APPLICATIONS OF PUBLIC OMICS DATA

by

Sean K. Maden

A DISSERTATION

Presented to the Department of Biomedical Engineering
within the Oregon Health & Science University
School of Medicine

in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

May 2022

Biomedical Engineering
School of Medicine
Oregon Health & Science University

CERTIFICATE OF APPROVAL

This is to certify that the PhD dissertation of
Sean K. Maden
has been approved.

Abhinav Nellore, Ph.D.
Advisor

Jeremy Goecks, Ph.D.
Chair

Reid Thompson, M.D., Ph.D.
Committee Member

Kasper Hansen, Ph.D.
Committee Member

Kyle Ellrott, Ph.D.
Committee Member

Acknowledgements

First and foremost I thank my incredible mentor Abhi Nellore, who played a vital role throughout my graduate school career. Abhi is an inexhaustable source of wisdom and has been an ever-supportive friend throughout my time at OHSU. I will cherish our many heated conversations, tangents, and rants, which inspired and motivated the work featured in this dissertation and set me on the path to success.

Huge thanks are owed to the current and former members of my dissertation advisory committee, including Reid Thompson, my external committee member Kasper Hansen, and my committee chair Jeremy Goecks. I am also grateful to my oral examination committee member Kyle Ellrott, and to my former (i.e. newly retired) committee chair Joe Gray. These individuals contributed key critical insights and helped me to find the blindspots in my own work, for which I'm extremely grateful.

Other faculty and staff at OHSU were crucial to my success as a graduate student. In particular, I thank Monica Hinds, JoAnn Takabayashi, Holly Chung, Alex Brieding, and Sandra Rugonyi for their ready guidance and endless patience. I'm also grateful to my supportive and thoughtful BME Qualifying Exam Committee, including Clara Mosquera-Lopez, Kent Thornberg, Joe Aslan, and my qual chair Dan Zuckerman. For their advice, intellectual interest, and passion I also thank my current and former PDXgx lab mates, including Julianne David, Ben Weeder, Mary Wood, Chris Loo, Austin Nguyen, and Max Schreyer. Our internal shared lab server `spiracle` was crucial for many projects, and I thank it for all the good memories. Finally, the Student Health and Wellness Center was instrumental to my well-being as a graduate student, and my thanks and gratitude go especially to Carrie Milligan and Melissa Mannke.

I appreciate the support and enthusiasm of many other individuals outside of OHSU. In particular, I thank John Greally for connecting me to the wider epigenetics researcher community. I also thank Stephanie Hicks, Leonardo Collado-Torres, Chris Wilks, Tim Triche, Sean Davis, Andres Cardenas, Bret Barnes, Rishi Porecha, Roberto Avelar, Ludwig Geistlinger, Robert Gentleman, and Alan Hubbard. Special thanks and love go to my moral support group, including my girlfriend Lindsay Dawson, my parents Chris Maden and Kim Riddell, my brother Eric Maden and brother-in-law Morris Stegosaurus, and my two-wheeling friends at TnP Scooter Club PDX.

I can't overstate the gratitude I feel to have been able to thrive during a years-long pandemic, and the above individuals provided me substantial help and encouragement through difficult times. My thanks and apologies go to those I've forgotten to mention above.

Dedicated to the dreamer

TABLE OF CONTENTS

ABSTRACT

Public omics datasets from sequencing and array-based platforms now amount to millions of samples across thousands of studies and projects. This wealth of data is difficult to access and analyze due to heterogeneous and incomplete metadata, varying assay quality, and limited amounts of raw data provided for certain platforms. Despite this, accessing and reusing public omics data is an economical and prudent means of planning and conducting new experiments.

DNA methylation (DNAm) consisting of a cytosine-bound methyl group is an epigenetic mark that is widely probed across the human genome using Illumina's serialized Infinium BeadArray platforms. While tens of thousands of samples now have raw DNAm array image data available in the Gene Expression Omnibus (GEO), there have been few attempts to uniformly process sample metadata and raw signals for these samples. We programmatically obtained public DNAm array data from GEO, and we uniformly processed and compiled samples run on HM450K and EPIC, the two most popular platforms. We provided access to these compilations with support functions and vignettes for preprocessing and analyses in the `recountmethylation` Bioconductor package. We further developed the `recountmethylation_instance` Snakemake workflow to support the generation of future compilations.

Using the above data compilations, we conducted novel cross-study analyses of public DNAm array data. We studied DNAm variance patterns across seven normal tissues, and identified probes with either low variances across tissues or high tissue-specific variances. We further conducted cross-study and cross-platform analyses of prevalent blood sample types, which we used in power analyses as well as to replicate subsets of probes with differential methylation across sexes from two prior epigenome-wide association studies (EWAS). Our findings can inform future experiment design efforts, expectations for replication of differentially methylated probes, and novel cross-study analyses of public DNAm array data.

We further tested the reliability of several software tools for calling retained introns (RIs) from short-read RNA-seq data by using long-read RNA-seq data as our reference. We identified and obtained sample-matched public short-read and long-read RNA-seq data from the Sequence Read Archive (SRA), including data from a human induced stem cell sample and from a human whole blood sample. We found that RI-detection tool performances according to precision, recall, and F1-score, were related to their key assumptions, and that tools showed uniformly low performances for calling RIs which were also persistent in long-read data. Our results show the limitations of widely used RI-detection tools, and they can inform future efforts to improve the reliability of tools for calling alternatively spliced transcripts from short-read RNA-seq data.

List of Figures

List of Tables

List of Abbreviations

Symbols

5mC 5-methylcytosine. 9, 12, 13

A

ANOVA Analysis Of VAriance. 30, 32, 35, 36, 47, 143, 153, 163

API Application Programming Interface. 3, 17, 18, 108

C

CCLE Cancer Cell Line Encyclopedia. 2

CCS Consensus Circular Sequence. 58

CpG Cytosine-Guanine dinucleotide. ix, xi, 10, 13, 14, 17, 23, 24, 40, 41, 46–48, 55, 74, 75, 154, 161–163

cRNA Circular RNA. ix, 57, 64, 65, 178

CTS Co-Transcriptional mRNA Splicing. 172

D

DMP Differentially Methylated CpG probe. ix, xi, 20, 28, 48, 54, 55, 166, 167, 171

DNAm DNA methylation. ix, xi, 2, 5–7, 9–12, 14, 17–19, 23, 27–30, 32–35, 38–43, 46–50, 53, 55, 56, 74, 75, 149, 152, 153, 161–166

DNMT DNA Methyltransferase. 9, 10

E

ENCODE Encyclopedia Of DNA Elements. 2, 5, 29, 31

EPIC Illumina Infinium HumanMethylomeEPIC BeadArray platform. 9, 12, 28, 33, 34, 41–43, 45, 46, 49, 52, 55, 74, 108, 149, 169

EWAS Epigenome-Wide Association Study. viii, 2, 19, 28, 29, 40, 55, 74

F

FASTQ Text-encoded sequencing data. 3, 5, 15–17, 60

FDC False Discovery Cost. 25

FDR False Discovery Rate. 20, 25

FEV Fraction Explained Variance. ix, xi, 47, 53, 163, 170

FF Fresh Frozen. 36, 38

FFPE Formalin-Fixed, Paraffin-Embedded. 36–38, 158

FLNC Full-Length Non-Concatemer. 58, 66

FN False Negative. 25, 64, 68–71, 181–183

FP false positive. x, 24, 64, 68–71, 181–183

FPKM Fragments Per Kilobase Million. 62, 63, 67

G

GEO Gene Expression Omnibus. ix, xi, 2, 3, 6, 7, 17, 18, 27–31, 33, 34, 42, 43, 46, 49–51, 55, 74, 108, 112, 116, 149

GFF3 Generic Feature Format version 3. 67

GPL GEO platform accession number. 18, 30

GSE GEO study accession number. ix, 32, 35, 38, 40, 155, 156, 159, 163

GSM GEO sample accession number. 33, 155, 156, 159

GTEx Genotype-Tissue Expression Project. 2

GTF Gene Transfer Format. 59, 60, 62–64, 67

H

HDF5 Hierarchical Data Format 5. 22, 30, 42, 46, 50, 74

HM27K Illumina Infinium HumanMethylome27K BeadArray platform. 6, 7, 12, 33, 34, 42, 149

HM450K Illumina Infinium HumanMethylome450K BeadArray platform. 6, 7, 12, 14, 28–30, 33, 34, 41–43, 46, 47, 49, 55, 74, 149, 165, 169

HM850K Illumina Infinium HumanMethylome850K BeadArray platform. 12, 33, 165

HNSW Hierarchical Navigable Small Worlds. 47

HX1 Human whole blood. ix, xi, 58–65, 67–70, 72, 73, 173–183, 185, 189, 190

I

ID Intron Detection. 57

IDAT Dual-channel color intensity data file. 5, 13, 18, 29, 30, 46, 50, 108

iPSC Induced Pluripotent human Stem Cell. ix, xi, 58–65, 67–72, 173–183, 185, 191, 192

IR Intron Retention. ix, 57, 62, 64, 65, 67, 72, 73

J

JSON JavaScript Object Notation. 5, 31, 112, 116

K

KMA Keep Me Around. 57, 60–62, 65, 67–71, 74, 128, 176, 178, 180

L

LWM Length-Weighted Median. x, 63, 184

M

M Methylated signal. 5, 13, 14, 31, 74, 151

MAD Median/mean absolute difference. 35

mRNA Messenger RNA. 172

N

NCBI National Center for Biotechnology Information. 2, 3, 17, 58

NGS Next-Generation Sequencing. 2, 14, 15, 17

NMD Nonsense-Mediated Decay. 72, 172

noob Normalized, Out-Of-Band signal. ix, 13, 18, 29–32, 38, 40, 42, 43, 46, 47, 152

NOS Not Otherwise Specified. 51, 54

P

PacBio Pacific Biosciences. 2, 5, 16, 58, 61, 64, 73, 176

PBMC Peripheral Blood Mononuclear Cells. ix, 28, 46, 48, 49, 51, 53–55, 165–167, 171

PC Principal Component. 38, 40, 164

PCA Principal Component Analysis. ix, 20–22, 30, 32, 36, 38–40, 47, 53, 150, 152, 164, 165

poly(A) Polyadenylated. 57, 172

PSI Percent Spliced In. 63, 67

PTS Post-Transcriptional mRNA Splicing. 172

Q

QC Quality Control. 46

R

RI Retained Intron. ix–xi, 17, 25, 26, 57, 58, 61–75, 126, 128, 176–181, 183, 187–192

RNA-seq RNA sequencing. 1, 3, 6, 8, 22, 28, 57, 64, 66, 72, 74, 75, 126, 128

S

SMRT Single Molecule Real-Time. 16

SOFT Simple Omnibus Format in Text. 31, 46, 50, 112

SRA Sequence Read Archive. ix, 2, 3, 5–7, 15, 55, 56, 58, 60, 64, 66, 74, 75

SRR SRA run accession number. 60

SVA Surrogate Variables Analysis. 48

T

TCGA The Cancer Genome Atlas. 2, 29

TN True Negative. 24, 64, 71

TP True Positive. 24, 64, 68–72, 181–183

TPM Transcripts Per Million. 62, 67

U

U Unmethylated signal. 5, 13, 14, 31, 74, 151

W

WGBS Whole-Genome Bisulfite Sequencing. 12, 75

1 Introduction

1.1 Public omics data types and sources, and issues for reanalysis

Omics data refers to a variety of data types enabling study of entire classes of molecules like DNA, RNA, and proteins. Analyses of omics data yield high-resolution snapshots of life inside a cell or tissue [1]. This drives the discovery and characterization of gene networks whose coordinated expression gives rise to specialized phenotypes, cell signaling pathways, tissue development, and much more. Comparative studies of omics data from normal and diseased tissues can reveal the dynamic molecular mechanisms underlying health and illness. These studies can inform development of novel clinical panels of molecular signatures called biomarkers, with the ultimate goal of improving screening protocol efficacy, diagnosing diseases earlier and more accurately, and personalizing interventions and treatments.

Public omics datasets have grown immensely in the decades since the first publication of a sequenced human genome [2], and newer omics technologies continue to improve the resolution and reliability of data collected [3–5]. With growth in available omics data, there is increasing need to make existing public omics data more readily accessible. These datasets can be useful for planning new experiments, testing new hypotheses, and performing independent validation and multi-omics analyses, especially when the raw or non-normalized assay data are paired with complete and descriptive metadata. Unfortunately, inconsistent reporting standards and completeness in public omics metadata remain common, making it difficult for other investigators to reuse samples in new analyses. Further, public omics data compilations can be very large in size, making scalable memory-efficient methods crucial for comprehensive and cross-study analyses.

This section provides an overview of the omics data types studied in this dissertation, how omics data are made available and accessible from public repositories, obstacles to more extensive use of public omics data in research, and modern issues surrounding reproducible research for computational disciplines.

1.1.1 Transcriptomics and the exome

Transcriptomics is the study of the transcriptome, or the set of all RNA molecules transcribed from DNA and processed by splicing. RNA molecules can include sequence regions spliced together during transcript processing, called exons, and these may cue the polymerization of amino acids into proteins. The entire set of such regions is called the “exome,” and whole-exome studies focus specifically on these regions [6]. Related to the concept of the exome is the set of all possible products resulting from RNA splicing, or what is occasionally referred to as the “spliceome” [6]. As with other omics data types, transcriptomics data can be generated from microarray-based or sequencing-based technologies, where the latter is usually called RNA-seq.

During sample processing, most sequencing platforms produce reads, and understanding read characteristics can be central to differentiating sequencing platforms. Reads commonly consist of hundreds to tens of thousands nucleotides, and they are produced by the amplification of library-specified DNA sequences where these are present in the sample being processed. Reads are sequenced, mapped to a reference genome sequence, then analyzed to determine the genes and transcripts expressed the processed sample. Next-generation sequencing (NGS) technologies include many high-throughput platforms which produce upwards of millions of reads per sample and can target regions throughout the human genome. Many platforms are used to produce RNA-seq data, of which we draw attention to Illumina's NextSeq and HiSeq platforms for short-read sequencing and PacBio's long-read sequencing platform (Section 1.3.3). In Section 4, we use public data from these platforms to study the reliability of tools for calling alternatively spliced transcripts containing retained introns [5] (Section 1.3). Short-read platforms have historically been more widely used than long-read platforms, but long-read platforms are gaining in popularity as costs to run samples decline and methods for preprocessing and analyzing short-read data have matured.

1.1.2 Epigenomics and the methylome

Epigenomics is the study of the epigenome, which refers to a large set of physical or chemical phenomena that can impact gene expression. For example, many types of non-protein encoding RNAs are considered part of the epigenome. The addition of methyl groups to a DNA nucleotide, otherwise called DNA methylation (DNAm), is the epigenetic mark of focus for this dissertation. The collective set of all DNAm in a cell is called the methylome. Just as the exome refers to a subclass of molecules in the transcriptome, the methylome is a subclass of molecules in the epigenome. The methylome is commonly studied in Epigenome-Wide Association Studies (EWAS) using several types of high-throughput microarray platforms. Platforms for studying DNAm vary widely in their scope, resolution, reliability, and costs (Section 1.2.4). In depth discussion of the biological and clinical relevance of DNAm and its quantification in omics data are provided in Section 1.2. This informs the cross-study analyses appearing in following chapters (Sections 2 and 3).

1.1.3 Omics data sources and access

Public omics data is typically accessed from a data source, called a public data repository, using a browser or FTP connection. Many resources varying in size and scope provide access to public omics data in some form [7–9]. Some of the most comprehensive resources contain data produced by large consortia, such as the Encyclopedia Of DNA Elements (ENCODE) [10, 11], the Cancer Cell Line Encyclopedia (CCLE) [12], the Genotype-Tissue Expression Project (GTEx) [13, 14], and the The Cancer Genome Atlas (TCGA) [15]. Data from these consortia represent the efforts of many labs over many years, or sometimes decades, and it is common to see these datasets cited in hundreds or thousands of papers [14]. Alongside consortia are smaller datasets hosted in the GEO and SRA repositories maintained by NCBI [8, 9, 16]. These repositories commonly host data published from one or several labs, to accompany one or several studies. The stated purpose of repositories like GEO and SRA is to archive data for published work on behalf of authors [8, 9, 16]. Since there is limited curation, spot checking, and error correction performed on data and metadata hosted by these sources, these important tasks are largely left to researchers [3, 4, 17].

In many ways, GEO's name belies its effective role as a metadata purveyor and a source for supplemental data, including epigenomics data. Many experiments are accompanied by a GEO study record ID, which corresponds to a unique record in the database which includes important study and sample metadata. Each record may also include a section for supplemental data that may include the platform annotations, processed forms of the analyzed data, and even raw data from array platforms. While raw or unprocessed array data can be found in the supplemental sections of GEO records, corresponding raw data from sequencing platforms is provided through the SRA. This includes all manner of sequencing data, from RNA-seq, to genome sequencing, to bisulfite sequencing and less popular data types from human, and non-human organisms. Despite this, many sequencing experiments will host study and sample metadata in GEO and other databases. This means researchers can often query GEO to learn details about sequencing experiments whose data is hosted in SRA.

GEO, SRA, and other NCBI repositories include helpful search features accessible from the browser. However, browser interaction is not optimal for automation and reproducibility. Thus, most computational researchers will find the Entrez Programming Utilities [18] software extremely valuable for programmatically accessing NCBI database Application Programming Interfaces (APIs). APIs provide an interface between the repository and the data miner, where key properties of the data are exposed for query. APIs can also dictate the query logic which a programmer can apply to reliably obtain a result based on certain information provided in the query. The Entrez software provides a standard way of identifying studies and samples by properties of the data type, platform, metadata, and so on. We used this software to automatically find sample and study record IDs for DNAm arrays and RNA-seq datasets across multiple platforms (Sections 2.2.1 and 3.2.1). We conducted Entrez queries using a Python script to automatically find record IDs and construct the download URLs to obtain supplemental files (Appendix A.1).

While Entrez Programming Utilities enables programmatic access to records and data from GEO, the SRA-Toolkit [16, 19] software enables programmatic access to sequencing data from SRA. For short-read samples, raw data takes the form of a pair of FASTQ-formatted files containing the reads generated in either the forward or reverse DNA strand. We downloaded FASTQs using the toolkit's `fastq-dump` or `fasterq-dump` functions. By contrast, the toolkit functions don't presently support download of raw long-read data, called movie files, and these files needed to be batch downloaded using custom scripts (Section 4.2.2).

1.1.4 Obstacles for analysis of public omics data

Several key obstacles impede analysis of public omics data. First, available information from published experiments can vary widely. A scan of public archives shows varying completeness of experiment records, ranging from only study metadata, to limited sample metadata paired with normalized assay data, to complete sample metadata provided with raw or non-processed assay data. The latter is the most reproducible way of reporting an experiment because the raw data can be used to repeat the entire preprocessing and analysis workflow from the beginning. Complete and thorough metadata is crucial for relating samples across projects and fully understanding the origins of assay data. In addition to data and code, studies should describe their pipelines in sufficient detail so that they can be fully reproduced. Better yet, a runnable workflow or pipeline script can be provided. Studies of omics data frequently use multistage pipelines that need to be thoroughly documented to ensure their reproducibility (Section 1.1.10), many studies unfortunately don't report their methods in sufficient detail to be reproduced [20].

Next, public omics datasets can be difficult to analyze and interpret. Manipulation of omics data often requires specialized training and skills, and omics data studies demand careful planning. Further, even well-designed omics studies can be difficult to interpret because of statistical assumptions and because observational data is usually insufficient to show a causal relationship between some molecular marker or signature and a condition of interest (Section 1.4.4). Many omics studies focus on discovering biological indicators of disease or pre-disease conditions, a pursuit called "biomarker discovery." Biomarker discovery describes a significant portion of biomedical research, yet even exceptionally robust and informative biomarkers need only be consistently indicative or correlated with a condition and need not have any causal relationship to the condition of interest [21–24]. Thus there is constant need to elucidate biological mechanisms and processes that explain health and disease at a deeper levels beyond robust correlations.

Finally, metadata heterogeneity and varying completeness can confound cross-study omics data analyses even where individual studies have provided raw data and sufficient sample metadata. Metadata heterogeneity arises due to lack of standards for metadata reporting, which might dictate how information is conveyed and organized, and which could entail the mapping of controlled vocabularies [17, 25] (Section 1.1.6). Varying metadata completeness contributes to metadata heterogeneity, and can arise in three key ways. First, metadata may be omitted to protect subject identity. For example, this could be a concern for studies of exceedingly rare disease, where there is reasonable concern a patient could be identified from provided metadata and/or provided omics data. Second, incomplete metadata can arise due to negligence or ignorance on the part of the researcher or data recorder. Lack of widespread metadata reporting standards exacerbates this particular problem, as this means more work on the part of researchers to familiarize themselves with previously archived metadata to determine how best to represent metadata for their own study. Third, metadata may appear to be lacking if metadata mapping or learning pipelines cannot access it. This can happen if metadata is erroneously recorded in an unexpected location or using a non-standard format. Finally, non-standardized and highly specialized recorded metadata contributes to metadata heterogeneity. For example, sample identifier codes are often meaningless outside of a particular lab or consortium, and manifests describing the meaning of sample codes are not always readily available. Ultimately, metadata heterogeneity confounds omics data discovery and reuse, and moving forward it will become more crucial to introduce standard metadata reporting practices as well as develop tools to help researchers properly represent published data in public archives [3, 4, 26].

1.1.5 The role of raw data

The above obstacles for analysis of public omics data can be overcome through several means. First, data miners can limit the scope of their data searches to include only records with raw or non-normalized data available. The rawest possible form of data for an assay is typically the most desirable for follow-up analyses. This is because follow-up analyses may involve data from several studies, and disparate normalization strategies can confound analysis of samples across studies and pipelines. Further, the bias introduced by disparate normalization procedures can impede efforts to perform bias correction across studies, and this is a major obstacle for most meta-analyses (Section 1.1.8).

For DNAm array experiments, raw data consists of two paired color intensity (a.k.a. "IDAT") files, one each for the red and green color channels on the array reader. These paired files are read by software such as Genome Studio or `minfi` [27] R package, then parsed into methylated (M) and unmethylated (U) signals for further processing. These are discussed in greater detail in Section 1.2.5. Raw data from sequencing experiments consists of FASTQ files for short-read data, and movie files for PacBio long-read data. These are discussed in greater detail below (Section 1.3).

Provision of raw data also encourages reproducibility because it allows one to repeat the published data processing and analyses exactly as described by the authors. However, raw data can consist of very large files requiring large and expensive servers to host. As with whole-genome bisulfite sequencing data, the raw form of some data further may not be its most informative or useful representation (Section 1.2.4). Further, some metadata must be excluded from archived experiments to protect subject identity, and grants or other official sources can impose limits on information that can be made publicly available. For these and other reasons, many studies neglect to provide access to raw forms of omics datasets [3] (Section 2.3.1).

1.1.6 Harmonization of heterogeneous metadata

Next, metadata heterogeneity may be addressed using metadata harmonization [17, 25]. This technique aims to combine samples under a single set of standardized descriptive terms using standard and well-defined vocabularies describing biological, medical, demographic, and technical aspects of the sample [3, 4, 28]. Harmonized metadata can be obtained by either systematically mapping controlled vocabularies to encountered metadata, by learning controlled vocabularies or predictions using natural language processing, or by prediction or imputation from omics assays themselves. For metadata mapping, many entire curated term hierarchy dictionaries, or "ontologies," are now freely available through projects such as ENCODE [29]. These were previously used to map SRA metadata and make them queryable as the `MetaSRA` resource [17]. Despite their availability, there is no standard way of using ontologies for research, and even the `MetaSRA` makes use of just a handful of the dozens of available term ontologies [17]. This means it is largely upon individual researchers to determine how best to harmonize metadata mined from public repositories.

Fortunately, metadata mapping is not an intractable problem. For `recountmethylation` data compilations, we incorporated metadata mapping using R scripts applying regular expressions, an extremely common and useful syntax for matching complex character patterns. First, sample metadata in JSON file format, which is stored using a predictable "key:value" format, is coerced into a flat table (Appendix A.3). Then these data are postprocessed by mapping a more informative and controlled vocabulary under informative variables for characteristics like sample type, disease condition, and so on (details in Section 2.2.7), which was an approach inspired by a prior compilation of DNAm array data [28] (Appendix A.4). While we did not map terms from ENCODE ontology dictionaries, we used descriptive terms capturing the most common information in the archived metadata, as determined by a combination of observation, manual review, spot checks, analysis, and summarization. We also used the `MetaSRA-pipeline`

(Section 2.2.6), which was used to compile the MetaSRA and which we found was also generalizable to metadata encountered in GEO. We further expanded the harmonized data using variables predicted from DNAm with canonical models, including the attributes sex [27], age [30], blood cell fractions [31], and genetic ancestry [32] (Section 3.2.2).

1.1.7 Previous public omics data compilations of note

Several notable efforts have been made to compile public omics data, where we take data compilations to be resources combining metadata and assay data in some way, such as in a relational database, where compiled data has usually been subjected to some kind of uniform processing. This definition excludes utilities which may mirror public data or perhaps provide a new dashboard or browser interface to query and summarize datasets [33–37]. While these resources can be useful, they don't strictly provide a means of jointly preprocessing and analyzing sample metadata and assay data without considerable additional effort.

