrieval of sequence features from generic TAB-delimited files". en. In: *Bioinformatics* 27.5 (Mar. 2011), pp. 718–719.

[190] Heng Li et al. "The Sequence Alignment/Map format and SAMtools". en. In: *Bioinformatics* 25.16 (Aug. 2009), pp. 2078–2079.

[191] *pysam: Pysam is a Python module for reading and manipulating SAM/BAM/VCF/BCF files. It's a lightweight wrapper of the htslib C-API, the same one that powers samtools, bcftools, and tabix*. en.

[192] Petr Danecek et al. "The variant call format and VCFtools". en. In: *Bioinformatics* 27.15 (Aug. 2011), pp. 2156–2158.

[193] Quincey Koziol and Dana Robinson. *HDF5*. [Computer Software] https://doi.org/10.11578/dc.20180330.1. Mar. 2018.

[194] Gabrielle Hartley and Rachel J O'Neill. "Centromere Repeats: Hidden Gems of the Genome". en. In: *Genes* 10.3 (Mar. 2019).

[195] Xena Giada Pappalardo and Viviana Barra. "Losing DNA methylation at repetitive elements and breaking bad". en. In: *Epigenetics Chromatin* 14.1 (June 2021), p. 25.

[196] Pauli Virtanen et al. "SciPy 1.0: fundamental algorithms for scientific computing in Python". en. In: *Nat. Methods* 17.3 (Mar. 2020), pp. 261–272.

[197] Andrew Collette et al. *h5py/h5py: 3.7.0*. May 2022.

[198] Charles R Harris et al. "Array programming with NumPy". en. In: *Nature* 585.7825 (Sept. 2020), pp. 357–362.

[199] McKinney. "pandas: a foundational Python library for data analysis and statistics". In: *Python for high performance and scientific computing* (2011).

[200] Andrew Collette. *Python and HDF5*. en. O'Reilly Media, Incorporated, 2013.

[201] C Bron. "Merge sort algorithm [M1]". In: *Commun. ACM* 15.5 (May 1972), pp. 357–358.

[202] James K Bonfield et al. "HTSlib: C library for reading/writing high-throughput sequencing data". en. In: *Gigascience* 10.2 (Feb. 2021).

[203] *modbam2bed*. en.

[204] Justin M Zook et al. "An open resource for accurately benchmarking small variant and reference calls". en. In: *Nat. Biotechnol.* 37.5 (May 2019), pp. 561–566.

[205] Valerie A Schneider et al. "Evaluation of GRCh38 and de novo haploid genome assemblies demonstrates the enduring quality of the reference assembly". en. In: *Genome Res.* 27.5 (May 2017), pp. 849–864.

[206] Heng Li. "Minimap2: pairwise alignment for nucleotide sequences". en. In: *Bioinformatics* 34.18 (Sept. 2018), pp. 3094–3100.

[207] Rene Snajder et al. "pycoMeth: A toolbox for differential methylation testing from Nanopore methylation calls". Feb. 2022.

[208] *liblzf*. `http://software.schmorp.de/pkg/liblzf.html`. Accessed: 2022-11-8.

[209] P Deutsch. *GZIP file format specification version 4.3*. Tech. rep. May 1996.

[210] R Snajder. *PMBio/MetH5Format*. 2022.

[211] Jinke Sui et al. "Discovery and validation of methylation signatures in blood-based circulating tumor cell-free DNA in early detection of colorectal carcinoma: a case-control study". en. In: *Clin. Epigenetics* 13.1 (Feb. 2021), p. 26.

[212] Shan V Andrews et al. "Case-control meta-analysis of blood DNA methylation and autism spectrum disorder". en. In: *Mol. Autism* 9 (June 2018), p. 40.

[213] Jiahui Si et al. "Epigenome-wide analysis of DNA methylation and coronary heart disease: a nested case-control study". en. In: *Elife* 10 (Sept. 2021).

[214] Skipper Seabold and Josef Perktold. "Statsmodels: Econometric and statistical modeling with python". In: *Proceedings of the 9th Python in Science Conference*. Austin, Texas: SciPy, 2010.