Many projects have attempted to comprehensively compile and process public RNA-seq data. Importantly for this dissertation is the recount project, from which the recountmethylation [38] Bioconductor [39] package gets its name. The recount project now spans three key versions [26, 40, 41] and a separate project for data from brain samples [42]. For recount, researchers compiled sequencing data mined from the SRA. In its newest manifestation as recount3, the project includes human and mouse samples from a wide array of bulk and single cell sample types [26]. Importantly, the recount data compilations are supported by helpful browser interfaces and complementary Bioconductor packages [39, 43, 44]. These resources ensure it is easy to summarize experiment and sample metadata, query the compiled data, and apply the same processing methods to new data. Further, data from recount was used to make a queryable index of splice junctions called Snaptron. Thus recount is a prime example of how public omics data can be made more accessible and used to develop new research tools.

For DNAm array data, the Marmal-aid [28] resource compiled uniformly processed HM450K [45, 46] and HM27K samples from GEO. It was initially released in 2013, and updated for several years after. Importantly, it included harmonized and uniformly formatted metadata under several variables such as tissue, disease, sex, and age. These variables informed the metadata variables we used in recountmethylation (Sections 2.2.5 and 3.2.2). While an impressive effort, the maraml-aid resource is now out of date, as relatively few HM27K samples have ultimately been published to GEO, and the newer EPIC [47] platform is now overtaking the HM450K platform in popularity and prevalence. Harmonized metadata can be further enhanced with reliable DNAm-based attribute predictions using canonical models. We were able to predict age, sex, blood cell fractions, and genetic ancestry for samples in the recountmethylation data compilations. Detailed discussion of the Illumina Infinium BeadArray platforms for DNAm array data are provided in Sections 1.2.4 and 1.2.5 (Section 2.3.3, Appendix 6.1).

1.1.8 Cross-study analyses versus meta-analyses

Cross-study analysis is an important experiment design strategy used throughout omics literature. In contrast to meta-analyses, cross-study analyses are best characterized as analyses of studies using uniformly normalized data rather than data normalized independently or from distinct normalization pipelines. For example, a cross-study analysis may compare results from many different studies, may uniformly process and normalize data from many different studies, or use some combination of the two approaches.

In theory, cross-study analysis allows one to leverage greater quantities and varieties of samples more effectively than would be possible in a meta-analysis. In practice, there is need to more fully develop and benchmark methods for effective cross-study analyses. We tackled one piece of this larger issue by evaluating different strategies for performing study ID-specific bias corrections across DNAm array data from thousands of blood samples and dozens of studies (Section 3.3.2). These efforts make it clear that study ID explains a lot of variances across samples while also serving as a surrogate for technical bias, and they underscore the importance of having uniformly normalized data, as disparate normalization strategies would inflate study ID-specific variances across samples and reduce explained variances after bias correction efforts.

Prior work includes numerous studies which effectively employ cross-study analysis experiment designs. For example, [30] introduced one of the first highly accurate pan-tissue epigenetic clocks. This paper conducted independent validation across studies as well as normalization across both the HM27K and HM450K array platforms (details in Section 1.2.5). Further, [48] leveraged many different tissues to evaluate the behavior of differentially methylated probes from sperm across a variety of other tissues. In addition, [49] used many birth cohorts to validate an epigenetic clock that accurately predicts gestational age. Importantly, public repositories feature samples from exceptionally rare conditions and diseases, and these have been used to develop novel DNAm-based panels for diagnosing multiple rare Mendelian disorders [50, 51]. Many prior studies use cross-study analyses of transcriptomics data, including a characterization of brain regions and cell types [42], and a systematic comparison of bulk and single-cell transcriptomics in mouse and human [26].

1.1.9 Present and future applications of public omics data

There are many useful applications of public omics data. A researcher planning her next experiment might use empirical effect size estimates from public data to estimate the number of samples needed to test a new hypothesis (Section 1.4.8). An analyst could periodically revisit an analysis pipeline using GEO data to repeatedly test a hypotheses over time as more data becomes available, and a clinician might query SRA for samples from an extremely rare condition in order to test the efficacy of a biomarker panel.

The increasing volume and variety of public omics data is driving novel applications such as appropriation of old methods for new platforms, development of new methods, and novel uses for old samples. Multiple omics data types for a single tissue can be analyzed together in a multi-omics experiment, and such analyses of orthogonal assays can further an understanding of the biological systems and networks underlying normal development and disease [52, 53]. For example, transcriptomics and epigenomics data from the same tissue could be integrated and jointly studied, which can possibly uncover epigenetic marks mediating gene expression, gene co-expression networks that change global epigenetic profiles, $trans$-acting epigenetic marks and transcription factors, and more.

Recent work applies the concept of transfer learning to complementary data types to improve analysis of either data type individually. For example, single-cell sequencing data can be used to impute cell types in less high-resolution spatial transcriptomics data [54]. Further, hybrid methods make joint use of short-read and long-read data for preprocessing to reduce the technical errors in long reads [55]. Innovative public omics data analyses can also be driven by older methods. For example, transcriptome and differential expression analysis of newer long-read data can make use of tools developed previously for short-read data. Further, cell type deconvolution methods developed for bulk RNA-seq data can also be appropriated for deconvolution problems involving single-cell and spatial transcriptomics data. Many recently developed methods make novel use of machine learning techniques, especially convolutional neural networks [54, 56]. Thus transfer learning, appropriation of old methods for new data types, and novel machine learning approaches are driving new applications of public omics data.

1.1.10 The central role of reproducibility in research

There are numerous opportunities and challenges presented by public omics datasets, and effective omics research demands critical thinking about not only biological systems but also the ways in which those systems can be studied. Reproducibility is the ability to recreate an analysis using the exact data, methods, and code provided in a study. This idea is central to empirical inquiry since rigorous science should be reproducible by definition. However, it has been specially emphasized for computational fields over concerns that far too few studies are reproducible in practice [57–59]. These concerns motivate frequent reference to "reproducible research" as a way of proactively calling attention to methods and practices that ultimately increase reproducibility.

As a scientific approach, reproducible research emphasizes key ideas like methodological rigor, transparent methods, detailed documentation, and other domain-specific practices. Adherence to reproducible research benefits individual researchers, the greater community, and even the wider lay community. First, it benefits those who practice it because it encourages better practices in reporting and documentation [58]. Second, reproducibility enables replicability, which refers to the ability to effectively apply the exact methods as reported in order to analyze new data. This clearly follows because reproducible methods are more transparent and detailed, which improves the researcher's ability to repeat those methods in a new setting. Third, reproducible research benefits the greater scientific and lay community overall because they clarify the specific accomplishments and limitations of scientific studies. This paves the way for more constructive and honest interpretation of results and it discourages erroneous or over-optimistic interpretations.

Domain-specific reproducibility measures must be considered for computationally-intensive disciplines. These measures encompass the inclusion of all dependencies with their versions in reported methods, making code legible with comments and docstrings, and making code open-access to promote transparency [60]. Reproducibility is further encouraged through complete provision of code with datasets, as this enables the study's entire pipeline to be repeated. Provision of code with dependencies is encouraged by virtual environments and containers (Section 1.4.9). Reproducibility standards may also be applied to data compilation efforts. For example, initial `recountmethylation` data compilations provided uniformly processed metadata and DNAm array data in several formats. These large compilations were used by several groups to study a variety of topics (private correspondence). However, this approach demanded manual curation, and the large datasets required significant server resources to host. After several updates and the introduction of EPIC platform samples, we are shifting towards providing access only to `recountmethylation_instance` [61], a Snakemake workflow for compilations of public DNAm array data from GEO. Hosting a workflow requires trivial cloud storage space, and users can fork, modify, and update the workflow to suit their needs. We attempted to limit the need for manual curation by automating parts of this workflow. More details about workflows and other tools to promote computational reproducibility for omics data analyses are discussed in Section 1.4.9.

1.2 Probing the methylome with DNA methylation microarrays

DNAm is the best-studied epigenetic mark and is now known to play crucial roles in development, gene expression regulation, genome stability, and more. DNAm is most commonly studied using microarray-based technologies. DNAm marks specifically refer to a class of chemical modifications applied to individual DNA nucleotides, resulting in the addition of a methyl (CH_3) group. The best studied of these modifications consists of the addition of the methyl group at the 5' position of cytosine, which yields 5-methylcytosine (5mC). 5mC is especially common at cytosine-guanine sequence regions, otherwise known as CG dinucleotides or CpG loci. Unless otherwise noted, usages of DNAm below refer strictly to 5mC occurring at a CpG locus. CpG loci are commonly assayed in bulk tissue samples using high-throughput microarrays, or simply "arrays," among which Illumina's Infinium BeadArray-based platforms have been the most popular and widely used. This section describes DNAm in greater detail, including relevant biological context and processes, its biological and clinical importance, and how DNAm arrays detect and quantify 5mC.

1.2.1 Molecular mechanisms of DNAm gain and loss

Like most epigenetic marks, DNAm varies across cells and tissues within an organism in a phenomenon called "epigenetic mosaicism." Characterizing and predicting the conditions, patterns, and limits of epigenetic mosaicism are key tasks of modern epigenetics research. For decades it has been appreciated that genome DNAm patterns reflect competing and interacting genetic and environmental influences, but the specific nature of those influences is just starting to be elucidated.

DNAm is linked to cell functions and activities, and especially to the cellular processes of replication and senescence. Conversion of cytosine into 5mC is mediated by the family of enzymes called DNA methyltransferases (DNMTs), especially subclasses known as DNMT1, DNMT3a, and DNMT3b. During and following cell replication, DNMT1 acts in concert with other proteins to facilitate addition of methylation back to cytosine on the newly polymerized DNA strand [62, 63]. This process is called "maintenance methylation," as DNMT1 attempts to exactly copy DNAm patterns in the original template DNA. DNMT1 has lower fidelity compared to similar replication enzymes like DNA polymerase, and this low fidelity explains replication-associated DNAm loss [64].

Alongside DNMT1 maintenance methylation, several types of DNMT3 enzymes facilitate de novo methylation, which is the addition of a new methyl group to cytosine at a previously unmethylated CpG locus [65]. DNMT3 activity is complex and driven by interactions with other factors including ten-eleven translocation (TET) enzymes [66]. Further, the subclass of DNMT3b enzymes may specifically increase DNAm in the gene body [67]. When DNAm is lost either through active demethylation or passively through imperfect DNMT1 maintenance, DNMT3s may promote maintenance DNAm by facilitating 5mC reformation [63].

1.2.2 The biological importance of DNAm

Our understanding of the biological importance of DNAm has increased drastically in recent years, and it is now widely accepted that DNAm is an important mediator of gene expression, transcript selection, and alternative splicing [63, 68]. Among DNAm's roles in regulating gene expression, it is well known to down-regulate expression in cis when it is present in the gene's promoter [69–71]. Among several key mechanisms explaining this phenomenon, one of the most well-described involves DNAm's prevention of binding by transcription factors which are required to promote transcription [63, 72].

DNAm in the gene body may up-regulate gene expression, including by discouraging spurious transcription initiation in a DNMT3b-dependent manner [67], or possibly because body DNAm stabilizes RNA polymerase during transcription. DNAm may promote or repress gene expression in $trans$ through its presence or absence at intergenic regulatory regions known as enhancers and insulators [69–71]. Through this and other means, DNAm acts in tandem with DNA-protein complexes called histones, 3d genome topology, non-coding RNAs, and other elements and conditions to regulate gene expression. Unfortunately, much of our understanding of DNAm is confounded by issues with unambiguously assigning CpGs to either a gene's body or promoter, especially where the CpG overlaps multiple transcripts asymmetrically. These concerns are discussed further in Section 1.4.1.

Besides gene regions, DNAm is associated with other cytosine- and guanine-rich regions located throughout the genome called CpG islands [73]. Further, cytosine- and guanine-depleted regions that surround CpG islands are called shores, shelves, and OpenSeas, respectively [74]. While up to 80% of CpGs are thought to be methylated genome-wide, CpG islands typically show low DNAm despite having high concentrations of CpG loci [63]. Most gene promoters either overlap or occur in cis to a CpG island [75]. CpG island-specific DNAm dynamics, such as age-associated drift, can mediate expression at these CpG island-proximal genes [76, 77].

DNAm plays a key role in X chromosome inactivation in human females, or the process by which sex chromosome gene dosage is controlled by systematic high DNAm and repressed gene expression at one X chromosome but not the other [78, 79]. Relatedly, DNAm plays an important role suppressing activity in repetitive and retroviral regions, preventing transcription interference and disease-promoting processes [80, 81]. DNAm is further tied to tissue and embryological development, cell fate, and cell identity [82]. Several waves of demethylation, or widespread loss of DNAm, occur prior to egg fertilization and during embryonic development in mammals, with focal conservation of parentally inherited 5mC at imprinted genes [63, 83]. Further, DNAm changes are triggered by cell activity and development, such as in T-cell activation and stem cell induction [84]. DNAm variation and loss is associated with cell mitotic activity and biological aging [85, 86] in a manner conserved across many species [87, 88] (Section 1.2.3).

1.2.3 Clinical applications of DNAm

Many studies have discovered potential DNAm biomarkers of conditions ranging from obesity and diabetes [24, 89–91] to cancer [92–99]. DNAm from blood samples yields information about immune cell content and activity, cell free circulating DNA, and other potential markers of certain diseases and conditions, and blood DNAm tests are being investigated in the development of minimally invasive panels for clinical use [100, 101]. It takes considerable effort and resources to develop a DNAm biomarker-based panel ready for clinical use, and statistically most promising biomarker candidates don't ultimately pass rigorous validation [21–24]. As a result, DNAm is used in only a few clinical tests to date [102, 103].

Because DNAm is closely tied to genetic variation, it has been used as a surrogate for individual genetic variants and small nucleotide polymorphisms [32, 104]. This motivated investigation of DNAm as a biomarker of rare genetic diseases and development of EpiSign, a molecular panel for the simultaneous and accurate diagnosis of numerous rare Mendelian genetic disorders from DNAm array data [50, 51]. DNAm tests also show promise for improving screening of patients with gastroesophageal maladies. For example, Barrett's esophagus is a common disease characterized by metaplastic tissue at the gastroesophageal junction. Barrett's is one of the best risk factors for esophageal adenocarcinoma, a rare but deadly cancer of the lower esophagus [105, 106]. Despite available screening procedures and tests, it remains difficult to stratify Barrett's patients according to their risk of progression to cancer. This motivated extensive research into DNAm as a possible molecular marker of cancer progression risk in Barrett's patients [92, 107–112]. A cheap and effective DNAm clinical panel that improves risk stratification of Barrett's patients could reduce population burden of esophageal adenocarcinoma and promote more effective interventions and treatments for patients at the highest levels of progression risk.

Recent DNAm research has had a profound impact in the fields of aging and rejuvenation therapies. Alongside an organism's temporal age, it is thought that the phenotype may reflect an independent and parallel age with relevance to health and longevity, otherwise known as a "biological clock". DNAm was found to correlate with temporal age in surprising ways, and inferring biological age from DNAm is now one of the key ways in which aging is studied in the context of biomedical research [30, 113, 114]. This work could further yield more precise gestational age estimates from neonate cord blood DNAm, which could ultimately improve the efficacy of prenatal screening [49, 115–118]. DNAm-based clocks have been applied in a wide variety of settings and organisms, including to study the immune system, predict mortality, characterize biological responses to low-gravity environments, and more [30, 49, 113, 119–121].

1.2.4 Profiling DNAm by sequencing

DNAm can be quantified using either sequencing or array-based technologies. Most technologies initially subject DNA to a chemical transformation called bisulfite conversion. Bisulfite conversion converts non-methylated cytosines to uracil while preserving 5mC bases [122, 123]. The converted DNA can then be amplified and sequenced to determine whether a CpG contained 5mC.

Whole-genome bisulfite sequencing (WGBS) yields high-resolution, high-density datasets [124–129]. It is high-resolution because it shows the methylation status of an individual CpG locus on an individual DNA strand, rather than an aggregate of DNAm across DNA in an entire sample. Bisulfite sequencing is high-density because it yields very large data files. Their size can make these files difficult to work with even in a modern remote server environment with excellent memory and disk space resources. Further, the information yield is higher than necessary for most purposes. For example, certain genome regions tend to have high or low DNAm with great uniformity. In such regions, it is only necessary to retain the methylation status of a few informative CpG loci rather than all regional loci. These drawbacks have motivated development of reduced-representation techniques for sequencing data, as well as motivated interest in far more compact Infinium BeadArray platforms for studying the human methylome.

Alternative sequencing strategies can yield phased single-nucleotide DNAm information without having to use bisulfite conversion. For example, long-read sequencing technologies (Section 1.3.3) can detect a variety of nucleotide modifications without special DNA treatment. However, prohibitively high coverage is required to detect certain modifications such as 5mC, preventing its reliable detection in the absence of additional reagents or procedural measures [55]. Alongside high-throughput sequencing methods, pyrosequencing [130], methylite [131, 132], and other assays quantify DNAm at targeted loci with high resolution and granularity [92, 124, 133–135]. Each of these yields phased methylation information at specific loci with high reliability. These assays are widely used to test promising DNAm biomarkers, especially if those biomarkers were discovered by more high-throughput approaches. Experimental validation is important to test the rigor of promising biomarker candidates prior to investigating their clinical applications in greater depth.

1.2.5 Infinium BeadArray technology

Among the many platforms profile DNAm, the most popular to date are Illumina's Infinium BeadArray platforms. These include three generations of serialized BeadArray platforms released over the course of several decades. With each new platform release, the amount of genome coverage has increased and further been expanded to new regions. The earliest platform was called the HumanMethylome 27K, or "HM27K" array. This was followed by the HM450K, and then the newest platform called EPIC or occasionally HM850K. The platform names roughly correspond to the number of CpG loci they assay after quality control and preprocessing. Thus HM27K assays about 27,000 loci, HM450K assays about 480,000 loci [45, 136], and HM850K assays about 850,000 loci [47]. Each platform assays regions throughout the human genome with concentration at genes and CpG islands. The EPIC platform notably introduced new probes covering non-coding RNA and intergenic regulatory regions that had been absent from the preceding HM27K and HM450K platforms [74].

Infinium BeadArray platforms assay CpG loci using one of two probe types, called Type 1 and Type 2. Both types target 50 base pairs of DNA sequence adjacent to a target CpG. During array processing, fragmented bisulfite converted DNA is hybridized to the array probe sequence and a single-base extension occurs to bind a fluorophore-bound nucleotide which signals either red or green in the color detection channels. Type 1 probes utilize two beads, one each for the Methylated (M) and Unmethylated (U) signals [27, 72]. In newer platform generations, Type 1 probes have been replaced by the more compact Type 2 probes which use just a single bead and sequence that can yield either U or M signal. This is accomplished with a 50bp sequence incorporating a special kind of synthetic and degenerate nucleotide which binds equally well to either internal cytosines and 5mCs, allowing better DNAm quantification at CpG-rich regions such as islands [74].

Certain BeadArray probes have been shown to have off-target effects. In other words, these probes bind one or more genome regions besides the intended target, and off-target regions can even be on different chromosomes [47, 137]. Off-target effects can result if the 50bp probe sequence is insufficient to uniquely specify a genome region. This issue is exacerbated for Type 2 probes in regions with greater tandem repeated sequences, including CpG islands. Because of this, it is common practice to filter likely cross-reactive probes prior to analyses [49, 76, 92, 138].

Illumina's Infinium BeadArray platforms quantify DNAm for bulk samples, or cell mixtures which may include many distinct cell types [45, 47]. This yields an aggregate signal detected across DNA strands from cells throughout the sample. Color signals are read by a dual-channel system that produces a pair of complementary raw intensity data (IDAT) files corresponding to the red and green color channels, respectively. The paired IDATs are read and parsed into either methylated (M) or unmethylated (U) signals, which are in turn used to compute a DNAm fraction or Beta-value. Intuitively, the Beta-value reflects the fraction of signal from DNA strands containing 5mC at the targeted CpG probe, divided by the total signal from all DNA at that site. While Infinium BeadArray platforms yield an aggregate signal from a cell mixture, methods have been developed to deconvolve the particular cell content in a sample. Notable examples can determine fractions of Natural Killer cells, B-cells, T-cell sub-populations, and so on from blood samples [31, 139].

1.2.6 Preprocessing DNAm arrays

There is a considerable body of literature establishing normalization procedures for BeadArray probes. One of the most widely used normalization techniques is called "noob-normalization," whose name combines Normal-Exp, its convolution method, and out-of-band signal, the type of Infinium probe signal it utilizes [140]. This technique determines CpG probe assay quality and attempts to correct for systematic biases in detected color intensity levels prior to downstream analyses. It was shown to reduce biases, increase the dynamic range, and improve recovery of biological variation for both Type 1 and Type 2 probes [140]. Noob normalization is generally recommended as an initial preprocessing measure for DNAm array analyses, and it can be combined with other normalization types in a single preprocessing workflow [141–146]. Another important feature of noob normalization is that it is performed within rather than across samples. This makes it an ideal normalization method to be performed for distributed data compilations, since it isn't known what subsets of studies and samples the end user will analyze.

Several quality filters are routinely applied in DNAm array preprocessing. First, CpG probe signal can be directly used in quality filters [3, 27]. As mentioned above, thousands of CpG probes may undergo cross-reactivity, and these may be removed prior to analysis [47, 137]. Further, each probe has an associated detection p-value which reflects signal confidence and is ultimately a function of the amount of M and U signal registered in each color channel. Thus probes with low absolute signal intensity in both channels may be considered low confidence or as having failed [3, 27], and removing this failed probes is a standard approach. In addition to probe signal, 17 BeadArray quality tests published by Illumina are used to assess whether various steps in the array processing pipeline were successful [74]. In Section 2, we evaluated the BeadArray test performance of tens of thousands of samples from HM450K arrays, and we identified a subset of tests explaining most observed variation in outcomes across tests [3]. We also made these assay tests available in the updated `recountmethylation` [38] Bioconductor package. Besides these standard quality control and normalization procedures which are recommended for most DNAm array studies, methods for cross-study analyses are comparatively sparse and not well defined. Because of this, we explored applications of linear adjustment on study ID as a surrogate for unwanted variation such as from technical bias and batch effects (Sections 2.2.10 and 3.2.4).

1.3 NGS technologies and their applications

Much of modern computational biology has focused on analysis of sequencing data, especially sequencing data produced by NGS technologies. With consistent decreases in the cost of producing sequencing datasets, a wealth of public sequencing data is now available for a huge variety of populations, tissues, and conditions from a variety of species [26]. As preprocessing and analysis methods for sequencing data mature, nuanced analyses of alternative splicing, transcript isoform expression, and large structural variants are becoming more common. This section provides crucial background about the platforms and technologies we employed to study the reliability of tools for calling retained introns in short-read sequencing datasets [5] (Section 4).

1.3.1 NGS

NGS, sometimes called "massively parallel sequencing," is a large class of sequencing technologies which can simultaneously sequence millions of DNA fragments [147, 148]. First introduced in 2004, NGS has come to replace alternative technologies such as Sanger sequencing, which is sometimes called the "first generation" sequencing technology [149]. Unlike Sanger sequencing, microarrays, and other technologies, NGS doesn't require pre-knowledge of a gene sequence or genome target to be run. The reliability of NGS experiments can be fine-tuned either technically by increasing sequencing depth, or experimentally by using control runs and replicates. Among its chief downsides, NGS technologies fail to reliably capture certain genome regions, especially repetitive sequence and large structural variants. Further, NGS technologies require specialized equipment run by trained personnel, and that their analysis demands technical knowledge and usually considerable computing power. These issues have been alleviated somewhat as costs to generate NGS data have come down, as technical knowledge has become more widespread, and as computational methods have matured and become widely available and readily usable. Thus NGS allows nuanced and large-scale profiling of DNA or RNA in a sample, and it can now be used to profile millions of sequence variants or sequence the entire human genome in a matter of hours. Of importance to this dissertation is the distinction between NGS technologies which use short reads (Section 1.3.2) versus long reads (Section 1.3.3), as both data types are studied below [5] (Section 4).

There is often an interplay between the development of new technologies for omics data and the development of new methods to preprocess and analyze data from those technologies. Resources tracking tools for processing and analyzing NGS data have become increasingly common, and significant resources now track specific tools for single-cell sequencing [150], long-read sequencing [55, 151], and expression analyses [152]. Among newer analysis tools, it is increasingly common to see novel applications of machine learning algorithms, reapplication and appropriation of old methods for new data types, and transfer learning that leverages information from multiple complementary sequencing data types [54–56].

1.3.2 Short-read RNA-seq

Short-read sequencing technologies describe platforms using reads of up to several hundred base pairs in length [149]. Short-read data is typically low cost per sample, with relatively low error rates compared to long-read sequencing [147]. Many platforms are available to produce short-read data, and short-read sequencing has been widely used to profile many conditions and tissues across many species [44]. Short-read data typically has greater coverage at most genome regions compared to long-read data. The availability of mature and robust computational methods for preprocessing and analysis of short-read RNA-seq data enables many types of analyses, such as to quantify differential expression and alternative splicing, detect genetic mutations and small nucleotide variants or structural variants, and more [153–155]. Short-read sequencing data has several chief limitations. First, it is usually difficult or impossible to phase more than two genomic features due to the short read length. Further, coverage biases prevent reliable capture of structural variants and repetitive regions. Finally, raw FASTQ files produced by short-read platforms are often very large and may be difficult for private parties to host. Data from short-read platforms can be categorized as either single-end, meaning including reads from only one DNA strand, or paired-end, meaning reads are generated for both the forward and reverse DNA strands. Below, we analyzed public paired-end short-read sequencing produced for Illumina's NextSeq and HiSeq platforms, which we accessed from the public data archives of the SRA [5] (Section 4).

1.3.3 Long-read RNA-seq

Long-read sequencing technologies use reads of up to tens of kilobases in length. Long-read sequencing has historically been relatively high-cost to run, with relatively high error rate compared to short-read sequencing. While these technologies have been available for years, or over a decade in the case of PacBio single molecule real-time (SMRT) sequencing platforms, they have not seen widespread use until recently. This is mainly because of costs as well as the need for specialized tool development to preprocess and analyze long-read sequencing data [55]]. During sequencing, the PacBio long-read platform produces fluorophore flashes recorded as movie files, where the precise timing of recorded flashes indicates the presence of specific nucleotides and even specific chemically modified nucleotides such as 5m (Section 1.2.4). PacBio templates are circular molecules containing forward and reverse sequence joined with so-called “SMRTbells,” or ligated hairpin adapter sequences that are bell-shaped. Polymerization proceeds continually around this circular DNA template, producing a continuous strand of potentially many copies of the template sequence. As bases are held by polymerase, they produce a light pulse that is stored as a movie, where the pulse duration is specific to a particular base. These movie files are then processed into standard FASTQ format and analyzed [156].

Long-read sequencing has distinct advantages over short-read. First, long reads enable most or all of an entire transcript isoform to be captured including multiple functional elements phased together, offering an unprecedented look at isoform-level gene expression. Further, long reads can be better suited to capturing repetitive regions of the genome which were previously intractable for short reads [157]. Related to this, long reads can better capture structural variants, especially large and complex or multi-stage structural variants not reliably captured from short reads. This will have growing importance over time as structural variances are being increasingly studied for their roles in normal gene regulation and diseases including cancers [154]. Further, the cost of generating long-read data has decreased over time, and the data produced from long read platforms can be more compact and much easier to host and obtain than for short-read data. Despite these advantages over short-read data, long-read sequencing tends to be more error prone than short-read from a technical standpoint. Further, the cost of producing long-read data is still prohibitive for large-scale population profiling. Also, the low coverage of long-read data can confound transcriptome-level and differential expression analyses. In short, as costs decrease, methods for preprocess and analysis continue to improve, and applications continue to be introduced, long-read sequencing data is likely to become more popular and widely used in the omics literature [55].

1.4 Computational methods for analyzing public omics data

Technologies and methods for omics data have developed in parallel for decades. Publication of preprints, provision of runnable code examples in notebooks, and other key practices have been widely adopted in computational fields. Further, it is common for methods previously developed for older data types to be appropriated for newer data types. Alongside these tangible signs of progress there has also been an increasing push for open access code and reproducibility in computational disciplines. This section provides an overview of the key methods and tools which we used to conduct novel research and promote computational reproducibility in the work that follows.

1.4.1 Importance of reference genome annotations

Much of omics data analysis depends on underlying assumptions made about the genome. For instance, array design is defined by the identification of a marker that can be probed. While this benefits the design of a reliable probe assay to quantify an established marker of interest, it typically prevents possible discovery of new markers. By contrast, high-throughput sequencing technologies like NGS platforms can yield a more faithful snapshot of a biological system because they aren't targeted to a previously defined region. However, both sequencing data and array data commonly rely on the use of reference genomes and genome annotations. Periodic revisions and updates to these references can challenge us to consider when annotations might require revision in light of new evidence, and when analyses may hinge too greatly on key assumptions about the genome.

For many sequencing data pipelines, reference genomes inform DNA library construction prior to sequencing runs (upstream) and they enable read alignment prior to analysis (downstream). The reference human genome sequence and its accompanying annotations are periodically updated to reflect new understandings of gene expression patterns, protein products, transcript isoforms, intergenic regions, structural variants, and more [158]. For our efforts to analyze RIs, updates to genome annotations can cause uncertainty in how we define an intron (e.g. when is an intron retained sufficiently often that it needs to be considered an exon instead?). Further, the rise in available long-read sequencing data has expanded our view of the spliceome by showing greater amounts of alternative splicing and diverse isoform expression than was previously recognized from short-read data [5, 55] (Section 4).

The reference genome and its annotations are further used to determine which CpG loci to target and what target sequence to use when probes are designed for DNAm array platforms. These annotations are included in the platform manifests [159–161] which are in turn used to determine probe properties like whether it resides in a gene's promoter or body, whether it is in a CpG island or OpenSea region, and whether the probe sequence contains any high-frequency single nucleotide variants. Unfortunately, a precise probe annotation is frequently confounded by overlapping transcripts, as probes can simultaneously map to the promoter in one transcript isoform and the body in another. Further, the most widely used DNAm array platforms were developed using the hg19 human genome version, first released in 2009, but many newer omics technologies use the more recent hg38 genome version, first released in 2013 [162]. To address genome version discrepancies prior to integrative analyses, it is necessary to convert genome coordinates from one version to the other in a process known as "lifting over." Unfortunately, this process is not always straightforward and can be subject to uncertainty with how exactly two regions across genome references relate to each other [163]. Thus genome references and annotations inform not only how omics technologies are designed, applied, and analyzed, but also how we think about the definition of functional genome regions.

1.4.2 Programmatic omics data downloads using `equery`

As mentioned above (Section 1.1.3), several software tools enable programmatic query and download from NCBI-maintained public data repositories. These tools include Entrez Programming Utilities [18] for submitting queries the GEO API and other databases, and the SRA-Toolkit [19] for managing downloads of FASTQs and other related files for sequencing experiments. In general, study and sample metadata could be obtained by piping calls to the `equery` function from Entrez Programming Utilities, and these IDs could be used to construct URLs to download supplemental data and metadata from GEO DataSets, or these IDs could be used in calls to `fastq-dump` to obtain FASTQs from the SRA.

Despite the availability of excellent browser interfaces for querying NCBI databases, we avoided manually downloading datasets and metadata unless absolutely necessary. This is because browser interfaces may change over time, and manually interacting with a browser to download thousands of samples is not a viable option. Instead, we wrote scripts to automate calls to query tools and manage downloaded data programmatically. We used filters with queries looking up the GEO GPL IDs for DNAm array platforms in order to identify records with raw array image IDAT files in their supplement fields (Appendix A.1). Scripting saved considerable time over manual data access, and is reproducible so long as fundamental database schema and APIs aren't drastically modified. In summary, programmatic resources for accessing public omics data repositories can save time in querying and obtaining sample records and datasets while further encouraging reproducible research (Section 1.1.10).

1.4.3 Normalization of DNAm arrays using the noob method

Preprocessing is a necessary and routine data manipulation step prior to omics data analysis. It often entails some combination of quality assurance of assays and samples, normalization of assay signals, and explicit or implicit measures to correct for batch and other potential sources of bias. Preprocessing DNAm array data in Sections 2 and 3 entailed all of these steps prior to cross-study analyses. We normalized DNAm arrays using out of band signal with a technique called "noob-normalization" [140]. Unlike other normalizations for DNAm array data, noob-normalization is performed within rather than across samples, and it can be effectively combined with other normalization techniques [142, 144, 145]. This means noob normalization can be routinely performed on data compiled across studies and conditions, regardless of which subset of the data is used in a particular analysis [3, 4].

We can describe noob-normalization in greater detail by paraphrasing from [140]. First, we define the background signal X_b as:

$$X_b = N(\mu, \sigma)\,. \tag{1}$$

Where μ is the mean and σ is the variance, and each of these background parameters is estimated from the distribution of available control probes. We then define the signal distribution X_S:

$$X_S = Exp(\gamma)\,. \tag{2}$$

Where γ is the signal parameter remaining from subtracting mean background signal from foreground signal ($X_f - \mu$). Next, we define the foreground intensity X_f in terms of both X_b and X_S:

$$X_f = X_b + X_S \,. \tag{3}$$

We then calculate the conditional expectation for the Infinium probe signal, $E[X_S|X_f]$, in terms of $\mu_{S,f}$ as follows:

$$\mu_{S,f} = X_f - \mu - \frac{\sigma^2}{\gamma} \,. \tag{4}$$

$$E[X_S|X_f] = \mu_{S,f} + \sigma^2 \frac{\phi_s(0; \mu_S, \sigma^2)}{1 - \phi_c(0; \mu_{S,f}, \sigma^2)} \,. \tag{5}$$

Where ϕ_s is the standard normal distribution and ϕ_c is the cumulative normal distribution. This estimation borrows from prior work in [164]. Note [140] also used this calculation on the negative control probes to define the "normexp" normalization method.

1.4.4 Overcoming the curse of dimensionality for association studies

For over a century it has been recognized that having many variables with relatively few observations is a specific experimental circumstance causing a unique set of methodological issues. Among the most important of these issues is that as the dimensionality of a dataset grows, the necessary computations to analyze that data may grow exponentially. Related to this concern, the size of the dataset may grow exponentially even while the data themselves become relatively sparse. Further, increasing dimensionality may introduce redundant, uninformative, or noisy measurements which need to be filtered or otherwise dealt with through additional analyses. These issues collectively describe the "curse of dimensionality" [165].

Several key issues arise from this curse. First, many studies use large association experiments that apply numerous statistical tests to isolate highly significant variables from high-dimensional datasets. Unfortunately, as the test count increases, so does the likelihood of drawing false conclusions from the evidence. The error types of principal concern are called Type 1 errors (the case of falsely rejecting the null hypothesis when it's true) and Type 2 errors (the case of falsely accepting the null hypothesis when the alternative hypothesis is true). Further, the emphasis on identifying a very few highly significant results may cause one to omit variables with marginal effects which may nonetheless be explanatory or informative [166]. Finally, analyses handling high-dimensional data can entail the use of very large file sizes and memory demands, and standard statistical functions may need to be strategically utilized in parallel (Appendix A.9) or to reference stored data outside of active memory.

Of key importance for this paper are EWAS, or association studies that commonly use DNAm arrays. Similar to genome-wide association studies, EWAS often include tests using hundreds of thousands of loci measurements as variables, with observations drawn form perhaps tens or hundreds of samples. Commonly tested conditions include tissue type, disease status, or condition subgroup, where the goal is to identify a subset of highly significant and robust molecular markers explaining or predicting the condition of interest. Many of the available DNAm array datasets are accompanied by published EWAS analyses [3, 4]. Fortunately, many methods previously applied in previous association studies can be effective for novel EWAS, including various types of P-value adjustments. P-value adjustments are used increase the stringency of the significance threshold beyond which an outcome is considered significant, and more stringent thresholds may be used when more tests are conducted. Below, we applied the Benjamini-Hotchberg [167] false discovery rate (FDR)-based P-value adjustment when conducting cross-study EWAS. In addition to increasing test stringency thresholds, the total tests can be reduced either by pre-filtering tested probes or markers (e.g. based on high absolute effect sizes, etc.) or by other means known as dimensionality reduction techniques (see below, Sections 1.4.5 and 1.4.6). Further, initial tests can use methods like ANOVA or penalized regression to adjust for many markers simultaneously, and these can be followed up with more focused tests involving subsets of only the most significant variables [168].

To better understand and interpret the results of large-scale association studies, several key assumptions need to be mentioned. The first assumes test independence, or that each tested locus or assay can be treated as a truly independent and not reliant on the status of other tested loci. This assumption is often weak or false if we consider information such as gene co-expression networks, regionally correlated DNAm patterns, epigenetic loci which may mediate gene expression in $trans$, and more. Another key assumption is that high test significance means the tested assay has greater biological importance. In fact, a highly significant outcome for an assay may not have any causal relation to the studied condition at all. To show causality, follow up tests called validations are needed, and these should be independent or orthogonal to the original data used to discover a trend. There are many approaches to validation, and independent validation or replication of published differentially methylated probes (DMPs) is of particular importance below (Sections 3.6).

Finally, there is evidence that high dataset dimensionality is not always a bad thing, and may even be a blessing in disguise [165, 169, 170]. The fact that most omics data is high-dimensional can mean the data is useful in the future for testing new hypotheses with more effective and sophisticated analysis methods. Further, as the quantity of public omics data continues its steady increase, future experiments can leverage greater sample quantities to improve statistical power and conduct new analyses to, for instance, better detect the influences of loci with marginal effects on a condition of interest [166]. These prospects underscore the importance of efforts to continue monitoring, summarizing, and compiling available public omics datasets.

1.4.5 Principal component analysis helps to identify key variance sources

Principal component analysis (PCA) is a widely used dimensionality reduction method. It is non-parametric, meaning it does not assume the properties and assumptions of parametric methods, and it is unsupervised, meaning all dataset variables and observations are considered equally important [171]. These properties help PCA to be highly generalizable and useful across a variety of analysis settings. PCA is closely related to the singular value decomposition method, and occasionally the two methods are referred to interchangeably [172]. PCA is highly computationally efficient because of several key simplifying assumptions and conditions, namely that the dataset variance patterns can be represented as linear functions, and that means and variances are sufficient to represent the meaningful information contained in the data. Each of these assumptions has its drawbacks, as variance patterns may violate the assumption of linearity, and the mean and variance may not be sufficient to explain the data [171].

Determining the principal components for a dataset in PCA chiefly involves centering the data then performing singular value decomposition [172]. Centering involves subtracting the distribution means from each point separately for each variable, which often helps the first component to be correctly identified. Centering is so ubiquitous that functions for PCA, such as the `prcomp()` function in `stats` base R package, may perform data centering by default. The specific application of singular value decomposition for PCA involves finding the set of dimensions, or components, which explain the most linear variances in the data. This step usually entails specification of a unit vector constraint that allows reformulation of the problem into an optimization problem with an unconstrained solution [173]. PCA yields a set of orthogonal, or non-correlated, dimensions consisting of eigenvectors, or the component directions, and eigenvalues, or each observation's value for each component. The component number is its presumed importance and is assigned according to the magnitude of variance it explains. Thus the first and most important principal component explains the most variance, the second component is the second most important and explains the second largest amount of variances, and so on. Note that "importance" here strictly relates to the dataset variances rather than any particular interpretation of the biological system or clinical setting. Often, follow up analyses help elucidate how biological, clinical, and other important variables contribute to each principal component in an omics dataset.

It is common practice to study the top components from PCA using scree plots. These are a type of bar plot showing the magnitude of variance explained by each of the top components and how sharply this variance drops off after the first few components. Scree plot examples can be found in the chapters below (Sections C.3 and 3.3b). It is also common to perform clustering analysis on scatter plots in the dimensions of the top few components. This is intended to reveal underlying similarities across observations that may reflect the influence of variables of interest and/or confounding factors. Example scatter plots and cluster analyses of top PCA components from autosomal DNAm can be found below (Sections 2.3 and C.4).

1.4.6 Using feature hashing to analyze large datasets

In addition to PCA, less common dimension reduction methods include feature hashing. Feature hashing is otherwise known as the "hashing trick" because it expands on prior data manipulations called kernel transformations or the "kernel trick" [174]. The kernel-trick was previously applied for implicit comparison of high-dimensional objects, without having to fully realize those objects in memory. Calculating a kernel matrix can have prohibitive memory costs when datasets are very large and high-dimensional. Feature hashing, or calculation of the hashed kernel, enables k matrix calculations in big data settings by projecting input vectors x with unmodified dimensional space $\mathbb{R}^d$ into the low-dimensional space $\mathbb{R}^m$ with some hash function ϕ [175].

Feature hashing functions, or “hash functions,” have the advantage of being very fast to implement even for thousands of high-dimensional data points. Hash functions have several important characteristics. Among the most important of these is they are deterministic where the same function applied to the same data will always produce the same result. Further, hash functions evenly distribute similar values in a dataset across new values such that similar data becomes widely separated. Hashing may further be effectively uni-directional where it is impossible to recover the original data even if both output and function are known, a useful feature for cryptography. Finally, hash functions may cause overlaps, or “collisions,” where the same value is assigned to distinct data points. Worthwhile hash functions take precautions to mitigate or avoid collisions, such as by increasing the size of the hash table or the pool of possible hash values.

Below, we implemented a signed hash function [174]. As mentioned, the introduction of a sign effectively increases the hash table to discourage collisions. Details about this function are provided in Appendix A.10. We used this signed hash function to reduce the dimensions of a large DNAm array dataset, and we performed PCA on the hashed features to identify sample clusters as well as quantify the relative importance of biological, demographic, and technical variables [3, 4].

1.4.7 `HDF5`-based file formats enable efficient use of large data compilations

Large datasets are now commonplace in omics disciplines. Not only are there data from tens or hundreds of thousands of individuals, but the volume of data for even a single individual for RNA-seq platforms can amount to gigabytes worth of information. This means a key issue for utilization of public omics data is how best to handle such large datasets in a manner that conserves time and disk space. Early work on `recountmethylation` provided access to large static database files containing uniformly processed public data. To store data in a format conducive to rapid queries and filters, we used formats incorporating the Hierarchical Data Format 5 (HDF5) [176] file format. `HDF5` combines data compression and data chunking to reduce occupied disk space enable rapid look-ups within subsets of the data, respectively. The `HDF5` data class has many other varied applications for omics data as well, including for efficient storage of nanopore long-read sequencing data [55].

An important feature of the `HDF5` data type is that it’s supported across many programming environments, including Python and R. It is further augmented with other data types, including the `SummarizedExperiment` class, for use with Bioconductor packages. `SummarizedExperiment` objects are specialized for storage and retrieval of assays and related data for omics experiments, and it features multiple slots for various information like assay data, platform manifests, hardware versions, and sample and experiment metadata. The hybrid object class `HDF5-SummarizedExperiment` combines benefits from each of its constituent features. Backend support for this hybrid object is provided by `DelayedArray`, a technology that uses caching to expedite certain summary operations and delay certain memory intensive operations like filtering and normalization. This hybrid object class is also optimized for handling sparse matrices, or highly-repetitive assay matrices containing many 0 or near-0 values, which are commonly encountered in omics fields. Support for `DelayedArray`-powered `HDF5-SummarizedExperiment` objects is provided by the `HDF5Array` [177] and `rhdf5` [178] R/Bioconductor packages.

1.4.8 Simulation-based power analyses of DNAm arrays

Statistical power is the ability to correctly reject the null hypothesis when the alternate hypothesis is true. Power analysis, or the determination of the number of samples to collect to achieve some minimum power threshold (usually 80%), is an important initial step for experiment planning. Few studies have implemented power analyses for DNAm arrays, and these did not present a generalizable method or computational tool for this purpose [168, 179, 180]. More recently, the `pwrEWAS` [181, 182] R/Bioconductor package was written for generalizable DNAm array power analysis across tissues. `pwrEWAS` allows for simulation-based power analyses informed by empirical DNAm means and variances from CpG loci. While the original work was restricted to a set of pre-computed CpG means and variances, we lightly modified it to run on new data including our de novo cross-study and -platform compilation of blood autosomal DNAm data. We showed how to use this modified code in an update to the `recountmethylation`. In practice, the CpG probe summary statistics can be calculated for a particular tissue, disease, condition, or organism to ensure the power analysis is relevant to the conditions being studied in a new experiment.

The workflow for `pwrEWAS` starts with data generation from a set of empirical CpG summary statistics. Next, differential DNAm analysis is carried out using test parameters specified by the user, including expectations for the DNAm differences, the type of statistical test used, and the range in total samples involved. Finally, a power evaluation is performed and a number of summary statistics as well as the N-to-power models for each delta value are returned [181]. We employed the `pwrEWAS` method to compare power-to-sample size trade-offs across cross-study compilations of DNAm from normal blood and prevalent blood sample types (Sections 3.2.7 and 3.5).

We can describe the simulation-based approach to power analysis from the `pwrEWAS` method by paraphrasing the description in [181]. First, the population mean μ and variance σ are calculated for a CpG probe j across some set of samples $i \in \{1...N\}$ for a specific tissue and condition of study (e.g. normal whole blood from adults):

$$\hat{\mu_j} = \frac{1}{N}\sum_{i=1}^{N}\beta_{i,j}\,. \tag{6}$$

$$\hat{\sigma_j^2} = \frac{1}{N-1}\sum_{i=1}^{N}(\beta_{i,j} - \hat{\mu_j})^2)\,. \tag{7}$$

Next, P pairs of CpG means and variances, $(\hat{\mu_j}, \hat{\sigma_j^2})$, are sampled with replacement from the reference dataset where $P = 100,000$ by default. Some DNAm difference δ_β is then imposed on K CpG probes, where $K \leq P$. That is, the δ_β term is applied to the mean DNAm of one simulated comparator group but not the other. Since the technical range of β is $0 \leq \beta \leq 1$, the modified mean DNAm is drawn from a truncated normal distribution.

The standard deviation of simulated differences τ is then calculated from the specified target δ_β and the specified target differentially methylated CpG probes. This is accomplished by stepwise simulation of $P(\delta_{\beta,k})$ 100 times while adjusting τ such that τ falls within the intended absolute δ_β distance within a detection limit ± 0.005.

The value K is calculated in terms of T target total differentially methylated CpG probes, and F observed fraction of probes which are differentially methylated:

$$K = \frac{T}{F}\,. \tag{8}$$

The means and variances of both comparator groups are then used to calculate shape a_j and b_j of a beta-distribution:

$$a_j = \mu_j^2 (\frac{1 - \mu_j}{\sigma_j^2} - \frac{1}{\mu_j})\,. \tag{9}$$

$$b_j = a_j (\frac{1}{\mu_j} - 1)\,. \tag{10}$$

Next, N_1 and N_2 observations corresponding to each comparator group for the individual simulated CpG are generated from the beta-distributed observations. This yields two simulated β-value matrices, $P \times N_1$ and $P \times N_2$. These simulated value matrices are then used in differential methylation tests with three possible outcomes:

A. No differential methylation, or zero difference ($\delta_{\beta,k} = 0$)

B. Differential methylation with negligible, non-zero difference ($\delta_{\beta,k} < 0.01$)

C. Differential methylation with meaningful difference ($\delta_{\beta,k} \geq 0.01$)

Next, a chosen differential methylation test is applied to the simulated datasets. The outcome is compared to the actual differential methylation status from the simulated value matrices above. This comparison is then used to determine six categories.

Detected CpG probes fall into one of the following three categories:

1. True positive (TP) Probes with meaningful difference

2. Neutral positive (NP) Probes with negligible, non-zero difference

3. False positive (FP) Probes with zero difference

Undetected CpG probes fall into one of the following three categories:

4. True negative (TN) Probes with zero difference

5. Neutral negative (NN) Probes with negligible, non-zero difference

6. False negative (FN) Probes with meaningful difference

For a given False Discovery Rate (FDR) threshold, the empirical marginal power is calculated as:

$$marPower = \frac{TP}{TP + FN}\,. \tag{11}$$

The empirical classical power, calculated as the ratio of correctly detected probes to all differentially methylated probes:

$$classicalPower = \frac{NP + TP}{NP + NN + TP + FN}\,. \tag{12}$$

Further, the False Discovery Cost (FDC) is calculated as:

$$FDC = \frac{FP}{TP}\,. \tag{13}$$

1.4.9 Resources and methods that promote computational reproducibility

As available computing power increases, analysis methods mature and improve, and available omics datasets become larger and more complex, there are many tools one can employ to promote reproducibility in computational disciplines, or simply "computational reproducibility." Utilization of these tools in methods or supplement can promote methods transparency while ensuring code can be repeatedly run either on the same dataset or extended for different data inputs. An ideal for this pursuit is to include full open-access code along with the necessary data and metadata to repeat an analysis. Where this is done successfully, methods developed for older data types have even seen new life through novel applications with new data types. This section describes several of the tools and measures researchers use today to ensure computational reproducibility.

Virtual environments provide a sequestered digital space where software dependencies can be tightly controlled. Dependencies can be specified in a script and downloaded from a selection of hosts. Virtual environments can enable a user to successfully run even old software that is no longer maintained. They also simplify computational tasks, since the researcher can keep their preferred software shortcuts and versions on their main system while running an older tool in its own environment. We used the `conda` [183] software to set up virtual environments for several RI-detection tools (Appendix A.6), and we further exported these scripts in YML format (Appendix A.6) so they could be rapidly set up again in the future. We further provided a `conda` script to obtain all the proper dependency versions to run the `recountmethylation_instance` [61] workflow. Containers are another class of tools related to virtual environments. By contrast to virtual environments, containers are treated as discrete and self-contained virtual drives. This can lead to permission issues when running a container using a remote server where the user lacks admin privileges. Below, we were able to conveniently run the `IRFinder-S` [184] RI-detection tool from a Docker container. In summary, while virtual environments and containers each has benefits and downsides, both resources enable deployment of code or software with its dependencies.

As the complexity of omics datasets and analysis tasks increases, the need to carefully document and control intermediate preprocessing steps has also increased. This can be accomplished with workflows, or abstract recipes specifying specific inputs and outputs, and which may incorporate verbose automatic system logging. Workflows have been used in omics disciplines for years, and they have been implemented for many tasks including whole genome sequencing [185], differential RNA-seq expression [186, 187], variant calling [188], and more. They encourage reproducibility through explicit declaration of dependencies and setting restrictions for the inputs and outputs of pipeline steps, all of which helps to ensure code can be repeatedly run on the same data or adapted for new data [60]. Specific workflow definition standards and details have been laid out by the common workflow language project [189]. Deploying an analysis pipeline as a workflow enables a complex task to be broken into many smaller, well-defined tasks which can be run individually, which can aid with troubleshooting and encourages repeated running of a pipeline over time. Workflow development may be supported either by web-based platforms such as Galaxy [190] or specialized workflow management engines [191].