[215] Plotly Technologies Inc. "Collaborative data science". In: *Montreal: Plotly Technologies Inc Montral* (2015).

[216] Maarten van Iterson, Judith M Boer, and Renée X Menezes. "Filtering, FDR and power". en. In: *BMC Bioinformatics* 11 (Sept. 2010), p. 450.

[217] Nikolaos Ignatiadis et al. "Data-driven hypothesis weighting increases detection power in genome-scale multiple testing". en. In: *Nat. Methods* 13.7 (July 2016), pp. 577–580.

[218] Ren-Hua Chung and Chen-Yu Kang. "A multi-omics data simulator for complex disease studies and its application to evaluate multi-omics data analysis methods for disease classification". en. In: *Gigascience* 8.5 (May 2019).

[219] Thorsten Beier et al. "An Efficient Fusion Move Algorithm for the Minimum Cost Lifted Multicut Problem". In: *Computer Vision – ECCV 2016*. Springer International Publishing, 2016, pp. 715–730.

[220] Yongjun Piao et al. "Comprehensive Evaluation of Differential Methylation Analysis Methods for Bisulfite Sequencing Data". en. In: *Int. J. Environ. Res. Public Health* 18.15 (July 2021).

[221] Kasper D Hansen, Benjamin Langmead, and Rafael A Irizarry. "BSmooth: from whole genome bisulfite sequencing reads to differentially methylated regions". en. In: *Genome Biol.* 13.10 (Oct. 2012), R83.

[222] Yongseok Park et al. "MethylSig: a whole genome DNA methylation analysis pipeline". en. In: *Bioinformatics* 30.17 (Sept. 2014), pp. 2414–2422.

[223] Hao Feng, Karen N Conneely, and Hao Wu. "A Bayesian hierarchical model to detect differentially methylated loci from single nucleotide resolution sequencing data". en. In: *Nucleic Acids Res.* 42.8 (Apr. 2014), e69.

[224] Egor Dolzhenko and Andrew D Smith. "Using beta-binomial regression for high-precision differential methylation analysis in multifactor whole-genome bisulfite sequencing experiments". en. In: *BMC Bioinformatics* 15 (June 2014), p. 215.

[225] H B Mann and D R Whitney. "On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other". en. In: *aoms* 18.1 (Mar. 1947), pp. 50–60.

[226] William H Kruskal and W Allen Wallis. "Use of Ranks in One-Criterion Variance Analysis". In: *J. Am. Stat. Assoc.* 47.260 (1952), pp. 583–621.

[227] R A Fisher. "On the Interpretation of $\chi 2$ from Contingency Tables, and the Calculation of P". In: *J. R. Stat. Soc.* 85.1 (Jan. 1922), pp. 87–94.

[228] Karl Pearson. "X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 50.302 (July 1900), pp. 157–175.

[229] Alessio Farcomeni. "A review of modern multiple hypothesis testing, with particular attention to the false discovery proportion". en. In: *Stat. Methods Med. Res.* 17.4 (Aug. 2008), pp. 347–388.

[230] Carlo E Bonferroni. *Teoria statistica delle classi e calcolo delle probabilità*. it. Seeber, 1936.

[231] Zbyněk Šidák. "Rectangular Confidence Regions for the Means of Multivariate Normal Distributions". In: *J. Am. Stat. Assoc.* 62.318 (June 1967), pp. 626–633.

[232] Yoav Benjamini and Yosef Hochberg. "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing". In: *J. R. Stat. Soc. Series B Stat. Methodol.* 57.1 (1995), pp. 289–300.

[233] John D Storey. "The positive false discovery rate: a Bayesian interpretation and the q-value". en. In: *aos* 31.6 (Dec. 2003), pp. 2013–2035.

[234] Hong-Qiang Wang, Lindsey K Tuominen, and Chung-Jui Tsai. "SLIM: a sliding linear model for estimating the proportion of true null hypotheses in datasets with dependence structures". en. In: *Bioinformatics* 27.2 (Jan. 2011), pp. 225–231.