Several workflow languages and systems are commonly used, of which Snakemake [192] is most relevant here. Importantly, Snakemake implements `conda` [183] virtual environment support, and thus exemplifies how multiple tools that promote computational reproducibility can be implemented together effectively. We wrote a Snakefile describing the workflow for the `recountmethylation_instance` (Appendix A.7), which informs compilation of public DNAm array datasets for use with the `recountmethylation` [38] Bioconductor package. This script provides rules which utilize two specialized resources: `recountmethylation_server` [193] for server queries and downloads; and `recountmethylation.pipeline` [194] for compiling and preprocessing large compilation files.

Among the key obstacles for computational reproducibility is interoperability, or the ability to run functions or software across distinct systems and system configurations. Certain resources enable interoperability by making methods and functions in one environment callable from another environment. For example, the R packages `basilisk` [195] and `reticulate` [196] pair to provide a means of calling Python functions from an R environment in a manner that is reproducible and predictable (Appendix A.8). The `basilisk` package enables management of `conda` environments and Python dependencies, while the `reticulate` enables loading and calling Python functions from an R session. These packages can be combined to dynamically manage code from R and Python in a single script. These and other resources for interoperability can save time and effort by discouraging redundant redevelopment of effective programming tools across multiple programming environments.

Parallel processing is when a process is run using multiple simultaneous subprocesses to save on time and memory resources. For example, if the goal is to apply some function to every row in a large matrix, one could more rapidly process subsets of rows simultaneously, or "in parallel," rather than apply the function to each row in succession from a single session. Parallel processing encourages computational reproducibility because it ensures complicated processes are broken down into smaller faster-running processes, as these can be more readily troubleshooted when problems arise. The exercise of writing a script with parallelization can further challenge the programmer to compartmentalize each step in a complex process, which leads to more legible code. While limited parallelization is possible using modern personal computers, the full potential of parallelization is realized in remote server environments. Remote server environments frequently enable the highly flexible specification of process compute parameters, including the number of cores and amount of memory for a given process. We implemented parallelization for study ID bias adjustment simulation experiments (Appendix A.9) and to conduct power analyses using `pwrEWAS` [181] (Sections 3.2.4 and 3.3.2).

Runnable code examples can be conveniently provided as methods or supplement in the form of vignettes or notebooks. Notebooks enable researchers to document code, data, and analyses, and they encourage implementation of small runnable examples that spot check key steps in complex computational pipelines. For the `recountmethylation` [38] Bioconductor package, we used R Markdown to produce vignettes on a variety of topics, from reproducing analyses from the study [3], to showing how to perform power analysis with `pwrEWAS` [181, 182], to showing how to infer genetic ancestry using the GLINT software [32], and more [4]. Several vignettes further use `basilisk` [195] to manage a `conda` environment and `reticulate` to call Python code, all from the comfort of an R session. Thus vignettes can simultaneously reproduce published analyses while showing operations in a transparent way that can be more widely useful to the research community.

Progress on research projects should be tracked using version control systems like `git`, and should be periodically backed up to the cloud with resources like GitHub [57, 197]. This not only ensures that valuable effort isn't accidentally lost, it also enables other researchers to quickly fork and modify a project while keeping changes contained and preserving the original work. In summary, computational reproducibility is promoted through the use of virtual environments, containers, workflows, and notebooks, and utilization of version control, cloud backups, and interoperability measures can ensure a project is accessible and runnable in the future.

1.5 Chapter summaries

This sections provides abstracts that summarize the motivation, methods, and results for the three following chapters. Chapter 1 (Section 2) describes work that was previously published in the journal *NAR: Genomics and Bioinformatics* [3]. Chapter 2 (Section 3) describes work from the preprint [4], which has been submitted to the journal *Nature Communications*. Chapter 3 (Section 4) describes work from the preprint [5], which is currently under peer review for the journal *Genome Biology*.

1.5.1 Chapter 1: Human methylome variation across Infinium 450K data on the Gene Expression Omnibus

While DNA methylation (DNAm) is the most-studied epigenetic mark, few recent studies probe the breadth of publicly available DNAm array samples. We collectively analyzed 35 360 Illumina Infinium HumanMethylation450K DNAm array samples published on the Gene Expression Omnibus (GEO). We learned a controlled vocabulary of sample labels by applying regular expressions to metadata and used existing models to predict various sample properties including epigenetic age. We found approximately two-thirds of samples were from blood, one-quarter were from brain and one-third were from cancer patients. About 19% of samples failed at least one of Illumina's 17 prescribed quality assessments; signal distributions across samples suggest modifying manufacturer-recommended thresholds for failure would make these assessments more informative. We further analyzed DNAm variances in seven tissues (adipose, nasal, blood, brain, buccal, sperm and liver) and characterized specific probes distinguishing them. Finally, we compiled DNAm array data and metadata, including our learned and predicted sample labels, into database files accessible via the `recountmethylation` R/Bioconductor companion package [38]. Its vignettes walk the user through some analyses contained in this paper.

1.5.2 Chapter 2: `recountmethylation` enables flexible analysis of public blood DNA methylation array data

Thousands of DNA methylation (DNAm) array samples from human blood are publicly available on the Gene Expression Omnibus (GEO), but they remain underutilized for experiment planning, replication, and cross-study and cross-platform analyses. To facilitate these tasks, we augmented our `recountmethylation` R/Bioconductor package with 12,537 uniformly processed EPIC and HM450K blood samples on GEO as well as several new features [38]. We subsequently used our updated package in several illustrative analyses, finding (1) study ID bias adjustment increased variation explained by biological and demographic variables, (2) most variation in autosomal DNAm was explained by genetic ancestry and CD4+ T-cell fractions, and (3) the dependence of power to detect differential methylation on sample size was similar for each of peripheral blood mononuclear cells (PBMC), whole blood, and umbilical cord blood. Finally, we used PBMC and whole blood to perform independent validations, and we recovered 40-46% of differentially methylated probes (DMPs) between sexes from two previously published EWAS.

1.5.3 Chapter 3: Retained introns in long RNA-seq reads are not reliably detected in sample-matched short reads

There is growing interest in retained introns in a variety of disease contexts including cancer and aging. Many software tools have been developed to detect retained introns from short RNA-seq reads, but reliable detection is complicated by overlapping genes and transcripts as well as the presence of unprocessed or partially processed RNAs. We compared introns detected by 5 tools using short RNA-seq reads with introns observed in long RNA-seq reads from the same biological specimens and found: (1) significant disagreement among tools (Fleiss' $\kappa = 0.231$) such that 52.4% of all detected intron retentions were not called by more than one tool; (2) that no tool achieved greater than 20% precision or 35% recall under generous conditions; and (3) that retained intron detectability was adversely affected by greater intron length and overlap with annotated exons.

2 Chapter 1: Human methylome variation across Infinium 450K data on the Gene Expression Omnibus

2.1 Background

DNA methylation (DNAm) has been widely studied for its roles in normal tissue development [69, 198–200], biological aging [30, 86, 113], and disease [70, 72, 201–203]. DNAm regulates gene expression, either in *cis* if it occurs in a gene's promoter, or in *trans* if it overlaps an enhancer or insulator [69–71]. Whole-genome DNAm (or "methylome") analysis, especially in the form of an EWAS, is a common strategy to identify epigenetic biomarkers with potential for clinical applications such as in prognostic or diagnostic panels [136, 168, 204].

Most investigations probe DNAm with array-based platforms. Published DNAm array data and sample metadata are commonly available through several public resources. These include cross-study databases like the Gene Expression Omnibus (GEO) [9, 205] and ArrayExpress [7], as well as landmark consortium studies like TCGA [15] and the ENCODE [10, 11]. Recently published databases and interfaces provide access to samples from these sources [33–37].

While over 1,604 HM450K array studies and over 104,000 samples have been submitted to GEO since 2009 (Fig. B.1), there have been few attempts to rigorously characterize technical and biological variation across these studies. In 2013, two studies independently compiled DNAm array samples from GEO and elsewhere, analyzing epigenetic age across tissues and diseases [30], and investigating cross-study normalization [28]. More recent cross-study analyses include [206] from 2018, which evaluated metadata and sample quality across 8,327 DNAm array samples, and [48] from 2020, which validated sperm-specific DNAm patterns using 6,288 samples.

While the GEO website provides access to submitted experiment and sample metadata, the metadata are not necessarily structured and require harmonization to facilitate cross-study analyses. There are currently no R/Bioconductor [207] packages providing access to uniformly normalized array data across GEO studies accompanied by harmonized metadata. It should also be noted that most GEO studies do not include raw IDAT files, which are needed to uniformly normalize samples and thus limits their utility for novel cross-study analyses.

The vast majority of GEO DNAm array data is composed of samples using Illumina's HM450K BeadArray platform. Restricting attention to HM450K samples with IDATs published on or before March 31, 2019, we identified 35,360 samples from 362 studies, over three times the number of samples studied by either [30], [28], or [206]. From sample IDATs, we extracted raw signals and probe significance data, derived quality metrics from control probe data, and performed normalization on out-of-band signal with the noob method [140]. We also learned a controlled vocabulary of sample labels by applying regular expressions to metadata and used existing DNAm array-based models to predict sex, epigenetic age, and blood cell fractions [27, 30, 31]. We conducted analyses investigating the performances of standard quality assessments and identified studies with frequent failed samples. Finally, we characterized autosomal DNAm variation in 7,484 samples from seven non-cancer tissue types. This analysis complements recent independent efforts to quantify tissue-specific DNAm patterns [48] and showcases several of the relatively rare sample types we compiled from GEO (e.g. sperm, adipose, and nasal).

To aid other investigators interested in reanalyzing DNAm array data from GEO, we compiled raw and noob-normalized DNAm array data with our learned and predicted metadata into HDF5-based databases accessible using `recountmethylation`, a companion R/Bioconductor [207] package available at `https://doi.org/doi:10.18129/B9.bioc.recountmethylation`. Use of this package is covered thoroughly in accompanying vignettes, which also reproduce some of the results contained in this paper.

2.2 Methods

2.2.1 Discovery and download of DNAm array IDATs on GEO

We used the `esearch` function of Entrez Programming Utilities v10.9 to search for every HM450K sample published to GEO as of March 31, 2019 for which two `gzip`-compressed IDAT download URLs were available (Appendix A.1 and A.2). We ultimately downloaded IDATs for 35,360 sample records. Search and download were performed using the script `https://github.com/metamaden/recountmethylation_server/blob/master/src/server.py`. Note, the HM450K platform accession ID, GPL13534, is specified in the script `https://github.com/metamaden/recountmethylation_server/blob/master/src/settings.py`, and changing this will cause the server to target a different array platform.

2.2.2 Preprocessing of DNAm array IDATs on GEO

We preprocessed DNAm array IDAT pairs for 35,360 HM450K samples on GEO using the R/Bioconductor package `minfi` v1.29.3 [27], applying the normalized exponential out-of-band probe method (i.e., noob normalization) in the analysis pipeline at `https://github.com/metamaden/recountmethylation.pipeline`. The noob normalization technique mitigates run-specific technical biases and precedes batch- and/or study-level normalization steps [140].

2.2.3 Quality control results

We computed 19 quality metrics from red and green color channel signals for HM450K samples (Table 6.2). To obtain the 17 BeadArray controls, we referred to Illumina's official documentation [74, 208] as well as methods in the `ewastools` v1.7 package [206]. For our calculations, we used signal from the extension green control as background, and we used a denominator offset of 1 where it would otherwise be 0 (Supplemental Information) [206]). These calculations were done with the script `https://github.com/metamaden/recountmethylationManuscriptSupplement/blob/main/R/beadarray_cgctrlmetrics.R`. We thereby obtained a binary matrix of outcomes across the 17 BeadArray controls, where pass = 1, and fail = 0, on which we performed principal component analysis (PCA) using the "prcomp" R function from the stats v4.0.2 R package. We then used ANOVAs to determine the variances explained by each control for each component, and we obtained stacked barplots of component variances with `ggplot2`

(Fig. B.4). The script `https://github.com/metamaden/recountmethylationManuscriptSupplement/blob/main/inst/scripts/figures/figS4.R` reproduces our steps.

We subsequently computed array-wide log_2 median methylated (M)) and log_2 median unmethylated (U) signals, as reproduced in the `recountmethylation` data analysis vignette at `https://www.bioconductor.org/packages/release/bioc/vignettes/recountmethylation/inst/doc/recountmethylation_data_analyses.pdf`.

2.2.4 Obtaining sample metadata

Sample metadata was downloaded from GEO as study-level SOFT files using the script `https://github.com/metamaden/recountmethylation_server/blob/master/src/dl.py`. From SOFT files, sample-level metadata were extracted as JSON-formatted files. Study-specific metadata fields were filtered prior to learning sample annotations (2.2.5). These steps were performed using the scripts at `https://github.com/metamaden/recountmethylationManuscriptSupplement/tree/main/inst/scripts/metadata` (Appendix A.3 and A.4).

2.2.5 Learning sample annotations

We took a partially automated approach to learn sample annotations from mined metadata. Our annotations were inspired by those in `marmal-aid` [28] and included disease/experiment group, age, and sex. To learn labels, we first coerced SOFT-derived metadata into annotation terms, then used manually constructed regular expressions to extract new labels (Supplemental Materials).

2.2.6 Learning sample type predictions

We learned additional metadata using the MetaSRA-pipeline (`https://github.com/deweylab/MetaSRA-pipeline` [17] , Table 6.1, [29]). This pipeline uses natural language processing to map sample metadata to curated ontology terms from the ENCODE project. It returns mapped terms and sample type confidences for each of six categories. We retained categories with the highest-confidence predictions as the most-likely sample types (Supplemental Materials, Fig. B.2, and Supplemental Information).

2.2.7 Model-based metadata predictions from DNAm

After noob normalization, we performed model-based predictions of sample age [30], sex [27], and blood cell type fractions [31] using the `minfi` (v1.29.3) and `wateRmelon` (v1.28.0) R/Bioconductor packages in our script `https://github.com/metamaden/recountmethylationManuscriptSupplement/blob/main/inst/scripts/metadata/metadata_model_predictions.R`. We tested concordance of mined and predicted sex and age to inform the use of these predictions and reliability of learned annotations (2.3).

2.2.8 Principal component analyses of autosomal DNAm

We performed array-wide approximate PCA with the `stats` v3.6.0 R package, using noob-normalized autosomal DNAm from all samples and a subset of filtered samples from seven non-cancer tissues (Beta-values, Fig. 2.3 and Fig. B.6). Missing values were imputed by array-wide DNAm medians (Beta-value scale) within samples. To improve computational efficiency, we first applied feature hashing (also known as the hashing trick, Appendix A.10) [174, 209] to project the normalized Beta-value arrays into an intermediate reduced space before performing PCA. PCA results were visually almost identical whether we invoked an intermediate dimension of 1,000 or 10,000 (results not shown). We used the 1,000-dimension mapping for analyses in Fig. 2.3 (data provided in Supplemental Files). The above analysis steps are shown in the script `https://github.com/metamaden/recountmethylationManuscriptSupplement/blob/main/inst/scripts/analyses/pca_analysis_fig3.R`.

2.2.9 Annotation of studies for cross-tissue DNAm variability analyses

We identified samples of seven distinct tissues (adipose, blood, brain, buccal, liver, nasal, and sperm), where each tissue included at least 100 samples across at least 2 study records (Supplemental Materials). While we noted sufficient samples from placenta (study accessions GSE100197, GSE71678, and GSE74738), these were omitted due to high differences between mined and predicted ages, which prevented imputation using epigenetic age as for other tissues (below). We summarized study characteristics, including phenotype or disease of interest (Supplemental Materials). Targeted samples were from a variety of studies targeting various diseases, syndromes, disorders, and exposures. Patient demographics spanned all life stages, including fetal, infant, child, and young and old adult, and several studies focused on ethnic groups not commonly studied (e.g. Gambian children from GSE100563;GSE100561).

2.2.10 Preprocessing and analyzing seven non-cancer tissues for DNAm variability analyses

We studied samples in seven tissue types, including adipose, blood, brain, buccal, nasal, liver, and sperm (Supplemental Materials, Fig. B.7a and Fig. B.7b). We removed likely low-quality samples that showed low study-specific ($<$ 5th quantile) methylated and unmethylated signal, or showed signal below manufacturer-prescribed quality thresholds for at least one BeadArray control. We also removed putative replicates according to genotype-based identity predictions from `ewastools` [206].

We preprocessed noob-normalized DNAm for each tissue separately. First, we performed linear model adjustment on study IDs using DNAm M-values, defined as $logit(\beta)$, with the `limma` v3.39.12 package. We then converted the adjusted DNAm to Beta-value scale. To account for the impact of confounding variables, we removed probes whose DNAm variances showed significant (p-adjusted < 0.01) and substantial (percent variance $\geq 10\%$) contributions from model-based predictions of age, sex, and cell type fractions, which removed 39,000 to 194,000 (8% to 40% of) probes across tissues (ANOVAs, Fig. B.7c).

After preprocessing, we identified probes with recurrent low variance and low mean intervals (max - min, mean tissue-wise DNAm, $<$0.01 or 1%) across seven distinct tissues. We also identified probes with high and tissue-specific variance. For each analysis we used a two-step probe selection process in each tissue where we selected (i) probes in the highest or lowest 10th quantile of variance (e.g. an absolute quantile variance filter), and (ii) probes in the highest or lowest 10th quantile variance across mean DNAm bins (e.g. a binned quantile variance filter, 10 bins of magnitude 0.1 or 10% DNAm, Fig. B.7a). The `recountmethylation` Data Analyses vignette reproduces these analyses for two tissues, and the full analysis scripts are contained at `https://github.com/metamaden/recountmethylationManuscriptSupplement/tree/main/inst/scripts/analyses`.

2.2.11 Statistical analyses and visualizations

Statistical analyses and visualizations were conducted with the R and Python programming languages. We used the `numpy` v1.15.1, `scipy` v1.1.0, and `pandas` v0.23.0 Python packages to manage jobs and downloads, perform data extraction, and calculate summary statistics. We used the `minfi` v1.29.3 and `limma` v3.39.12 R/Bioconductor packages for downstream quality control, preprocessing, and analyses. Plots were generated using base R functions, `ggplot2` v3.1.0, and `ComplexHeatmap` v1.99.5 [210, 211]. To reproduce analyses, see Supplemental Methods online, files at `https://github.com/metamaden/recountmethylationManuscriptSupplement`, and the Data Analyses vignette in the `recountmethylation` [38] R/Bioconductor package.

2.2.12 Supplemental Information

Supplemental Information, including methods, code, scripts, and data files are accessible at `https://github.com/metamaden/recountmethylationManuscriptSupplement`. Large supplemental data files are accessible at `https://recount.bio/data/recountmethylation_manuscript_supplement/`.

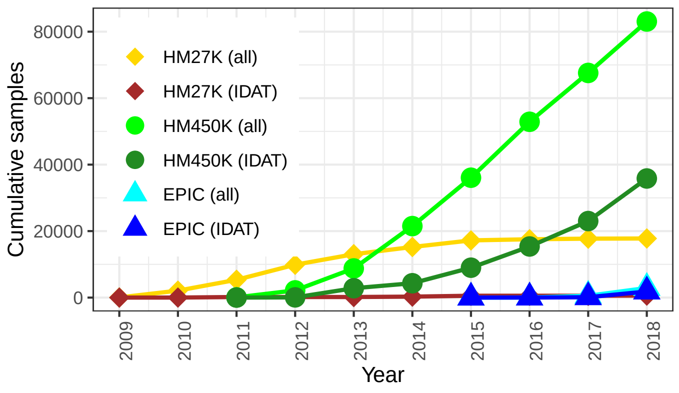
2.2.13 Companion R/Bioconductor package

Databases of the samples compiled and analyzed in this manuscript are accessible, along with comprehensive instructions and analysis examples, in the `recountmethylation` [38] R/Bioconductor package at `http://bioconductor.org/packages/devel/bioc/html/recountmethylation.html`.

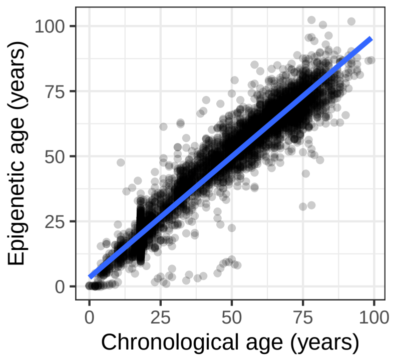
2.3 Results

2.3.1 Recent growth in GEO DNAm array samples is linear

We obtained sample IDATs and metadata for studies from the GEO. GEO is the largest public database for human DNAm array studies, and the majority of GEO's DNAm array samples use one of three of Illumina's BeadArray platforms: the HM27K, the HM450K, and HM850K or "EPIC". On GEO, we identified 104,746 unique sample accession numbers (a.k.a. GSM IDs) from 1,605 study accession numbers (a.k.a. GSE IDs) published using one of the three major Illumina DNAm array platforms (Fig. 2.1a and B.1). Among 1,605 published studies, 74% used HM450K, 21% used HM27K, and 5% used EPIC. Among 104,746 published samples, 79% were on HM450K, 18% on HM27K, and 3% on EPIC. All three platforms showed increasing publication rates of samples and studies over the first three



(a)



(b)

Figure 2.1: Cross-study summaries of DNAm array samples from GEO. (a) Cumulative samples by year using one of three major Illumina BeadArray DNAm array platforms (HM27K, HM450K, and EPIC, point shapes), showing either all samples or subsets with available IDAT files for each platform (line colors). Samples with IDATs using the HM450K platform (dark green line, circle shape) were compiled and analyzed (Section 2.2 and 2.3). (b) Scatter plot of mined chronological (x-axis) and epigenetic (y-axis) ages, in years, with linear model fit (blue line), for 6,019 non-cancer tissues run using the HM450K platform (Section 2.3). Chronological age was mined from sample metadata. Epigenetic age was calculated using the model in [30] (2.2.7).

years of their availability. Few new studies and samples from 2013-2018 used the HM27K platform, while samples and studies using HM450K have grown linearly through 2018.

2.3.2 Fewer than half of DNAm array studies on GEO include raw data

Raw data for a DNAm array sample is comprised of two IDAT files, one for each of the red and green color channels. Accessible raw data is important for uniform normalization of samples across studies, yet not all samples on GEO come with these data. In total, 37,919 samples (36% of total) included sample IDATs, where 93% were run on HM450K, 5% on EPIC, and 2% on HM27K. By platform, EPIC included the largest percentage of sample records with available IDATs at 63%, followed by HM450K at 43%, and HM27K at just 3%. The more frequent availability of IDATs for newer arrays seems to reflect a significant shift in data submission norms well after the inception of the HM27K platform.

2.3.3 Most annotated GEO HM450K samples with available raw data are from blood or brain

There were enough study and sample metadata for us to annotate 27,027 samples, 76% of the 35,360 we analyzed. We annotated these samples by applying regular expressions to the mined metadata. Our vocabulary for annotations was composed of 72 distinct terms (2.2.5) strongly inspired by those used in the methylation array resource `marmal-aid` [28]. Tissue terms for blood accounted for the majority (18,212 samples, 67% of total), followed by brain (6,690 samples, 25% of total), tumor (1,977 samples, 7% of total), breast (1,525 samples, 6% of total), and placenta (1,338 samples, 5% of total). We further annotated disease and experiment group for 22,790 samples (64% of total) using 38 distinct disease- and group-related terms. Among these, disease terms for cancer were assigned to over half (13,131, 58% of total) of samples, while terms for normal, control, or healthy were assigned to 10,808 samples (47% of total). The most frequently annotated cancers included leukemia (2,585 samples, 20% of total), breast cancer (511 samples, 4% of total), colorectal cancer (314 samples, 2% of total), and prostate cancer (196 samples, 1% of total). We compared disease and tissue characteristics to distinguish between tumor and normal samples from cancer patients, estimating that a third of samples were from tumor (2.2.9, Supplemental Materials).

2.3.4 Chronological age is accurately predicted from epigenetic age in non-cancer tissues

Prior work showed chronological age can be predicted with high accuracy from DNAm among non-cancer tissues [30, 113]. We calculated model-based age predictions (a.k.a. "epigenetic ages") from IDATs for 35,360 samples using the clock from [30], and we were able to mine chronological ages from metadata (a.k.a "chronological ages") for a subset of 16,510 samples (47% of total, Table 6.1, Section 2.2). We investigated variance sources and differences between these ages, and determined whether missing chronological ages could be imputed using the epigenetic ages for certain types of samples.

In the 16,510 samples for which we were able to mine chronological ages from metadata, ANOVA showed most epigenetic age variation was explained by chronological age (52% of variances, $p < 2.2e\text{-}16$), followed by study (i.e. GSE ID; 24%, $p < 2.2e\text{-}16$), cancer status (7e-2%, p = 1.3e-9), and predicted sample type (8e-3%, p = 1.6e-2). Compared to variances attributed to the study variable, the relative low variances attributed to the cancer status and predicted sample type variables may be due to high study-specific variance in metadata completeness or availability. High age differences (12.9 years mean absolute difference, or MAD) and errors (R-squared = 0.76) likely resulted from either metadata inaccuracies, age label misattributions from our mining strategy, or inclusion of cancers and non-tissue samples (e.g. cell lines, 2.2). In the subset of 6,019 likely non-cancer tissue samples across 37 study records with study-wise MADs $\leq$ 10 years, epigenetic age variance contribution from mined age increased to 93% (ANOVA, P-value $< 2.2e\text{-}16$) and contribution from study decreased to 2% (P-value $< 2.2e\text{-}16$, Figure 2.1b). Unsurprisingly, the non-cancer tissue samples showed lower age differences (MAD = 4.5 years) and errors (R-squared = 0.94), and ages were highly correlated (Spearman Rho = 0.96, P-value $< 2.2e\text{-}16$), supporting the well-established finding that chronological age is accurately predicted from epigenetic age in non-cancer tissues [30, 113]. We therefore imputed missing chronological ages using the epigenetic age for non-cancer tissue analyses below.

We next studied age acceleration [30, 113] by probing the differences between epigenetic and chronological ages among the 6,019 previously identified samples with low study-wise age differences. Among the 68 samples with outlying positive age acceleration ($\geq$ 15 years), the most frequently represented study accounted for 18 adipose samples from severely obese patients (accession ID: GSE61454 [212]). We observed 86 negative age acceleration outliers ($\leq$ -15 years), including 14 saliva samples from control subjects in a study of Parkinson's disease (GSE111223 [213]), and 19 whole blood samples from patients with genetic syndromes (GSE97362 [214]). In the latter study, we suspect reported ages are inaccurate and older than actual ages (private correspondence, investigation ongoing).

2.3.5 Almost a fifth of samples fail at least one of 17 BeadArray quality control assessments

Illumina prescribes 17 quality assessments for its 450K array, each measuring the performance of a different step in a methylation assay such as extension or hybridization [74, 208]. A given assessment comprises a quality metric and a minimum threshold value below which the assessment is failed. We call these assessments BeadArray controls. We used the 17 BeadArray controls and their minimum quality thresholds to evaluate assay qualities in 35,360 samples (Supplemental Materials). Results are summarized in Fig. 2.2a. The highest proportions of samples failed the non-polymorphic green and biotin staining green controls, with about 6.7% failing each (2,381 and 2,368 samples, respectively). By contrast, there are six BeadArray controls, each failed by fewer than 100 samples. A substantial number of samples (6,813, 19% of total) failed at least one control. Of samples that failed at least one control, 4,456 samples (66%) failed exactly one control, while 2,357 samples (34%) failed more than one control. Of samples that failed more than one control, 634 failed both biotin staining controls and 648 failed both non-polymorphic controls. Samples failing at least one control were significantly enriched for certain labels including "cord blood," "brain cancer", "prostate cancer", "arthritis", and "obese" (binomial test, BH-adjusted P-value $<$ 1e-3).

2.3.6 The intrinsic dimension of the 17 BeadArray controls is small

We studied signals and outcomes to determine how best to use the BeadArray controls for sample quality assessments. Cross-sample signal distributions for five BeadArray controls were bimodal, with distinct low- and high-signal modes; minimum quality thresholds fell near low-signal modes (Fig. B.3). For these controls, modifying minimum thresholds to more robustly capture low-signal samples could improve their utility. PCA of sample control performances showed the top five components explained 84% of overall variances. Component-wise ANOVAs showed that just five of the 17 controls explained the majority of sum of squared variances across these top five components (minimum = 67%, maximum = 99%, median = 97%). This suggests that the intrinsic dimension of sample quality is around 5. We conclude that sample quality is adequately captured by the performance of only 5 of the Illumina control probes (both biotin staining controls, both non-polymorphic controls and bisulfite conversion I red, Figure B.4).

2.3.7 FFPE samples fail at least one BeadArray control almost twice as often as fresh frozen samples

We investigated the impact of storage conditions on sample quality across 28 studies by comparing 3,467 FFPE and 5,729 FF samples (Table 6.4 and Figure 2.2b). FFPE samples showed greater variance than FF samples in both methylated (0.36 for FFPE vs. 0.27 for FF) and unmethylated (0.50 for FFPE vs. 0.21 for FF) signal channels. The trend could be driven either by condition-related sample characteristics (e.g. increased DNA deamination and/or lower DNA yield in FFPE, etc.) or differing preparation protocols (e.g., addition of the DNA restoration step for FFPE, [215–217]). Enriched labels also varied by storage condition among low-signal samples (binomial test, BH-adjusted p $<$ 1e-3), where "colorectal," "intestine," and "mucosa" were enriched among FFPE, while "nasal," "pancreas," and "epithelial" were enriched among FF samples.

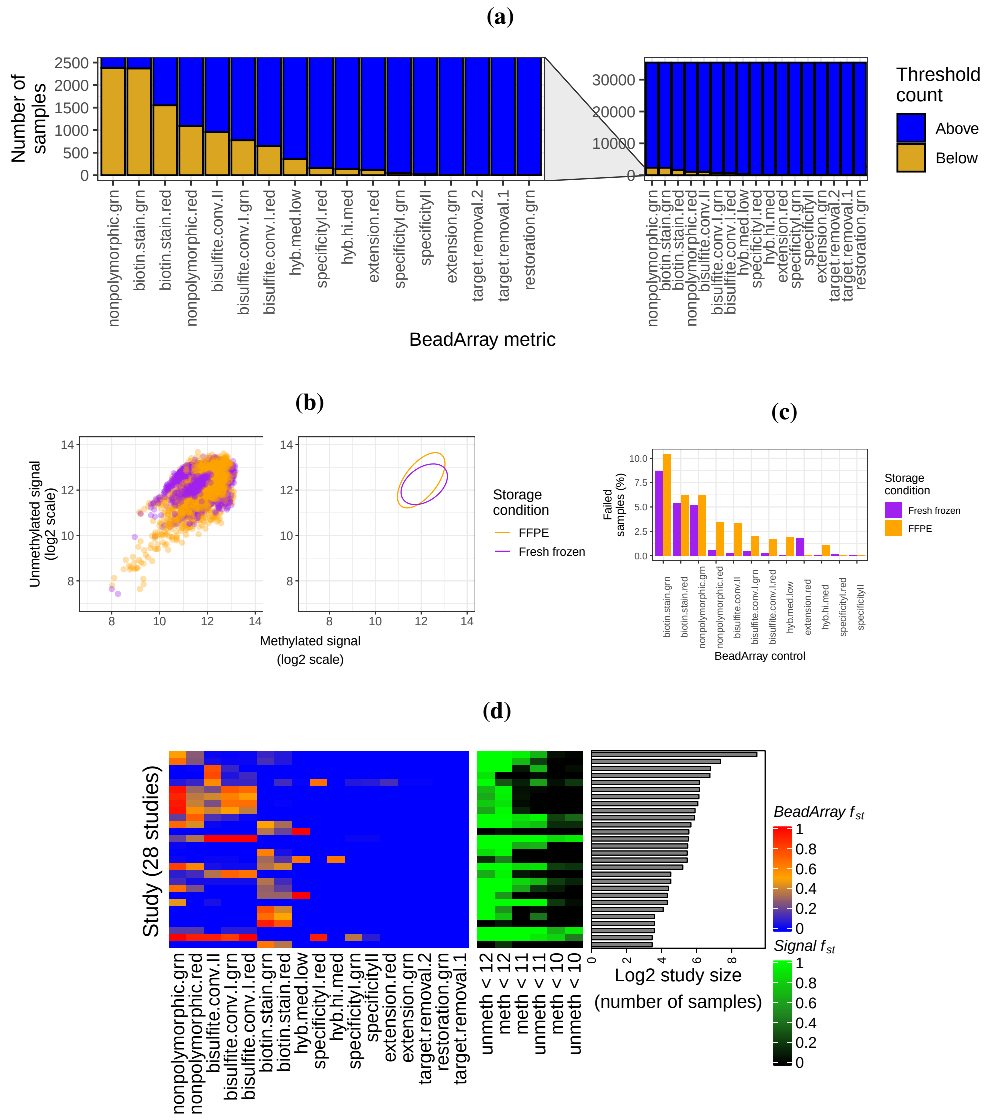


Figure 2.2: Quality analyses across samples, storage conditions, and studies. (a) Barplots counting samples (y-axis) falling above(blue) or below (gold) manufacturer-prescribed thresholds across the 17 BeadArray controls (x-axis). Full view is on right, and magnification is on left. (b) Scatter plots (left) and 95% confidence intervals (right) for log_2 median methylated (x-axis) and log_2 median unmethylated (y-axis) signal of 3,467 FFPE (orange) and 5,729 fresh frozen samples (purple). (c) Percentages of FFPE (orange) and fresh frozen (purple) samples failing BeadArray controls. (d) Heatmaps depicting fraction (f_{st} in legends) of samples in a study failing quality assessments across 28 studies with high failure rates ($f_{st} > 60\%$) and >10 samples. *BeadArray f_{st}* values are shown on the left, where blue is low, orange is intermediate, and red is high. *Signal f_{st} values* for three methylated (M, "meth") and unmethylated (U, "unmeth") signal levels (10, 11, and 12) are shown in the middle, where black is low, dark green intermediate, and light green is high. The $\log_2$ study sizes are shown on the right.

Across the 12 of 17 total BeadArray controls each with at least one failing sample, 228 FFPE samples (8.31% of total FFPE sample count) and 241 FF samples (4.21% of total FF sample count) failed at least one control. These failure rates are much lower than the failure rate across all samples (19%) and point to a correlation between study metadata completeness (e.g., inclusion of storage procedure details) and sample quality. FFPE samples failed 10 of the 12 metrics between 0.1-3.2% more often than FF, including all three bisulfite conversion metrics (Figure 2.2c). FF samples had higher failure rates for two BeadArray controls (extension red and specificity I red; Table 6.4). While no samples failed the restoration BeadArray control, increasing the minimum threshold for failure from the default manufacturer-prescribed value of 0 to 1, which is recommended as an alternative in Illumina documentation [74, 208], failed 69 FFPE samples (2%) and one FF sample (2e-3%). In summary, while FFPE samples were of lower quality than FF samples across assessments, the differences were modest, and the vast majority of FFPE samples passed all controls considered.

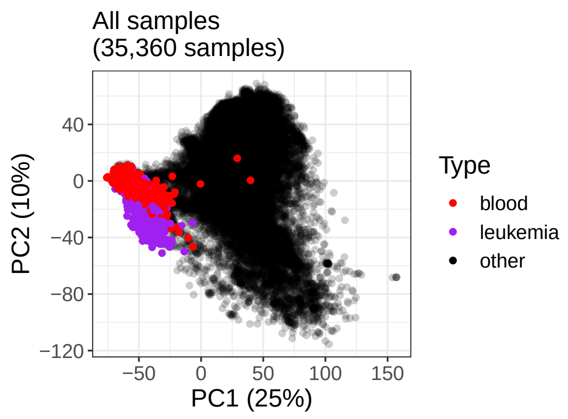
2.3.8 10% of studies each have greater than 60% samples failing quality assessments

Across 362 studies, we evaluated the fraction f_{st} of failed samples per study, defining a failed sample as one that either (i) fails at least one BeadArray control, or (ii) has $\log_2$ median methylated and $\log_2$ median unmethylated signals each $<$11 as described in [27] (Supplemental Materials, Fig. B.5). Of the 36 studies each with $f_{st} > 60\%$, samples fail in each of 23 studies due only to (i), samples fail in each of five studies due only to (ii), and samples fail in each of the remaining eight studies due to either (i) or (ii). These 36 studies ranged in size from 6 to 692 samples and comprised a total of 2,020 samples, with a median study size of 23 samples. Of the 320 studies that remained after removing those with $\leq$10 samples, 28 showed $f_{st} > 60\%$ (8.8% of remaining studies, Fig. 2.2d). One of these was a study of condition-specific DNAm data reliability (GSE59038, [216]) and included several stress tests of the assay, so many failed samples are not unexpected. Another study was GSE62219 [218] and included blood from 10 young individuals. We further noted the previous study [219] also determined these samples were of low quality.

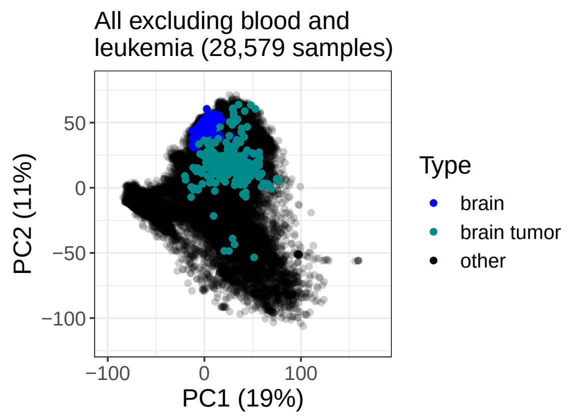
2.3.9 DNAm principal component analysis shows clustering by tissue with greater variances among cancers

We studied DNAm variance using PCAs of autosomal DNAm (Fig. 2.3) as measured by noob-normalized Beta-values (Section 2.2). The first two components from PCA of 35,360 samples explained 35% of total variance, with PC1's contribution 25% and PC2's contribution 10% (Fig. 2.3a). Four outlying blood samples (PC1 $>$ -10) included two from whole blood, one of T-cells, and one stem cell sample from umbilical cord blood (left plot of Fig. B.6a). For the top two components, leukemia samples showed greater variances than blood samples: the ratio of variance in PC1 for leukemia samples to variance in PC1 for blood samples was 1.25 (F-test P-value $<$ 1e-2), and the ratio of variance in PC2 for leukemia samples to variance in PC2 in blood samples was 6.18 (F-test P-value $<$ 1e-2). This is consistent with how (1) leukemia samples have greater variance than blood samples at each of the majority of probes (311,127, or 66%), and (2) leukemia samples have greater median variance than blood samples across probes (median Beta-value variance for blood samples = 1e-3, median Beta-value variance for leukemia samples = 5e-3; Fig. B.6b).

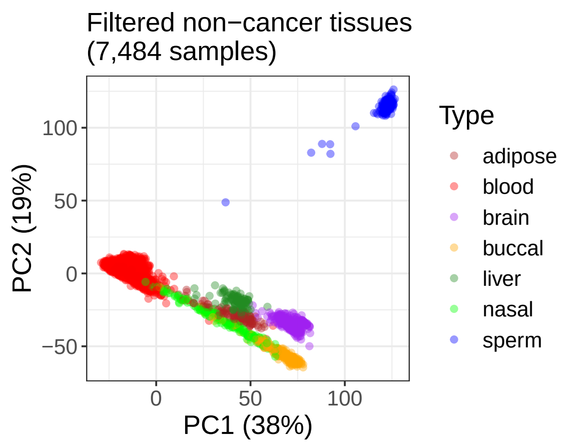
(a)



(b)



(c)



(d)

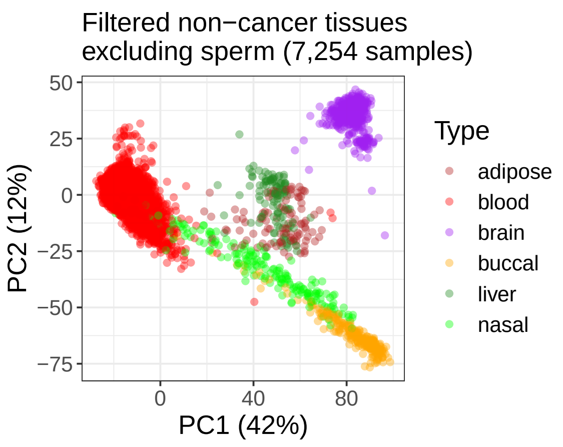


Figure 2.3: Scatter plots of top 2 components from PCA of autosomal DNAm (2.2.8). Each axis label also contains percent of total variance explained by the component. (a) PCA of 35,360 samples, with color labels for non-cancer blood ($N = 6,001$ samples, red points) and leukemias (780, purple), and remaining samples (28,579, black). (b) PCA of 28,579 samples remaining after exclusion of blood and leukemias from (a), highlighting non-cancer brain ($N = 602$ samples, blue), brain tumors (221, dark cyan), and remaining samples (27,756, black points). Facet plots of sample subsets in (a) and (b) are shown in Fig. B.6. (c) and (d) display samples from seven non-cancer tissues for which at least 100 samples were available from at least two studies (Section 2.2). (c) PCA of 7,484 samples from all seven tissue types, including sperm ($N = 230$ samples, blue), adipose (104, dark red), blood (6,001, red), brain (602, purple), buccal (244, orange), nasal (191, light green), and liver (112, dark green). (d) PCA of 7,254 non-cancer tissue samples remaining from (c) after exclusion of sperm, with color labels as in (c).

From PCA of the 28,579 samples remaining after blood and leukemia samples were removed (Fig. 2.3b), the first two components explained 30% of total variance, with PC1's contribution 19% and PC2's contribution 11%. Seven outlying (PC1 $>$ 0, PC2 $<$ -5) brain tumor samples included two primary tumors and one metastasis each from medulloblastoma cases, as well as four brain metastases from uncertain primary tumors, from the studies GSE108576 [220, 221] and GSE63669 [222] (Figure B.6c). For the top two components, brain tumors showed greater variances than non-cancer brain samples: the ratio of variance in PC1 for brain tumors to variance in PC1 for non-cancer brain samples was 12.05 (F-test p $<$ 1e-5), and the ratio of variance in PC2 for brain tumors to the variance in PC2 for non-cancer brain samples was 22.40 (F-test P-value $<$ 1e-5). This is consistent with how (1) brain tumors have greater variance than non-cancer brain samples at each of the majority of probes (444,304, or 94%), and (2) brain tumors have greater median variance than non-cancer brain samples across probes (median Beta-value variance for non-cancer brain samples = 1e-3, median Beta-value variance for brain tumors = 1e-2; Fig. B.6d). Our findings are consistent with previous evidence of higher DNAm variances in cancers compared to non-cancer samples [223, 224].

PCA of 7,484 samples from seven non-cancer tissues (adipose, nasal, blood, brain, buccal, sperm, and liver), which we also used to study DNAm variability (below), showed clear clustering by tissue. The first two components explained 57% of total variance, with PC1's contribution 38% and PC2's contribution 19%. Sperm samples clustered far apart from the six somatic tissues (Fig. 2.3c). After repeating PCA with sperm samples excluded, the first two components still explained over half (54%) of total variance, with PC1's contribution 42% and PC2's contribution 12% (Fig. 2.3d).

2.3.10 Over two-thirds of CpG probes that do not distinguish tissues map to gene promoters near CpG islands

CpG probes with low DNAm variation and low mean DNAm differences across experimental groups are less informative for quantifying group-specific DNAm differences. We analyzed autosomal DNAm variation in seven distinct tissues (adipose, nasal, blood, brain, buccal, sperm, and liver), as measured by noob-normalized, study-corrected Beta-values (see 2.2.10). We identified 4,577 probes each with consistently low variance ($\leq$10th quantile) in each tissue and low difference between highest and lowest mean Beta-value ($<$0.01) across tissues (Fig. B.7a and B.8, and Supplemental Materials). Among probes with consistently low variance, 4,111 (90% of total) mapped to genes in CpG islands, typically at promoter regions of CpG island-overlapping genes (2,203 probes), and these fractions represented significant increases compared to the background of all autosomal CpG probes (binomial tests, P-values $<$ 1e-3). It is likely the 4,577 probes are of low utility for quantifying DNAm differences across tissues, and their removal prior to performing an EWAS across non-cancer tissues could help increase statistical power.

2.3.11 Over two-thirds of CpG probes that distinguish tissues map to genes

We identified 2,000 CpG probes in each of seven distinct non-cancer tissues (adipose, nasal, blood, brain, buccal, sperm, and liver) with high and tissue-specific variation in autosomal DNAm, as measured by noob-normalized, study-corrected Beta-values (Section 2.2, Supplemental Materials, and Fig. 2.4 and B.7a). Distinctive patterns in DNAm across these probe sets may point to tissue-specific factors such as differences in environment exposure, cellular signaling, and

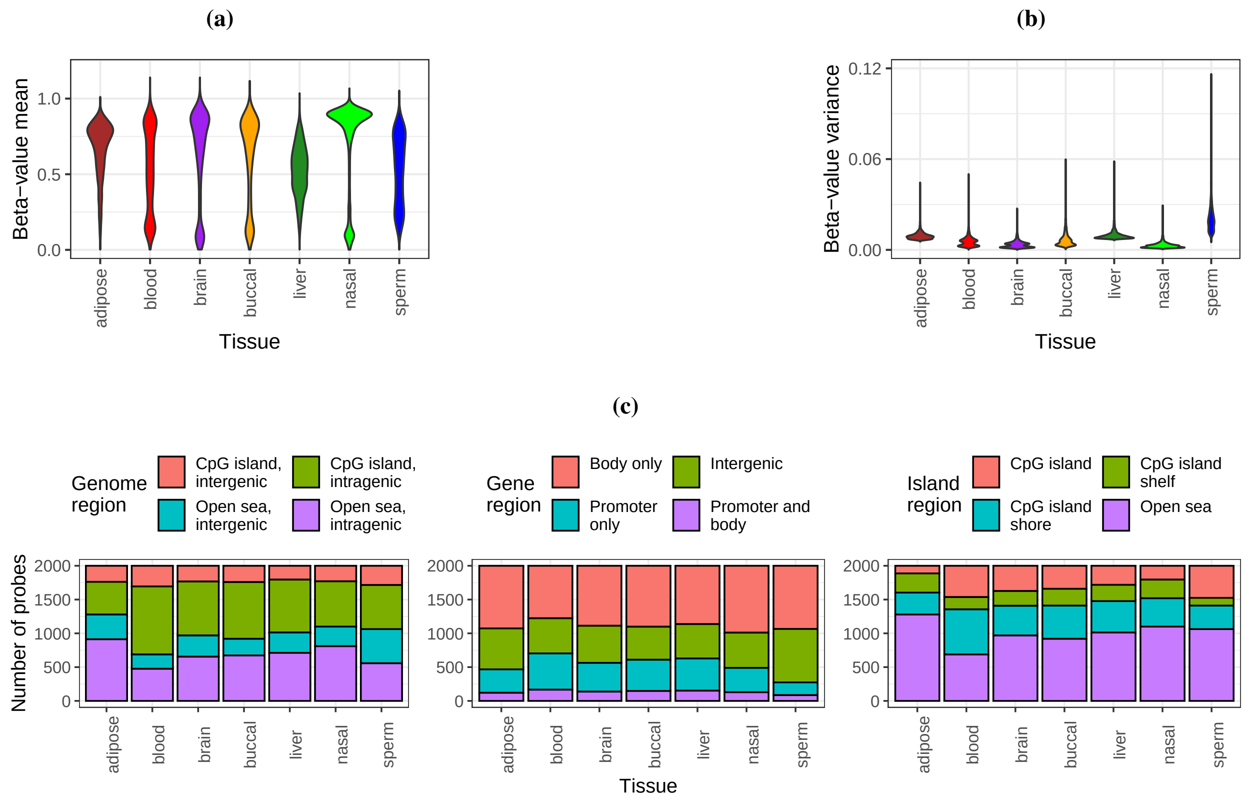


Figure 2.4: DNAm and genome mapping patterns among 14,000 CpG probes showing tissue-specific high variance in 7 tissues (2,000 probes per tissue, tissues: adipose, blood, brain, buccal, liver, nasal, and sperm). (a) and (b) Violin plots of (a) means and (b) variances of normalized Beta-values across tissue-specific probes. (c) Stacked barplots of genome region mappings (Number of CpG probes, y-axis) across tissue-specific probes (x-axis). Color fills depict (left) island and gene overlap, (center) gene region overlap, and (right) CpG island region overlap.

cell division rates. Compared to the background of all autosomal CpG probes, adipose and sperm had significantly lower fractions of gene-mapping probes, and all tissues except for blood had significantly greater fractions of both open sea-mapping probes and gene body-mapping probes (binomial tests, BH-adjusted P-value $<$ 1e-3). Of the 14,000 total high-variance probes, 10,016 (71%) mapped to a gene region, typically at the gene body (8,006 probes, Supplemental Materials). The highest mean Beta-values were observed for nasal and adipose tissues (Fig. 2.4a), and the highest variances were observed for sperm and adipose tissues (Fig. 2.4b), while probes in blood had relative low means and variances. While most probes mapped to open seas in liver (1,014 probes), nasal (1,100), adipose (1,280), and sperm (1,063), greater fractions of open sea probes mapped to genes in liver (70% of open sea probes), nasal (74%), and adipose (71%) than in sperm (52%, Fig. 2.4c). This observed sparsity of CpG island regions and enrichment of open sea, intergenic, and gene body regions among CpG probes with tissue-specific DNAm was recently corroborated in an independent study comparing DNAm in matched sperm and blood samples directly [48]. This corroboration was especially striking because the discovery set samples in [48] were processed on the newer EPIC rather than the HM450K platform.

2.3.12 Normalized Beta-values for GEO DNAm array studies are rapidly accessed via the `recountmethylation` package

To accommodate a wide range of analysis strategies, DNAm assays and sample metadata were compiled into databases in two distinct formats, including HDF5 and `HDF5-SummarizedExperiment`. `HDF5-SummarizedExperiment` compilations are tailored for rapidly executing data summaries and query operations in the R/Bioconductor framework via `DelayedArray` objects. Raw red and green signals are provided as HDF5 (120 GB) and `HDF5-SummarizedExperiment` (119 GB) files, and raw methylated and unmethylated signals and noob-normalized Beta-values are provided as `HDF5-SummarizedExperiment` files (94 GB and 133 GB, respectively). The `recountmethylation` [38] R/Bioconductor package facilitates database access as described in the User's Manual. It allows full database utilization with rapid queries on the provided sample metadata, including model-based estimates for sex, epigenetic age, and blood cell types [27, 30, 31]. The package Data Analyses vignette further provides code to reproduce our comparisons of mined and epigenetic ages, sample storage type quality comparisons, and tissue-specific DNAm variability analyses described above.