[235] Rene Snajder and Adrien Leger. *PMBio/pycoMeth*. 2022.

[236] Paul A Northcott et al. "Medulloblastoma". en. In: *Nat Rev Dis Primers* 5.1 (Feb. 2019), p. 11.

[237] Paul A Northcott et al. "Molecular subgroups of medulloblastoma". en. In: *Expert Rev. Neurother.* 12.7 (July 2012), pp. 871–884.

[238] Paul A Northcott et al. "The whole-genome landscape of medulloblastoma subtypes". en. In: *Nature* 547.7663 (July 2017), pp. 311–317.

[239] Tobias Rausch et al. "Genome sequencing of pediatric medulloblastoma links catastrophic DNA rearrangements with TP53 mutations". en. In: *Cell* 148.1-2 (Jan. 2012), pp. 59–71.

[240] Yilong Li et al. "Patterns of somatic structural variation in human cancer genomes". en. In: *Nature* 578.7793 (Feb. 2020), pp. 112–121.

[241] Felix Mölder et al. "Sustainable data analysis with Snakemake". en. In: *F1000Res.* 10.33 (Jan. 2021), p. 33.

[242] *nanopore*. en.

[243] Sarah Griffiths. *Quality Scores And Read Accuracy*. en. `https://labs.epi2me.io/quality-scores/`. Accessed: 2022-12-21. July 2021.

[244] Murray Patterson et al. "WhatsHap: Weighted Haplotype Assembly for Future-Generation Sequencing Reads". en. In: *J. Comput. Biol.* 22.6 (June 2015), pp. 498–509.

[245] Andrew D Yates et al. "Ensembl 2020". en. In: *Nucleic Acids Res.* 48.D1 (Jan. 2020), pp. D682–D688.

[246] M D Bacolod et al. "The gene expression profiles of medulloblastoma cell lines resistant to preactivated cyclophosphamide". en. In: *Curr. Cancer Drug Targets* 8.3 (May 2008), pp. 172–179.

[247] Wushuang Du et al. "Methylation of NRN1 is a novel synthetic lethal marker of PI3K-Akt-mTOR and ATR inhibitors in esophageal cancer". en. In: *Cancer Sci.* 112.7 (July 2021), pp. 2870–2883.

[248] Ewa Byzia et al. “Recurrent transcriptional loss of the PCDH17 tumor suppressor in laryngeal squamous cell carcinoma is partially mediated by aberrant promoter DNA methylation”. en. In: *Mol. Carcinog.* 57.7 (July 2018), pp. 878–885.

[249] Joel I Pritchard and James M Olson. “Methylation of PTCH1, the Patched-1 gene, in a panel of primary medulloblastomas”. en. In: *Cancer Genet. Cytogenet.* 180.1 (Jan. 2008), pp. 47–50.

[250] T H Bestor and V M Ingram. “Two DNA methyltransferases from murine erythroleukemia cells: purification, sequence specificity, and mode of interaction with DNA”. en. In: *Proc. Natl. Acad. Sci. U. S. A.* 80.18 (Sept. 1983), pp. 5559–5563.

[251] Oriol Fornes et al. “JASPAR 2020: update of the open-access database of transcription factor binding profiles”. en. In: *Nucleic Acids Res.* 48.D1 (Jan. 2020), pp. D87–D92.

[252] Vahid Akbari et al. “Megabase-scale methylation phasing using nanopore long reads and NanoMethPhase”. en. In: *Genome Biol.* 22.1 (Feb. 2021), p. 68.

[253] Dominik Stanojević et al. “Rockfish: A Transformer-based Model for Accurate 5-Methylcytosine Prediction from Nanopore Sequencing”. en. Nov. 2022.

[254] *Genome Browser bedGraph Track Format*. `http://genome.ucsc.edu/goldenPath/help/bedgraph.html`. Accessed: 2023-1-2.

Colophon

This document has been typeset in LaTeX, using the koma-script `scrbook` class. The following figures have been created with BioRender.com: Figure 1.1, Figure 1.4, Figure 1.6.