2.4 Discussion

2.4.1 Limitations of this study

We conducted a cross-study analysis of methylation array samples comprising a large subset of available HM450K samples on GEO. While we omitted studies using the HM27K and EPIC platforms, our compilation strategy could also be generalized to these platforms (Section 2.2, below). Further, while our results suggest BeadArray controls could be improved by applying different quantitative thresholds for failure, it remains unclear whether a single universal threshold or multiple experiment-specific thresholds is desirable for each (2.3.5, Fig. B.3). Nonetheless, five of 17 BeadArray controls (both Biotin Staining controls, both Non-polymorphic controls, and Bisulfite Conversion I Red) are demonstrably useful for assessing the quality of an experiment. This finding also means a stringent quality threshold of ≥ 1 failed controls, which we used for cross-study analyses, mainly filtered samples due to failure in at least one of these five principal controls. We further lacked a definitive gold standard set of well-described DNAm array samples, which could allow for more detailed estimations of metadata errors beyond direct concordances, or for more informative assessments of quality metric behaviors. Our metadata mapping and annotation strategy can also be improved to better capture available metadata for certain samples, such as gestational and maternal ages for placenta samples (Section 2.2). Finally, our DNAm variability study across seven distinct non-cancer tissues used a within-tissue preprocessing approach (Section 2.2). We were constrained this way because study-specific variation was high relative to tissue-specific variation, and effective study-and-tissue normalization would have required considerably more data from studies of multiple tissues than were available at the time of analysis.

2.4.2 Recommendations for metadata reporting

Thoroughly characterizing samples submitted to public archives with accurate metadata makes them easier to repurpose for new studies. After manually inspecting hundreds of DNAm array studies on GEO, we formulated some best practices for the submitter who is labeling samples in a study to facilitate their discoverability and improve their utility for other investigators:

1. Include key attributes (sex, age, tissue, disease, etc.) even when any one is the same across a sample set, since that attribute may vary in a cross-study analysis.

2. Repeat study-level metadata in sample-level metadata. This includes sample types (e.g., tissue, cell line, etc.) and characteristics (e.g., storage conditions, preparation steps, etc.).

3. Include units of numerical variables to ensure their proper interpretation.

4. Clarify circumstances under which attributes were obtained where appropriate. Examples: age *at diagnosis*, *tumor-adjacent* normal tissue, blood *from leukemia patient*.

2.4.3 Next steps

We have several methodological changes planned that will improve the DNAm array databases accessible with the `recountmethylation` [38] R/Bioconductor package. First, future compilations will add samples run on the newer EPIC platform, allowing for novel cross-platform analyses. Further, our metadata handling pipeline will be revised to be fully automated by using regular expressions to recognize key metadata. This will replace the manual variable aggregation step (2.2). Finally, we will support regular compilation updates by enabling rapid setup and better dependency handling (e.g., with virtual environments). These improvements will empower the researcher to maintain a comprehensive compilation of DNAm array IDATs from GEO in a wide array of computing environments.

2.5 Conclusion

We performed extensive analyses of 35,360 HM450K samples with IDATs from 362 studies in GEO, approximately three times the number of samples considered in prior cross-study analyses [28, 30, 48, 206]. We further released the R/Bioconductor package `recountmethylation` including our new tissue and disease state labels, model-based predictions for age, sex, and blood cell composition, as well as noob-normalized Beta-values for array samples. This resource should prove valuable for reusing publicly available methylation data.

2.6 Acknowledgements and funding

This research was funded in part by NIH 5R01GM121459-02 to KH. KH was further supported by NIH U01AG060908. We are especially grateful to John Greally for supportive comments and helpful suggestions as this work progressed.

We also thank Julianne David, Mary Wood, Ben Weeder, Austin Nguyen, and Chris Loo for early feedback on our manuscript.

2.7 Copywrite statement

3 Chapter 2: `recountmethylation` enables flexible analysis of public blood DNA methylation array data

3.1 Background

DNA methylation (DNAm) is the most commonly studied epigenetic mark, and most public DNAm array samples are generated from blood [3]. In prior work [3], we conducted comprehensive cross-study analyses of human DNAm array studies with raw data deposited on the Gene Expression Omnibus (GEO) [8, 9], the largest archive of publicly available array data. We confined attention to the HumanMethylation450K (HM450K) platform introduced by Illumina in 2012. HM450K arrays profile 485,577 CpG loci concentrated in protein-coding genes and CpG island regions [45, 46]. We found that: (1) a subset of Illumina's prescribed BeadArray controls explained most quality variances; (2) samples clustered by tissue and cancer status in a principal component analysis (PCA) of autosomal DNAm; and (3) subsets of CpG probes showed high tissue-specific DNAm variation among 7 normal tissues. We further released the `recountmethylation` Bioconductor package [39] along with uniformly processed data compilations pairing DNAm with harmonized metadata labels for age, sex, tissue, and disease state.

The initial `recountmethylation` release left open several important issues. First, the delay between data compilation and reporting meant our initial release of data compiled in March 2019 was over a year out of date at the time of publication. Second, the prevalence of raw data from the newer EPIC platform [47] is rapidly increasing while our initial data compilation included only samples run on the older HM450K platform. Finally, several practical research concerns were not accommodated in the initial package release, including how to leverage public array data compilations to determine the required number of samples to test a new hypothesis, how to account for confounding factors in cross-study analyses, and how to leverage public data to independently validate previously published DMPs and identify subsets of high-confidence biomarker candidates.

We address these outstanding issues in the present paper using novel cross-study and cross-platform analyses, confining attention to normal human blood samples. Blood DNAm is often probed in epigenome-wide association studies (EWAS) to discover, test, and validate biomarkers [21–23] for diseases such as type II diabetes [24, 89, 90], obesity [91], non-alcoholic fatty liver disease [225], asthma [226], and dementia [227], as well as colorectal [96–99], esophageal [92], breast [93], pancreatic [95], and head-and-neck [94] cancers. It is widely used to study biological aging [30, 49, 113, 119] and normal tissue epigenetics [48, 228], including development and function of the immune system [229]. Recent work studied how gestational age-related differential DNAm relates to fetal health and disease risk [119, 230]. Further, cord blood DNAm is increasingly used to precisely quantify fetal gestational age [49, 115–117], which may lead to improvement in the efficacy of prenatal screening [118]. In addition, many software tools were trained and designed for use with blood DNAm data; these included methods for cell-type deconvolution [31, 139, 231], inference of population genetic structure and shared genetic ancestry [32], and power analyses [181].

DNAm differences between sexes have been observed in mouse [232] and multiple human tissues including brain [232], pancreas [233], nasal epithelium [234], cord blood [235], and whole blood [138, 236]. These DNAm differences can impact insulin secretion [233], risk of disease [234, 235], and biological age [232]. Using cross-study and cross-platform compilations of whole blood and PBMC, we performed novel independent validation of previously published sets of probes with differential methylation between the sexes (a.k.a. "sex DMPs") from two previous studies in whole blood [138, 236].

3.2 Methods

3.2.1 Compiling recent public DNAm array data across platforms

DNAm array data were identified, downloaded, and processed using the `recountmethylation_instance` v0.0.1 [61] [61] Snakemake workflow (Appendix A.7). It comprises the `recountmethylation_server` v1.0.0 [193] and `recountmethylation.pipeline` v1.0.0 [194] tools, which we used previously to compile HM450K data [3]. We uniformly processed samples into cross-study and cross-platform data compilations. Data were from samples run on the HM450K [45] and EPIC [47] DNAm array platforms and available on GEO by March 31, 2021. Compilations paired noob-normalized [140] DNAm fractions, or Beta-values, with harmonized sample metadata for 68,758 cumulative samples for which raw image data files, or IDATs, were available as gzip-compressed supplementary files.

Compilations were stored as HDF5-based SummarizedExperiment files generated using the `HDF5Array` v1.18.0 and `rhdf5` v2.34.0 R/Bioconductor packages [177, 178]. These formats used `DelayedArray` v0.18.0 [237] to support rapid access, summaries, and filters. For most analyses, DNAm data were merged across platforms for the 453,093 CpG probes [47] they shared. We made compiled data available online at `https://recount.bio/data/gr-gseadj_h5se_hm450k-epic-merge_0-0-3/`.

3.2.2 Prediction and harmonization of sample metadata

We generated harmonized sample metadata from heterogeneous metadata mined from SOFT files accompanying GEO studies. This involved writing regex to detect keywords in the files that we then mapped to controlled vocabularies under "tissue", "disease," and other categories, as described in [3] and [28]. We also predicted sample types from mined metadata using the method from [17]. To add sex annotations, we used the `minfi` v1.37.1 [27, 238]) R package; to add six blood type cell fractions, we used the method from [31]); and to add age annotations, we used the pan-tissue epigenetic clock model from [30]. Finally, we calculated the top components of genetic ancestry using the method from [32] (Tables C.1 and C.2).

3.2.3 Sample QC filters

We used metadata filters to find the 3 most prevalent blood sample types (whole blood, cord blood, and PBMCs), and the aggregate type "all," which includes the above types and samples of whose type was indeterminate from mined metadata. We then performed QC with reference to prior findings from [3]. We removed samples for which either:

(1) log2 median M and U signals were both <10; or (2) the sample failed $\geq 2/5$ most informative BeadArray metrics. These criteria removed 245 samples, and all but one was run on the HM450K platform.

3.2.4 Simulation of study bias adjustments

We used simulations to show the impact of study ID adjustment on explained variance. As detailed in Fig. C.1, simulations consisted of 4 steps: (1) calculate sample DNAm M-values from 500 CpG probes and 5 studies, selected randomly; (2) adjust study ID across all 5 selected studies (i.e. "adjustment 1") or subsets of 2-4 studies (i.e. "adjustment 2"); (3) perform ANOVA for 3 models; (4) get Fraction Explained Variance (FEV) for each variable across 3 models. In total, simulations used 29,028 unique CpG probes and 62 unique studies (Table C.3).

Multiple regression models accounted for sample type, platform, study ID, DNAm-based predictions for age, sex, and six cell type fractions, and two genetic ancestry components, which were determined as described above. Variables were grouped as one of biological (i.e. six blood cell type fractions), demographic (i.e. age, sex, and two genetic ancestry components), and technical (i.e. platform). Study bias adjustments were performed using the `removeBatchEffect()` function from the `limma` v3.46.0 [239] R package. Parallel sessions were deployed using the `parallel` v4.1.1 R package (Appendix A.9).

3.2.5 PCA of autosomal DNAm

We performed autosomal DNAm PCA on compiled blood samples using a reduced 1,000-dimensional representation of the normalized and bias-corrected Beta-values [209, 240] obtained via feature hashing. (See [3] and Appendix A.10 for details on this approach.) For the top 10 components, we calculated FEV from ANOVA using multiple regression models containing the 13 variables from the 3 categories described above.

3.2.6 Blood autosomal DNAm search index construction

We used the `hnswlib` v0.5.2 Python library to make a DNAm-based search index [241] (Appendix A.8). The Hierarchical Navigable Small Worlds (HNSW) algorithm implemented in `hnswlib` was among the top search index algorithms in a recent benchmark [242]. With the `mmh3` v3.0.0 and `numpy` v1.20.1 [243] Python libraries, we applied feature hashing to generate a reduced 1,000-dimensional representation of each sample [174, 209] of each blood sample's noob-normalized Beta-values. The search index files are available online at `https://recount.bio/data/sindex-hnsw_bval-gseadj-fh10k_all-blood-2-platforms.pickle` and `https://recount.bio/data/sidict-hnsw__bval-gseadj-fh10k__all-blood-2-platforms.pickle`.

3.2.7 Power analyses using `pwrEWAS`

We used the method provided in the `pwrEWAS` v1.4.0 R/Bioconductor library to perform power analyses across DNAm array platforms [181]. Parameters for these analyses included 100 total simulations varying the total samples N from

50 to 850. We targeted 500 Differentially Methylated Probes (DMPs) and assessed test group Beta-value differences δ of 0.05, 0.1, and 0.2.

3.2.8 Replication of whole blood sex DMPs

We replicated sex DMPs from [138], a study of whole blood from Japanese individuals, using independent compilations of whole blood and PBMC samples in `recountmethylation`. After filtering out sex chromosome and cross-reactive probes [137], there were 375,244 CpG probes in whole blood and 375,244 CpG probes in PBMCs. After filtering for sample quality, we used data from 5,980 whole blood samples (3,942 females and 2,924 males) and 642 PBMC samples (397 females and 230 males). Ages tended towards young adult and middle-aged for whole blood (age, mean$\pm$SD, 39$\pm$21 years) and samples from [138] (46$\pm$12 years), but were more frequently from adolescents and young adults among PBMC (25$\pm$19 years). We preprocessed DNAm M-values using Surrogate Variables Analysis (SVA) with the `sva` v3.4.0 R package [244]. We determined sex DMPs using coefficient P-values for the sex variable in multiple regressions, where regression models used biological, demographic, and technical variables described above (Table C.4).

3.2.9 Statistical analyses and visualizations

Data processing and analyses were performed using the R v4.1.0 and Python v3.7.1 programming languages [245, 246]. Statistical summaries and tests were performed using base R libraries. DNAm array processing, normalization, analysis, and prediction of sex and six blood cell type fractions was performed using the `minfi`, `minfiData`, and `minfiDataEPIC` R packages. Workflow diagrams were created using `BioRender.com`. Visualizations in Results made us of the `ggplot2` v3.3.2, `grid` v4.1.3, `gridExtra` v2.3, `UpSetR` v1.4.0, `ggpubr` v0.4.0, `ggforce` v0.3.3, and `png` v0.1-7 R packages [211, 247]. P-value adjustments used either the Bonferroni method or the Benjamini-Hotchberg method [167]. Enrichment tests used the `binom.test()` base R function with the background of 453,093 total probes overlapping both array platforms [245]. Supplemental scripts and functions recreating our results are available online at `https://www.github.com/metamaden/recountmethylation_v2_manuscript`.

3.2.10 Supplemental data, files, and code

The following resources have been provided to reproduce results, figures, and tables in this paper:

1. The updated `recountmethylation` Bioconductor package is now available (`https://doi.org/doi:10.18129/B9.bioc.recountmethylation`). It features new functions supporting analysis of large data compilations, and new vignettes showing how to perform novel power analysis, infer genetic ancestry, and more using DNAm array data.

2. Supplemental code and scripts for this paper, including support for creating and querying a search index of DNAm array samples, are available in the manuscript GitHub repository (`https://github.com/metamaden/recountmethylation_v2_manuscript`).

3. The `recountmethylation_instance` Snakemake workflow is available on GitHub [61]. This will be useful for researchers hoping to make and update new compilations of public DNAm array data from GEO.

3.3 Results

3.3.1 12,537 normal blood samples spanning 3 sample types were incorporated into `recountmethylation`

We uniformly processed raw intensity data generated on the HM450K or EPIC platforms for 68,758 samples available on GEO before March 31, 2021 (Fig. 3.1, Section 3.2, [27, 140]). We narrowed focus to 12,537 normal human blood samples from 63 studies, each of which had ≥ 10 samples after quality control. After harmonizing metadata across studies, we found these samples were predominantly of three types (Fig. 3.2a): whole blood, umbilical cord blood (a.k.a. "cord blood"), and Peripheral Blood Mononuclear Cell (PBMC). Whole blood was distinguished from PBMC by the presence of erythrocyte and granulocyte DNA, as these cell types are removed during PBMC preparation [248] (Fig. 3.2a). Each blood sample type included ≥ 245 samples from ≥ 2 studies per respective platform (Fig. 3.2b, Tables C.1 and C.2).

We subsequently updated our Bioconductor package `recountmethylation` [3] to facilitate cross-study and cross-platform analyses of the blood samples. The package's new features permit search for samples with DNAm profiles similar to a query sample [241], inference of shared genetic ancestry [32], and novel power analyses [181]. These features are explained in package vignettes. Further, a new `recountmethylation_instance` Snakemake workflow available on GitHub [61] allows users to create their own compilations of public DNAm array data on GEO [192], with the functionality to customize output data types and attributes predicted from GEO metadata. As shown below, our resources enable identification of biomarker candidates, independent validation and replication of previous research, experiment planning, and more.

3.3.2 Study ID adjustment increased variation explained by biological and demographic variables

We conducted simulations investigating the impact of bias correction by study ID, a surrogate for technical confounders [3]. Three DNAm values were modeled in multiple regressions: (1) unadjusted DNAm, (2) uniform adjustment on 5 randomly selected studies (a.k.a. "adjustment 1"), and (3) exact adjustment on 2-4 randomly selected studies (a.k.a. "adjustment 2"). Regression models 2 and 3 were compared to test whether two distinct study ID bias adjustment strategies had comparable outcomes. We determined the fraction of explained variance (FEV) for each of 13 variables from ANOVA, yielding 3 results per variable per simulation rep (Section 3.2). Total non-residual variances almost invariably decreased after applying either of the 2 study ID adjustment strategies (Fig. C.2a, median fractions of non-residual variances, adjusted over unadjusted, adj. 1 = 6.88e-1, adj. 2 = 6.84e-1). Variance reduction magnitudes were identical across adjustment strategies, with the exception of a few outlying models from adjustment 1 simulations (Fig. C.2b).

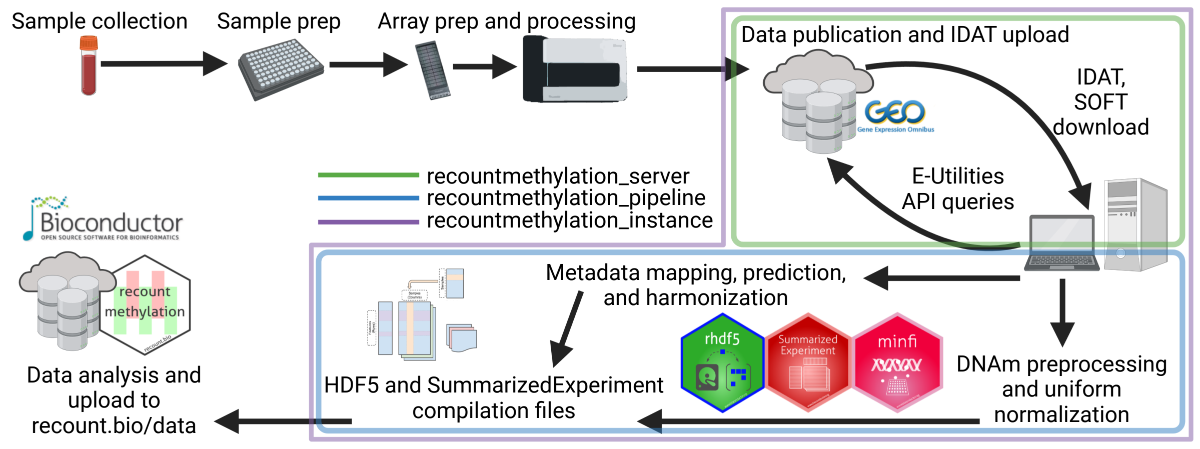

Sample collection
Sample prep
Array prep and processing
Data publication and IDAT upload
IDAT, SOFT download
E-Utilities API queries
recountmethylation_server
recountmethylation_pipeline
recountmethylation_instance
Bioconductor
recount methylation
Data analysis and upload to recount.bio/data
Metadata mapping, prediction, and harmonization
HDF5 and SummarizedExperiment compilation files
rhdf5
SummarizedExperiment
minfi
DNAm preprocessing and uniform normalization

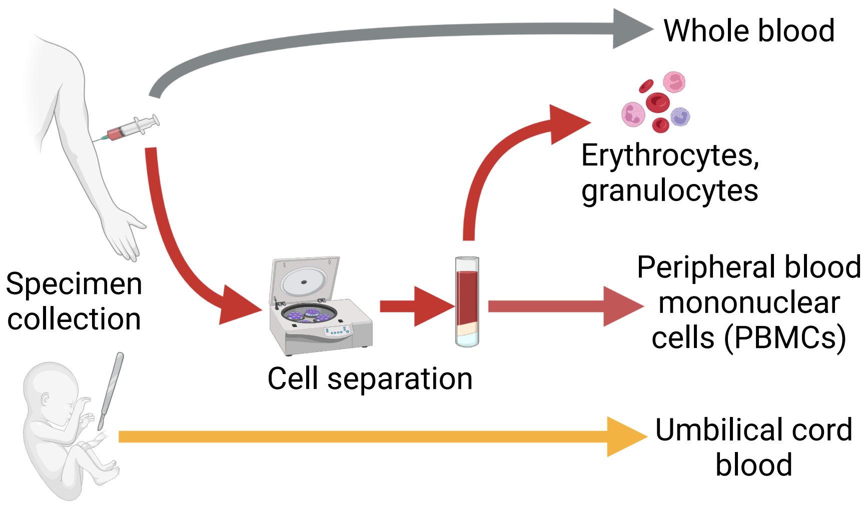
Figure 3.1: Workflow to obtain public DNAm array data from GEO. Collection, preparation, and processing of array samples (top left) as well as publication of GEO datasets were performed by other investigators (top right). We downloaded raw intensity data (IDATs) and metadata (SOFTs; top right), processed GEO metadata (middle) and DNAm signals (bottom right) into HDF5-based data formats (bottom middle), and finally updated our server and the `recountmethylation` Bioconductor package (bottom left). Color outlines indicate data access and processing using tools we developed (green = `recountmethylation_server` [193], blue = `recountmethylation_pipeline` [194], green = `recountmethylation_instance` [61]). Diagrams were created with `BioRender.com`.

We categorized variables as biological (e.g., six predicted blood cell type fractions), demographic (e.g., predicted sex, age, and genetic ancestry), or technical (e.g., platform, where applicable). Across all 3 variable categories, FEV increased relative to unadjusted models after either adjustment strategy, and FEV distributions were far more similar among adjusted models than between adjusted and unadjusted models (Fig. 3.3a). The largest median FEV differences were observed for demographic variables, while the smallest were observed for technical variables. Among individual variables, median FEV was < 0.1 across most models and variables, where study ID showed the maximum median FEV of 0.47 for unadjusted DNAm. After either adjustment, study median FEV decreased drastically to $\leq$ 2e-3, while median FEV for all remaining variables increased (Table C.3).

Because performing compilation-wide corrections on study ID substantially increased variation explained by biological and demographic variables, we included Beta-values under our adjusted models in `recountmethylation` for reuse in cross-study analyses.

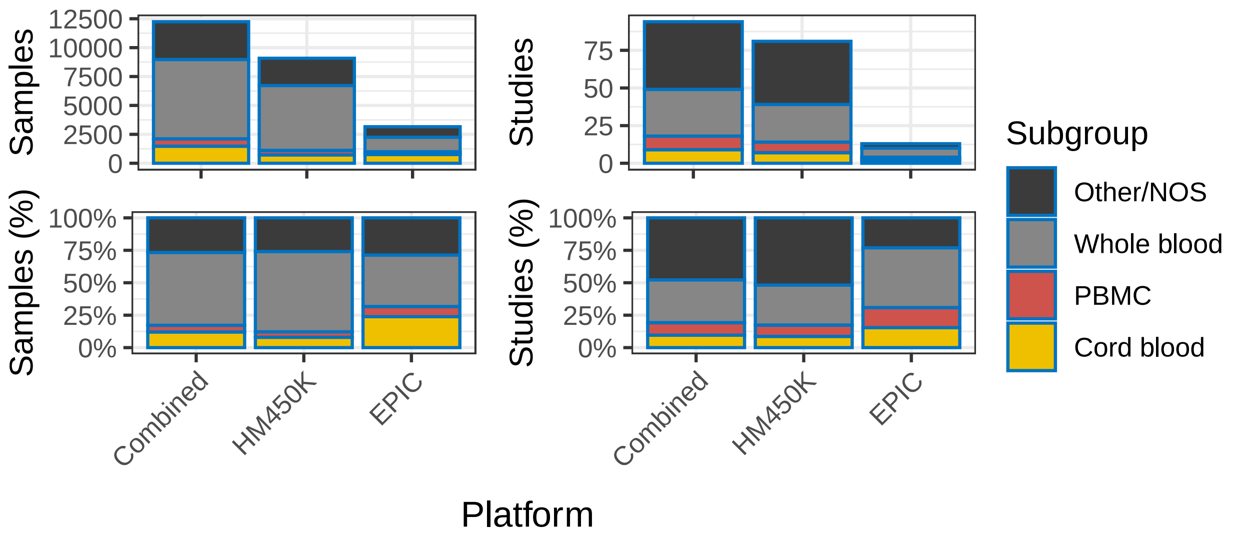
3.4 Most explained DNAm variation is from predicted genetic ancestry and predicted cell composition

To better understand key sources of variation in compiled blood data, we performed principal component analysis (PCA) on normalized [140], study ID-corrected autosomal DNAm, followed by ANOVA on regressions with 13 variables categorized as either biological, demographic, or technical (Section 3.2). While most variation was residual across most components, explained variation was mainly from demographic variables at components 1 and 3, biological variables at components 4,5,6, and 10, and from technical variables at component 8, and split between demographic and biological variables at component 2 (Fig. 3.3b). Most explained variation from demographic variables was from genetic ancestry in the first component, and while CD4+ T-cell fraction explained substantial biological across remaining top components



Whole blood
Erythrocytes, granulocytes
Specimen collection
Cell separation
Peripheral blood mononuclear cells (PBMCs)
Umbilical cord blood

(a)



(b)

Figure 3.2: Blood specimen collection and DNAm array data availability by sample type. (a) Blood sample collection and handling prior to upload to the GEO. (b) Barplot summaries of available samples (left) and studies (right), showing counts (top) and percentages (bottom) of blood sample types (black = Not Otherwise Specified [NOS], gray = whole blood, red = PBMC, yellow = cord blood). Bar heights indicate the aggregate sample group "all." Diagrams were created with `BioRender.com`.

(Fig. C.3). The top two principal components showed samples clustered largely independent from sample type and platform labels, but showed distinct gradient patterns for genetic ancestry, CD8+ T-cells, CD4+ T-cells, and B-cells (Fig. C.4).

3.5 Dependence of statistical power on sample size was similar across blood sample types

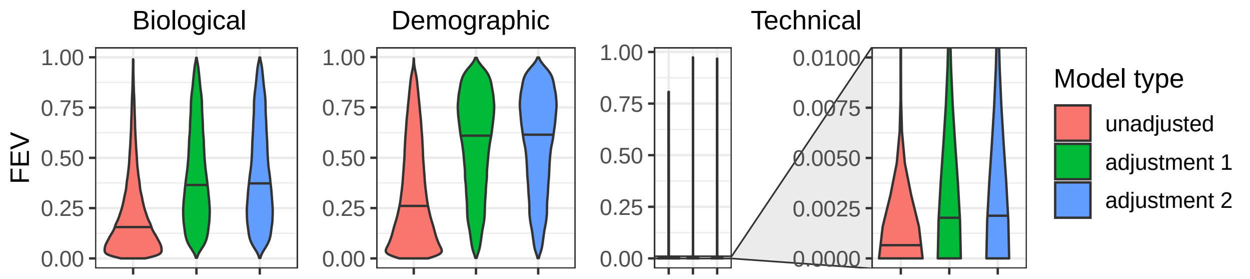
We conducted power analyses on the blood samples included in `recountmethylation` by applying the simulation-based `pwrEWAS` approach [181] (Section 3.2). To attain $\geq 80\%$ power to detect DMPs between two groups of roughly equal size, the N estimated total samples required were similar across sample types, where $N \approx 300$ samples at mean Beta-value difference between groups $\delta = 0.05$, $N \approx 150$ samples at $\delta = 0.1$, and $N \approx 80$ samples at $\delta = 0.2$. We assumed an FDR threshold of 5%. Outcomes were similar within each of the whole blood, cord blood, and PBMC groups, but they were worse when including all blood samples, likely due to greater sample heterogeneity (Fig. 3.4).

Our results suggest fewer samples are necessary than the results of [181], where adult PBMCs showed $\geq$80% power with $N = 220$ samples at $\delta = 0.1$. Further, an independent power analysis using whole-blood EPIC arrays [249] found 85% of probes had $>$80% power with $N = 200$ and $\delta = 0.1$, although their FDR cutoff value of 15% was less stringent than our cutoff value of 5%.

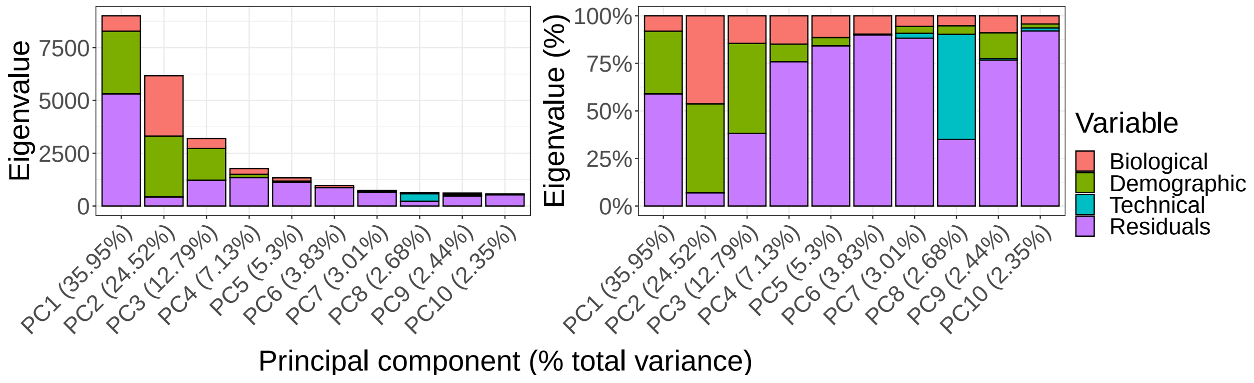
3.6 40% of sex DMPs from a previously published EWAS study were replicated in either whole blood or PBMC

We queried a search index of blood autosomal CpG DNAm, which is included in the updated `recountmethylation` resource, for each of the 113 whole blood samples from [138]. In the process, we quantified the similarity of queried sample methylation profiles to other samples by analyzing the k nearest neighbors returned (Section 3.2). Among the 1,000 nearest neighbors returned per queried sample, the whole blood label was common while the PBMC label was rare (Fig. 3.5a), in agreement with Methods in [138] describing the queried samples as "peripheral whole blood." This greater similarity to compiled whole blood may reflect greater similarity in subject ages, cell composition [248], and/or genetic ancestry (Figs C.6a and C.5b), and we corrected for these potential confounders in regressions for identifying sex DMPs from either compilation (Section 3.2).

We next considered the threshold of the top 1,000 most significant DMPs from whole blood and PBMC. We set this threshold because these DMPs captured the long tail of between-sex DNAm differences for each tissue (Fig. C.5a- C.5d), and because there was less replication divergence between tissues compared to less stringent thresholds observed from concordance at the top analysis (Fig. 3.5b). Among these whole blood and PBMC DMPs, (117/292 =) 40% replicated sex DMPs from [138] (Fig. 3.5b). Of these, 95 were only replicated in whole blood, 80 were only replicated in PBMC, and 58 were replicated in both tissues (Fig. C.6b). Further, (248/544 =) 46% of whole-blood sex DMPs independently reported in [236] overlapped DMPs in whole blood or PBMC. However, just 26 sex DMPs appeared in all of PBMC, whole blood, [138], and [236]. Mean (normalized) Beta-value was typically higher in females than males in whole blood (64-81% of DMPs) but not PBMC (35% of DMPs). There was high agreement in mean Beta-value differences between males and females, with slightly greater agreement among whole blood than PBMC (99% $>$ 96% direction agreement)



(a)



(b)

Figure 3.3: Variance analyses of study bias adjustments and principal components. (a) Distributions of FEV. Violin plots show results grouped by 3 variable categories (plot titles, one of biological on left, demographic in middle, or technical on right), and color fills show model type (pink = adjustment 1 or adjustment on 5 studies, green = adjustment 2 or adjustment on 2-4 studies, and blue = unadjusted). (b) Autosomal DNAm PCA results across normal blood samples. Stacked barplot y axes show the eigenvalue magnitudes at left and percentages at right for the top ten components on the x axes. Fill colors indicate magnitudes of component sum of squared variances explained by variable categories (red = biological, green = demographic, blue = technical, purple = residuals).

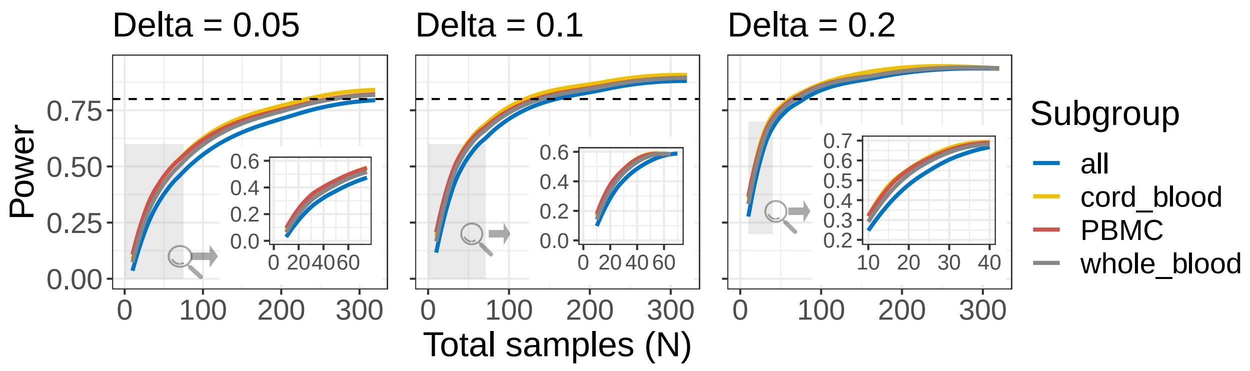
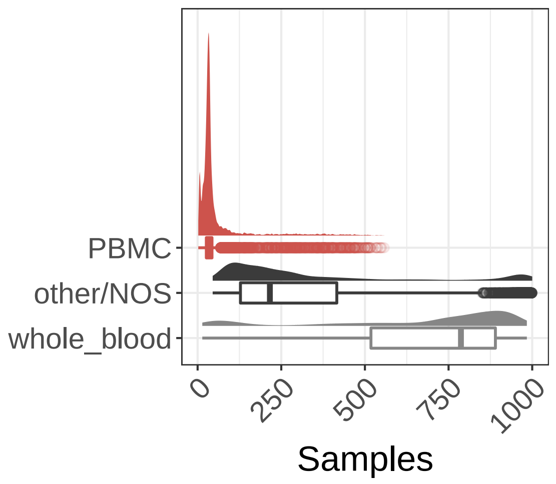
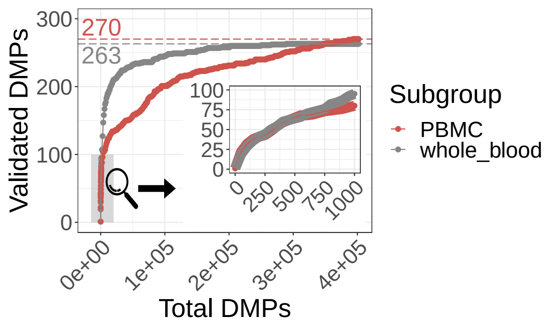


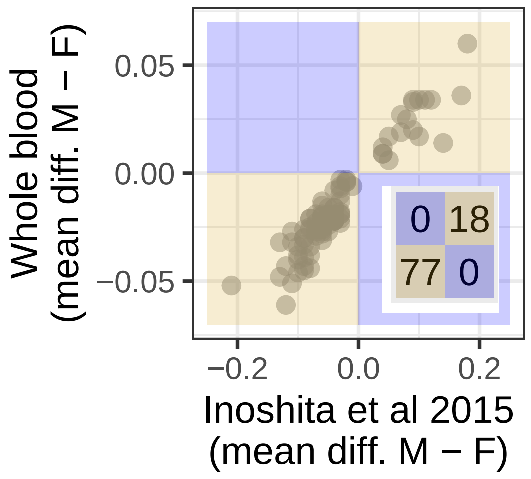
Figure 3.4: Results of power analyses with the simulation-based `pwrEWAS` method [181]. Curves indicate trade-offs between power on the y axes and total samples on the x axes, across 3 delta values (0.05 at left, 0.1 at middle, 0.2 at right). Curve colors show results for each sample type (line colors, blue = all, yellow = cord blood, red = PBMCs, gray = whole blood). Dotted horizontal lines show the 0.8 (80%) target power.



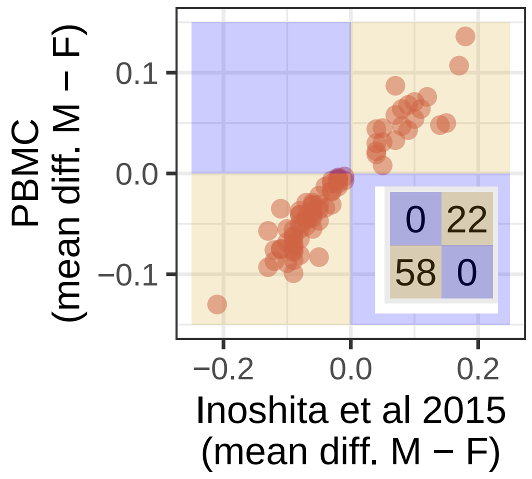
(a)



(b)



(c)



(d)

Figure 3.5: Replication of sex DMPs from [138] (a.k.a "Inoshita et al 2015") using cross-study compilations of whole blood and PBMC. (a) Sample label distributions among the 1,000 nearest neighbors from querying [138] samples, where density and box plots show returned frequencies of 3 sample labels (red = PBMCs, black = other/NOS, gray = whole blood). (b) Concordance of [138] DMPs on the y-axis among the top significant compilation DMPs on the x-axis, ranked on P-values, for whole blood in gray and PBMCs in red. The zoom shows the top 1,000 DMPs, and colored dotted lines and colored numbers indicate total DMPs from [138] (3.2.8). (c and d) Mean Beta-value differences (male - female) at [138] DMPs on y axes and in (c) whole blood and (d) PBMC compilations. Region colors show direction agreement in gold and disagreement in blue, and insets show DMP counts by region.

across sex DMPs from [138], and 100% agreement among the subset of replicated DMPs in each compilation (Figs 3.5c and 3.5d). DNAm proximal to cytosine- and guanine-rich regions, known as CpG islands, can functionally regulate gene expression [73, 75, 77, 250, 251], and most replicated sex DMPs mapped to CpG islands (65/117, 56%). The most significant of these DMPs mapped to a variety of gene regions, including two body DMPs at *RFTN1* and *LOC644649*, and one promoter DMP at *SLC6A4* (P-adjusted $<$ 5.1e-47, Bonferroni method, Table C.4).

3.7 Discussion

We analyzed DNAm array data from the three most prevalent blood sample types in the GEO database and updated the `recountmethylation` Bioconductor package to make reproducible [252, 253] cross-study and cross-platform analyses of these data easier. Since HM450K and EPIC data continue to accumulate rapidly on GEO, we further developed the `recountmethylation_instance` Snakemake workflow to enable semi-automated compilation of the DNAm array data on GEO [61].

We replicated 40% of sex DMPs from [138] and 46% of sex DMPs from [236] using independent whole blood and PBMC compilations. These rates were similar to prior studies of sex DNAm differences, including a 38% validation rate of cord blood sex DMPs between two independent cohorts [235], and 44% validation rate of genes in nasal epithelium with DNAm differences by sex [234]. These results could represent a baseline expectation for replication or independent validation rate of DMPs for sex, and potentially other variables, across independent EWAS.

Our work has several limitations. First, we excluded blood spots from our analyses due to insufficient raw DNAm array data available from GEO, although this blood sample type accounts for a substantial fraction of publicly available data from younger subjects. Another limitation related to data availability is that far fewer blood samples were available for the EPIC platform compared to HM450K as of March 31, 2021. The larger EPIC platform could help expand analyses to new genome regions and clarify regional DNAm signals at CpG islands and genes. The `pwrEWAS` method assumes a technical detection threshold of Beta-value = 0.01 by default, and using this threshold ensures our findings are relevant for both single-study and cross-study analyses. However, this technical threshold likely should be lowered if the study being planned involves cross-study analyses using study ID bias correction, because we found this correction reduced explained variances (Section 3.3.2) and resulted in lower between-group differences in our sex DMP cross-study analysis compared to the single-study discovery EWAS (Section 3.6). Finally, we did not conduct orthogonal or wet-lab validation of replicated DMPs. Such steps would be essential to narrow biomarker candidates and elucidate biological mechanisms explaining differential DNAm.

Our cross-study and cross-platform approach could be applied to other highly prevalent tissues such as brain [3] or expanded to include public bisulfite-sequencing samples from the SRA [16], which could clarify the genome region specificity of high-confidence biomarkers from DNAm arrays [47, 254, 255]. A future update of `recountmethylation` may include such samples.

3.8 Conclusion

Our cross-study and cross-platform approach could be applied to other highly prevalent tissues such as brain [3] or expanded to include public bisulfite-sequencing samples from the SRA [16], which could clarify the genome region specificity of high-confidence biomarkers from DNAm arrays [47, 254, 255]. A future update of `recountmethylation` may include such samples.

3.9 Acknowledgements

We thank Joe Gray, Jeremy Goecks, and Stephanie Hicks for early feedback on the content of this manuscript.

3.10 Funding

NIH [5R01GM121459-02, in part; U01AG060908 to K.H.].

3.11 Disclaimer

The contents do not represent the views of the U.S. Department of Veterans Affairs or the United States Government.

4 Chapter 3: Retained introns in long RNA-seq reads are not reliably detected in sample-matched short reads

4.1 Background

During RNA transcription, multiple spliceosomes may act on the same transcript in parallel to remove segments of sequence called introns and splice together flanking exons [256]. Most splicing occurs stochastically [257] during transcription [258–260], although up to 20% of splicing may occur after transcription and polyadenylation [260, 261] (Fig. D.1). Introns are spliced by several known spliceosome types, of which the most-studied are called U2 and U12 [262]. Splicing is known to occur primarily in the nucleus [263], though there is evidence of cytoplasmic splicing [264–267].

Intron retention (IR) is a form of alternative splicing where an intron normally spliced out during transcript processing remains after processing is complete. IR occurs in up to 80% of protein-coding genes in humans [268] and may affect gene expression regulation [269–275] as well as response to stress [276–278]. Transcripts containing introns may also be stably detained in the nucleus before undergoing delayed splicing ("intron detention," or ID), with implications for temporal gene expression [279]. In cancers, high levels of IR [280–282] can generate aberrant splicing products with known and potential biological consequences for gene expression and cell survival [283]. IR rarely gives rise to a protein product [284, 285], but novel peptides derived from transcripts with retained introns (RIs) are increasingly being studied in disease contexts such as cancer [286–290].

Despite its biological relevance, detection of IR from bulk RNA sequencing (RNA-seq) data remains challenging for two principal reasons: (1) A short RNA-seq read (e.g., from Illumina's HiSeq, NovaSeq, or MiSeq platforms) is almost never long enough to resolve a full intron or its context in a transcript, particularly in genome regions with multiple overlapping transcripts; (2) RNA-seq data may contain intronic sequence from unprocessed or partially processed transcripts, DNA contamination, and non-messenger RNA such as Circular RNA (cRNA) [259, 291], potentially yielding spurious IR calls, independent of read length.

Existing tools designed specifically for RI detection make simplifying assumptions to address the above issues. These tools include Keep Me Around (KMA) [292], IntEREst [293], iREAD [294], superintronic [295], and IRFinder [268] and its most recent implementation as IRFinder-S [184]. Some mitigate challenge (1) by ignoring from consideration any intronic regions that overlap other features (KMA, IntEREst, iREAD), leaving biological blindspots in RI detection [292–294]. Some attempt to mitigate challenge (2) by recommending that a user provides poly(A)-selected data as their input [268, 292, 294, 295], assuming that poly(A) selected data represents fully processed, mature RNA. However, poly(A) selection during library preparation has been shown not to remove all immature post-transcriptionally spliced RNA molecules, and intronic sequences are commonly found in poly(A)-selected RNA-sequencing data [296, 297]. To clarify the quality of and best practices for RI detection, we performed tests on poly(A)-selected, sample-matched long- and short-read sequencing runs for two biological specimens, with processed long-read data providing a standard against which we evaluated short read-based RI detection.

4.2 Methods

4.2.1 Identification of paired short- and long-read data

Two advanced-search queries were performed on the SRA (`https://www.ncbi.nlm.nih.gov/sra`) on July 13, 2021, and all experiment accession numbers were collected from the query results by downloading the resulting `RunInfo` CSV files. For both searches, the query terms included organism "human," source "transcriptomic," strategy "rna seq," and access "public" with platform varying between the two searches: "pacbio smrt" for the long-read query and "illumina" for the short-read query. The `RunInfo` files were merged and projects with both Illumina and PacBio sequencing performed on the same NCBI biosample were identified. Due to relatively low sequencing depth of PacBio experiments, all projects with fewer than 20 PacBio sequencing runs were eliminated. PacBio experiments conducted on any PacBio platform earlier than RS II were also removed. Two remaining biosamples were chosen as data on which to test RI detection: 1) biosample SAMN07611993, an iPS cell line collected and processed by bioproject PRJNA475610, study SRP098984, with 1 short-read and 27 long-read runs [298], and 2) biosample SAMN04251426 (HX1), a whole blood sample collected and processed by bioproject PRJNA301527, study SRP065930, with 1 short-read and 46 long-read runs [299]. (See the project repository at `https://github.com/pdxgx/ri-tests` for accession numbers.)

4.2.2 Long-read data collection, initial processing, and alignment

Raw Iso-Seq RS II data were downloaded from the SRA trace site (`https://trace.ncbi.nlm.nih.gov/Traces/sra`), via the "Original format" links under the "Data access' tab for each run. These comprised three `.bax.h5` files for both samples, with an additional `.bas.h5` and metadata file for each HX1 run. For both samples, individual runs were processed separately as follows, with differences in handling of the two samples as noted. Subreads were extracted to BAM files from the raw movie files using `bax2bam` (v0.0.8). Circular Consensus Sequences (CCS) were extracted using `ccs` (v3.4.0) with `-minPasses 1` set to `1` and `-minPredictedAccuracy 0.90`. Barcodes were removed from CCS reads and samples were demultiplexed with `lima` (v2.2.0). For HX1, the input barcode FASTA files were generated from the Clontech_5p and NEB_Clontech_3p lines from "Example 1" `primer.fasta` (`https://github.com/PacificBiosciences/IsoSeq/blob/master/isoseq-deduplication.md`). For iPSC, forward and reverse barcode fasta files were downloaded from the study's GitHub page (`https://github.com/EichlerLab/isoseq_pipeline/tree/master/data`) and merged into a single FASTA file per the `lima` input requirements. Since `lima` generates an output file for each 5'-3' primer set, these were merged using `samtools merge` (`samtools` and `htslib` v1.9). Demultiplexed reads were refined and poly(A) tails removed using `isoseq3 refine` (`isoseq` v3.4.0) to generate Full-Length Non-Concatemer (FLNC) reads. FLNC reads were extracted to FASTQ files using `bedtools bamtofastq` (`bedtools` v2.30.0), and aligned to GRCh38 with `minimap2` (v2.20-r1061) using the setting `-ax splice:hq`. Sequence download and processing scripts are available at `https://github.com/pdxgx/ri-tests`.

After processing, the 46 HX1 Iso-Seq runs yielded 945,180 aligned long reads covering 32,837 transcripts of 11,813 genes for HX1, with 13,560 of these transcripts covered by at least 5 long reads and 4409 unique $5+$ read transcripts showing evidence of possible intron retention. In the iPSC sample, we obtained 839,558 aligned long reads covering 31,546 transcripts of 11,992 genes. 12,676 of these transcripts were covered by at least 5 long reads, with 3137 unique $5+$ read transcripts showing evidence of possible intron retention.

4.2.3 Assignment of long reads to transcripts

The long-read alignment files were parsed as follows. GENCODE v.35 annotated transcripts' introns, strand, and start/end positions were extracted from the GENODE v35 GTF file. Then for each aligned long read, spliced-out introns, strand and start/end positions were extracted using `pysam` (v0.16.0.1, using `samtools` v1.10) [300, 301]. A set of possible annotated transcripts was generated, comprising transcripts for which the read's set of introns exactly matched the annotated transcripts' introns sets ("all introns"), or if no such transcripts were found, transcripts for which the read's introns were a subset of the transcripts' intron sets ("skipped splicing"). Then the best transcript match was chosen from the shortlist of potential matches as the transcript whose length most closely matched the read length. Some reads did not cover the full lengths of their best-matched transcripts, defined by the read alignment start and end position encompassing all introns in the annotated transcript ("full length"); in the case where not all intron coordinates were covered, these were labeled "partial" reads.

4.2.4 Intron persistence calculation

Intron persistence was calculated only for every transcript that was assigned as the best match for at least 5 reads. We calculated persistence for each intron within these transcripts as the information density of the intron d_i (i.e., the proportion of reads assigned to the transcript that cover intron i) multiplied by the mean of the product of three terms across all long reads assigned to that isoform:

1. The **retention**, or presence, $R_{r,i}$ of a given intron i is 1 if the read wholly contains i or 0 if it is absent/spliced out as annotated in read r.

2. The **spliced fraction** ($SF_{r,i}$) for a given intron i and read r is defined as

$$SF_{r,i} = \frac{|\{i' \in I : R_{r,i'} = 0\}| + R_{r,i} - 1}{|I| - 1}\,, \tag{14}$$

   where I is the set of introns spanned by r and $R_{r,i}$ is defined above. This fraction of spliced introns in a read, with the target intron excluded, represents the splicing progression of the read. A mature RNA molecule should tend to have fewer unspliced introns present than an RNA from the same transcript at an earlier point in splicing progression.

3. The scaled **Hamming similarity** ($H_{r,i}$) for a given read r and intron i is defined as the average number of spliced or unspliced introns that match between the target read and other reads assigned to the transcript that

have intron i spliced the same as in read r, scaled to the number of introns in the isoform:

$$\begin{aligned} H_{r,i} =& \frac{1}{|\{r' \in M^t : R_{r',i} = R_{r,i}\}|} \\ &\times \sum_{\{r' \in M^t : R_{r',i} = R_{r,i}\}} \frac{|\{i' \in I_{r'} \cap I_r : R_{r',i} = R_{r,i'}\}|}{|I_{r'} \cap I_r|}, \end{aligned} \tag{15}$$

where I_r is the set of introns spanned by r, $I_{r'} \cap I_r$ is the set of introns covered by both r and r', M^t is the set of reads assigned as best matches to the same transcript as r and span the target intron i, and $R_{r,i'}$ is as defined above. Any partial reads that are assigned to the transcript as a best match but do not span the target intron are not included in this calculation, and the scaled Hamming similarity between two reads is only calculated for introns covered by both reads. This term accounts for the stochasticity of splicing initiation and progression, since a collection of reads would be more likely to have a dissimilar pattern of unspliced introns if the splicing process remained incomplete.

Persistence $P_{i,t}$ was calculated for each intron i in a given transcript isoform t as information density of the intron d_i times the mean of the product of the three terms above per Equation 16. Since short reads are not assignable to specific transcripts or isoforms, and certain introns fully or partially recur across multiple transcripts, we set the **intron persistence** P_i for a given intron i as the maximum $P_{i,t}$ found for that intron across all transcripts in which it occurs per Equation 17.

4.2.5 Alignment and BAM generation for short-read data

FASTQs were previously generated by other groups using either Illumina's NextSeq 500 (iPSC [298], run id: SRR6026510) or HiSeq 2000 (HX1 [299], run id: SRR2911306), and files were obtained from the SRA using the `fastq-dump` command from the SRA Toolkit (v2.10.8). A `STAR` (v2.7.6a) [153] index was generated based on the GRCh38 primary assembly genome FASTA (`ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_35/GRCh35.primary_assembly.genome.fa.gz`) and GTF (`ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_35/gencode.v35.primary_assembly.annotation.gtf.gz`) files from GENCODE v35 [158]. Reads were aligned with STAR to this index using the `-outSAMstrandField intronMotif` option. Primary alignments were retained for reads mapping to multiple genome regions. SAM files output by `STAR` were converted to both sorted and unsorted BAM files using `samtools sort` and `view` (`samtools` v1.3.1), respectively.

Additionally, for use with KMA [292], `bowtie2` (v2.3.4.3) [302] alignments were performed. Alignment statistics may be found in the project repository (`https://github.com/pdxgx/ri-tests`) and are summarized in Fig. D.18. A FASTA file with intron sequences was generated based on the GRCh38 primary assembly genome FASTA and GTF files from GENCODE version 35 using the `generate_introns.py` script from the KMA package setting 0 bp for the `extension` flag. These intron sequences were combined with the GRCh38 transcript sequence FASTA file from GENCODE version 35 (`ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_35/`

`gencode.v35.transcripts.fa.gz`), and this combined FASTA was used to create a Bowtie 2 index. Reads were aligned to this index using `bowtie2` according to specifications from KMA [303]. To quantify expression from the Bowtie 2 alignments, eXpress (v1.5.1) [304, 305] was used.

4.2.6 Selection of target gene subset

Due to variable short- and long-read coverage across the genome, we selected a subset of genes to use for our test dataset to ensure adequate sequencing coverage for RI detection on both platforms. For the short-read data, we chose a coverage cutoff based on the requirements of the short-read RI detection tools used. The two tools with clear coverage requirements are iREAD, which requires coverage of 20 reads across an intron for RI detection, and superintronic, which requires 3 reads per region. Since these are short-reads (126 bases for iPSC and 90 for HX1) required over potentially long intronic regions, we chose a median gene-wide coverage (including both intronic and exonic regions) of 2 reads per base, ensuring either consistent coverage across the gene or high coverage in some areas. For the PacBio data, we selected 5 long reads per gene, and a further filter of at least 5 reads aligned to a single transcript of the gene, as giving enough information for comparing splicing progression and splicing patterns between reads. The target gene sets, 4,639 genes for iPSC and 4,369 for HX1, were chosen from the aligned data, naive to potential RI detection, and then for both short- and long- read data, the gene subset was applied as a filter after running metric calculations or RI detection by short read tools. Within these genes, only transcripts with at least 5 long reads were studied.

4.2.7 Intron feature annotation

For the set of target genes, transcripts with at least 5 long reads were selected for analysis. Features of each intron in these transcripts including intron lengths, splice motif sequences, relative transcript position, spliceosome category, and transcript feature overlap properties were extracted as follows. Length was calculated as the difference between the right and left genomic coordinates of the intron ends. Relative position within the transcript is an intron-count normalized fraction where 0 represents the transcript's 5' end and 1 represents the 3' end. Splice motifs were assigned to each intron by querying the GRCh38 reference genome with `samtools faidx` (`samtools` v1.10) for the two coordinate positions at each end of the intron, and assigned to one of three canonical motif sequences (GT-AG, GC-AG, and AT-AC, and their reverse complements for $-$ strand genes) or labeled as "other" for noncanonical motifs. Three feature overlap properties were studied: the total number of exons from other transcripts with any overlap of the intron region; the percent of intron bases with at least one overlapping exon from another transcript; and the maximum number of exons overlapping a single base in the intron. These were calculated by extracting all exon coordinates from the GENCODE v35 annotation file, and using an interval tree to query each intron base position against the set of annotated exon coordinates. Spliceosome category was determined from recent U2 and U12 intron annotations [262]. BED files of U2 and U12 introns for GRCh38 were downloaded from the Intron Annotation and Orthology Database (`https://introndb.lerner.ccf.org/`) on 1/25/22. Introns were labeled "U2" or "U12" if they only overlapped ranges from one of either spliceosome category, and remaining introns were labeled "other."

4.2.8 Selection of short-read RI detection algorithms and identification of likely RIs

We successfully downloaded and ran five IR-detection tools for short-read data on our remote server using the CentOS v7 operating system. To run superintronic, KMA, IntEREst, and iREAD, we used conda virtual environments (see `https://github.com/pdxgx/ri-tests`, Appendix A.5 and A.6). We ran IRFinder-S from a fully self-contained Dropbox image per the tool's instructions (see below). IntEREst and superintronic are provided as R libraries which we ran from interactive R sessions, while iREAD, IRFinder-S, and KMA were run from command line, and a separate R package was used for RI detection for KMA. Outputs from all tools were read into R and harmonized to a single set of intron ranges after applying minimum coverage filters based on both short-read and long-read data. After running tools according to their provided documentation, we consulted literature and documentation on a tool-by-tool basis to devise starting filter criteria based on expression magnitude and other properties. We used these starting criteria to find the subset of most likely RIs, then we modified filter criteria to ensure filtered intron quantities were roughly one order of magnitude lower than unfiltered introns in both iPSC and HX1.

4.2.9 IR quantification with IntEREst

To run IntEREst (v1.6.2) [293], the `referencePrepare` function from the package was used to generate a reference from the GENCODE v35 primary assembly GTF file [158]. This reference was used along with the sorted STAR BAM alignment from each sample to detect intron retention with the `interest` function, considering all reads and not just those that map to junctions. We used the `interest` function with the `IntRet` setting, which takes into account both intron-spanning and intron-exon junction reads and returns expression as a normalized Fragments Per Kilobase Million (FPKM). The filter FPKM ≥ 3, recommended for iREAD, left $> 90\%$ of introns in both samples, so we increased the minimum filter to FPKM ≥ 45, and this retained $5038/32544 \approx 15\%$ of introns in HX1 and $6832/21820 \approx 31\%$ of introns in iPSC (Fig. D.11).

4.2.10 IR quantification with keep me around (KMA)

To run KMA [292], we used devtools to install a patched version of the software which resolves a bug unaddressed by the authors, available at `https://github.com/adamtongji/kma`. The `read_express` function was used to load expression quantification data output from eXpress, and the `newIntronRetention` function was used to detect intron retention. Returned intron expression was scaled as Transcripts Per Million (TPM). We noted the recommended filters of unique counts ≥ 3 and TPM ≥ 1 left just 7.2% of introns in iPSC versus 19% in HX1, so we used a less stringent filter of unique counts ≥ 10 for both samples, which left $6437/14155 \approx 45\%$ of introns in iPSC and $5089/20484 \approx 25\%$ of introns in HX1 (Fig. D.11).

4.2.11 IR quantification with iREAD

To run iREAD (v0.8.5) [294], a custom intron BED file was made from the GENCODE v35 primary assembly GTF file using GTFtools (v0.6.9) [306]. The total number of mapped reads in each sorted STAR BAM alignment was determined using samtools, and used as input to the iREAD python script to detect intron retention. Intron expression was returned scaled as FPKM. To identify the most likely RIs, we applied previously published filter recommendations for entropy score (≥ 0.9) and junction reads (≥ 1). Since there were relatively few introns remaining after applying published filters to the iPSC short-read data ($313/19316 \approx 1.6\%$ vs. $583/7748 \approx 7.5\%$ in HX1), we applied lower filters for FPKM (≥ 1 vs. ≥ 3) and read fragments (≥ 10 vs. ≥ 20) (Fig. D.11).

4.2.12 IR quantification with superintronic

To run superintronic (v0.99.4) [295], intronic and exonic regions were gathered from the GENCODE v35 primary assembly GTF file [158] using the `collect_parts` function. The `compute_coverage` function was used to compute coverage scores for each sample from sorted STAR BAM alignments, and the `join_parts` function was used to convert these scores to per-feature coverage scores. Intron expression was returned as $\log_2$-scaled coverage, and we identified retained intron ranges as those overlapping long read-normalized ranges with LWM ≥ 3, per the expressed introns filter described in [295] (Fig. D.11).

4.2.13 IR quantification with IRFinder-S

We ran IRFinder-S v2.0-beta using the Docker image obtained from `https://github.com/RitchieLabIGH/IRFinder`. We prepared the IRFinder reference files using the GENCODE v35 genome sequence reference and intron annotations [158]. Our analyses focused on the coverage and IRratio metrics, and the intron expression profile flags returned under warnings. Intron expression was returned as an IRratio, which is similar to Percent Spliced In (PSI), and we identified likely retained introns as having IRratio ≥ 0.5 without any flags per the methods in [184] (Fig. D.11).

4.2.14 Harmonization of intron retention metrics across algorithms and runs

Prior to analysis, we harmonized algorithm outputs on intron ranges returned by analysis of available long read runs. We harmonized intron expressions from short read RI detection tools to intron ranges remaining after long reads were uniquely mapped to transcript isoforms. For each short-read RI detection tool, we calculated the region median intron expression value after weighting values on overlapping range lengths (a.k.a. Length Weigthed Medians or "LWMs"). Calculation of LWMs is shown for an example intron in Fig. D.17. Inter-rater agreement among the output from different short-read algorithms was assessed by Fleiss' kappa [307] using the R package `irr` v0.84.1.67 [308].

4.2.15 Calculation of performances by intron length bins

We calculated called RI performance metrics across five short-read tools for a series of overlapping intron length bins. In total, 41 bins were calculated for each sample by sliding 300 bp-wide windows from 0 to 4300 bp lengths at 100 bp intervals. Plots were generated by computing LOESS smooths of the binned performance results.

4.2.16 Calculation of normalized binned coverages

We evaluated binned intron characteristics across intron truth metric categories for each sample. We assigned introns to truth categories if they were recurrent in that category for ≥ 4 of 5 short-read tools (e.g., an intron that was recurrent TP for four tools in iPSC, etc.). We then calculated the log10 median short-read coverage for 1,000 evenly spaced bins per intron for each truth category. We further calculated percent of introns overlapping an exon for each bin by using annotations from the GENCODE v35 GTF. Plots were generated by computing the LOESS smooths of the binned results.

4.2.17 Comparison of detected RIs with circular RNA and validated RIs

We downloaded a database of human circular RNAs from circbase [309] (`http://www.circbase.org/download/hsa_hg19_circRNA.txt`), most recently updated in 2017. We extracted all cRNA labeled with the "intronic" flag in the annotation column and performed a liftover of genomic coordinates for these cRNAs from hg19 to GRCh38 using the UCSC Genome Browser `liftover` tool (`https://genome.ucsc.edu/cgi-bin/hgLiftOver`). For each sample, we determined the percent of introns overlapping at least one cRNA for the $4+$ consensus truth metric groups TP, FP, and FN (e.g., intron was TP in ≥ 4 tools, etc.)

In order to test introns in this study against experimentally validated RIs, we identified wet-lab studies in the literature that had first predicted, and then validated intron retention. We identified 4 such studies [272, 310–312] that validated a total of 9 RIs in our sets of target genes as defined above (5 and 7 in HX1 and iPSC respectively) (Table D.3). (The above four plus an additional ten studies [264, 285, 313–320] experimentally validated RIs in an additional 6 and 9 genes that were found in our target gene sets for in HX1 and iPSC respectively, but without evidence of IR in our samples, and 41 and 36 genes, respectively, that did not pass our sample coverage thresholds for inclusion in this study.) The validated intron coordinates (Table D.3) were extracted either from the published intron number [272, 310, 311], assuming a count from the gene's 5' to 3' end, or via BLAT queries of the target sequence [312]. In each sample and tool, we determined the truth status (TP, FP, TN, or FN) of all introns of all transcripts in the target gene for transcripts with ≥ 5 long reads. Adequate intron expression information was available in both samples for the genes *LBR*, *CELF1*, *AB1G2*, but only one sample each for remaining genes.

4.3 Results

4.3.1 Testing RI detection using sample-paired short- and deep long-read RNA-seq data

To generate a dataset to test RI detection, we identified two human biological specimens on the SRA with RNA-seq data from both Illumina short-read and PacBio Iso-Seq RS II long-read platforms (Fig. 4.1). These were a HX1 [299] and iPSC [298], with, respectively, 46 and 27 Iso-Seq runs, 24.4 and 91.3 million aligned short reads, and 945 and 840 thousand aligned long reads (Table D.1). To confine attention to robustly represented loci, we identified a set of 4,369 and 4,639 target genes in HX1 and iPSC samples, respectively, each with ≥ 2 short reads per base median coverage across the full gene length and ≥ 5 long reads assigned to at least one isoform of the gene (Fig. D.2).

We sought to quantify IR in each biological specimen using long-read data, accounting for random splicing and sample contamination that may lead to noisy splicing patterns. For a given intron i and transcript t, we defined persistence $P_{i,t}$ as

$$P_{i,t} = d_i \cdot \sum_{r \in M^t} \frac{R_{r,i} \cdot SF_{r,i} \cdot H_{r,i}}{|M^t|} \,, \tag{16}$$

where r is a read among the set of all reads M^t assigned as best matches to transcript t, information density d_i is the proportion of M^t covering intron i, the binary variable $R_{r,i}$ is 1 if and only if r provides evidence for the retention of i, and the spliced fraction $SF_{r,i}$ and scaled Hamming similarity $H_{r,i}$ are defined in Section 4.2 (see Equations 14 and 15). In brief, the intron persistence $P_{i,t}$ incorporates the extent and similarity of splicing across transcript reads, accounting for stochastic splicing initiation and progression (Fig. D.1). Finally, to address ambiguity in transcripts of origin in short-read data, we defined intron i's persistence P_i as the maximum persistence across all isoforms T_i that contain i:

$$P_i = \max_{t \in T_i} P_{i,t} \,. \tag{17}$$

Going forward, we define a "persistent intron" as an intron for which $P_i \geq 0.1$.

Across all transcripts studied in both samples, a substantial majority (83.7%) of introns were fully spliced out ($P_{i,t} = 0$), and a small minority (0.15%) of introns were always unspliced within a transcript ($P_{i,t} = 1$) (Figs. D.5a and D.3). These extreme values are in keeping with our qualitative understanding of splicing patterns; however, the range of intermediate persistence values appears to represent a spectrum with varying extents of inconsistent splicing across and between reads. While we tested short-read RI detection on a per-sample basis, we also compared intron persistence patterns between HX1 and iPSC samples and found significant similarity in splicing patterns across matched transcripts (Figs. D.3 and D.4).

4.3.2 Similarities of intron properties across short-read RI detection tool outputs

We compared RIs called by five tools for short-read data (Table 4.1). While most introns were consistently spliced out, 39.9% (1,743/4,369) and 31.4% (1,457/4,639) of target genes in HX1 and iPSC, respectively, had at least one RI identified in either short- or long-read data. Expression of called RIs varied substantially between tools in both HX1 (Fleiss' $\kappa = 0.282$) and iPSC (Fleiss' $\kappa = 0.162$), though we did observe moderate overall correlation between the output of IntEREst, superintronic, and KMA (Fig. D.6). Further, using circBase [309] to probe whether cRNA contamination may have affected RI detection, we identified only a small percent ($< 5\%$) of called RIs that appeared to overlap intronic cRNAs (Fig. D.7).

We next examined the distributions of several intron properties (length, relative position in transcript, and annotated exon overlap) and their relationships with the set of RIs called by each short-read tool and their relative expression levels (Figs. D.5b and D.8). Unsurprisingly, tools that exclude introns with overlapping genomic features (i.e. KMA, IntEREst,

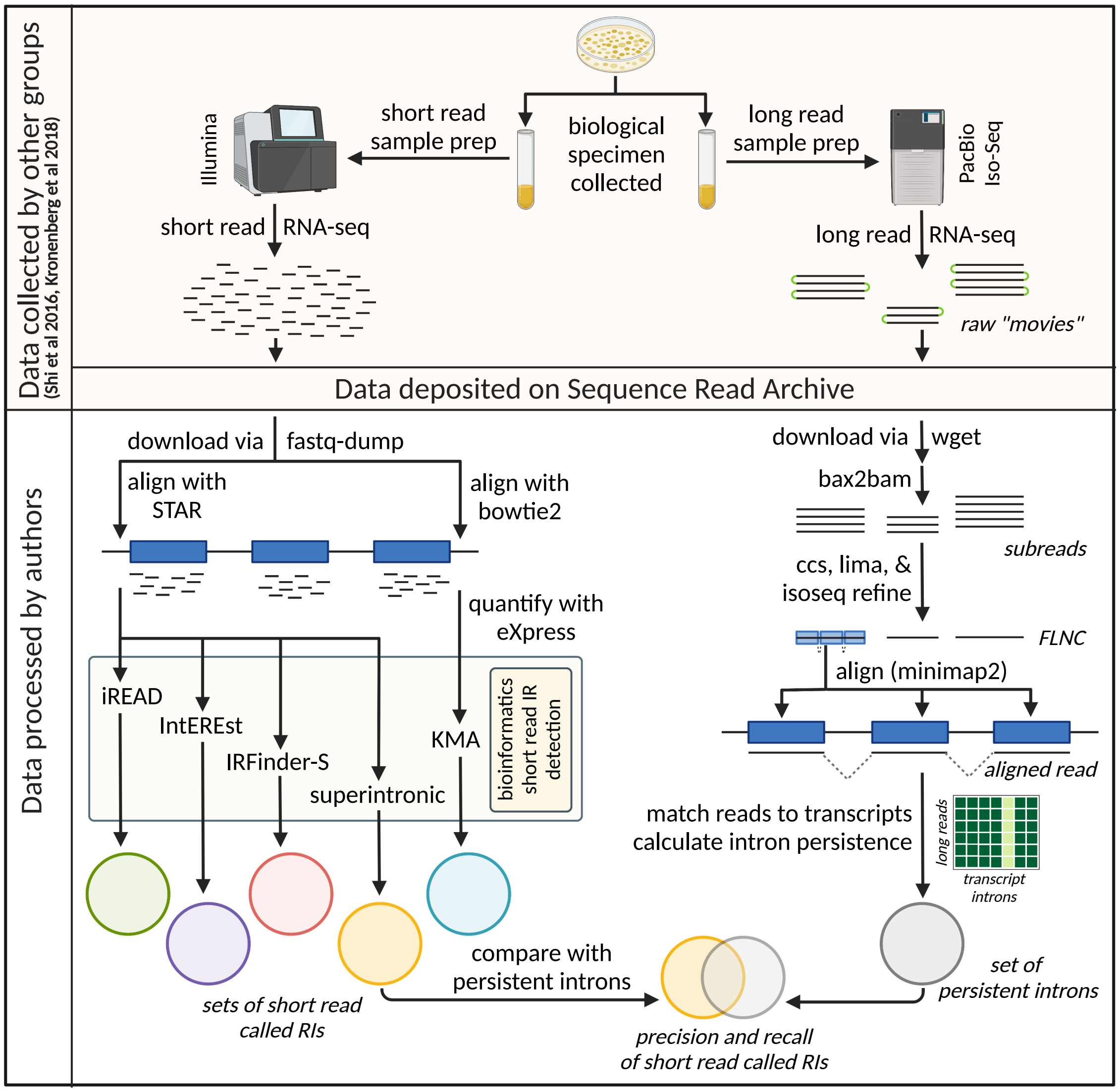

Data collected by other groups
(Shi et al 2016, Kronenberg et al 2018)
Illumina
short read sample prep
biological specimen collected
long read sample prep
PacBio Iso-Seq
short read
RNA-seq
long read
RNA-seq
raw "movies"
Data deposited on Sequence Read Archive
Data processed by authors
download via
fastq-dump
align with STAR
align with bowtie2
quantify with eXpress
iREAD
IntEREst
IRFinder-S
superintronic
KMA
bioinformatics short read IR detection
sets of short read called RIs
compare with persistent introns
precision and recall of short read called RIs
download via
wget
bax2bam
subreads
ccs, lima, & isoseq refine
FLNC
align (minimap2)
aligned read
match reads to transcripts calculate intron persistence
long reads
transcript introns
set of persistent introns

Figure 4.1: Overview of experimental plan. Long and short read RNA-seq data from the same biological specimen [298, 299] were downloaded from the SRA and subject to processing and analysis. Short reads (left path) were aligned and quantified according to the requirements of five short-read RI detection tools [184, 292–295], and retained introns were called by each of these. The raw long Iso-Seq reads (right path) were processed to the stage of FLNC reads, but left unclustered. After long reads were aligned to the reference genome, each aligned read was assigned to a best match transcript or discarded, and intron persistence was calculated. The called RI output of each short read detection tool was compared against the set of persistent introns identified in the long read data (where $P_i \geq 0.1$). Created with `BioRender.com`.

Tool	IRFinder-S [184]	superintronic [295]	iREAD [294]	KMA [292]	IntEREst [293]
Year	2021	2020	2020	2015	2018
IR measure[a]	IRratio	log coverage	FPKM	TPM	FPKM or PSI
Language	C++	R	Python	Python, R	R
Host website	GitHub	GitHub	GitHub	GitHub	Bioconductor
Sample data format	BAM or FASTQ	BAM	BAM	FASTQ	BAM
Reference format	GTF	GTF/GFF3	BED	FASTA, GTF/GFF3	GTF/GFF3
Intron definition	All introns	All introns	Independent introns[b]	Independent introns[b]	Independent introns[b]

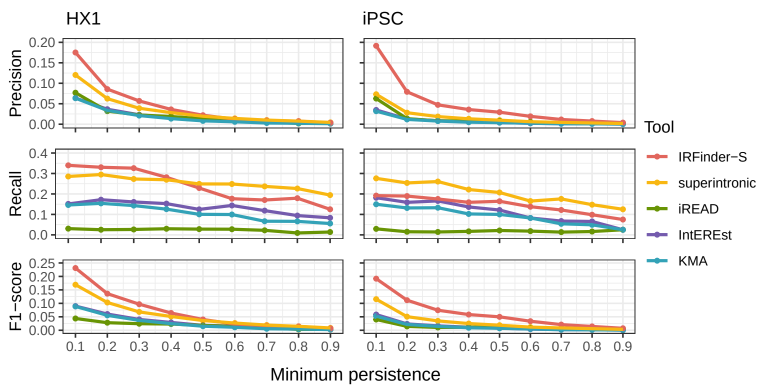
[a]See Section 4.2 for measure definitions.
[b]Independent introns are intron regions not overlapping features from other transcript isoforms.

Table 4.1: Short-read tools studied.

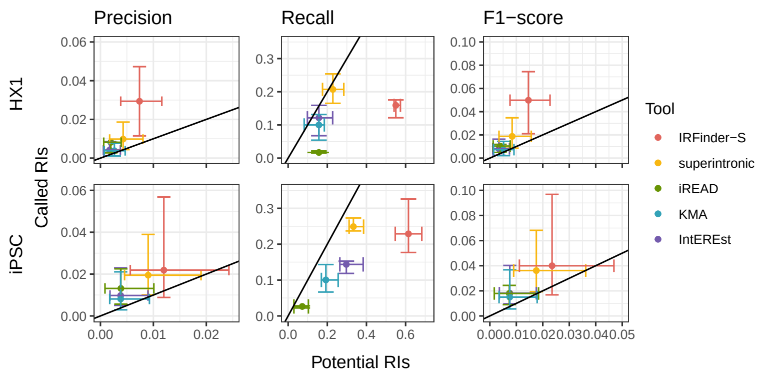
iREAD; Table 4.1) had exceedingly low overlap between exons and the IRs they reported. We also note that KMA and IntEREst called extremely long RIs (up to > 297 kilobases), compared to those called by other short-read tools or the persistent introns identified from long read data (maximum 6,275 and 5,926 bases in HX1 and iPSC). We observed a slight overall 3' bias among persistent introns from long-read data, as well as the set of RIs from several short-read tools (Fig. D.5b), potentially reflecting the relatively shorter duration of exposure of 3' introns to the cotranscriptional splicing machinery and/or implicit 3' bias of the Clontech sample prep [321] used in both samples [298, 299]. Despite this slight 3' tendency, there was no appreciable association between intron persistence and intron position in transcript (Fig. D.9). Among all tools, IRFinder-S called a set of RIs with characteristics most similar to persistent introns from long-read data (Fig. D.5b).

4.3.3 Precision and recall are poor across short-read RI detection tools

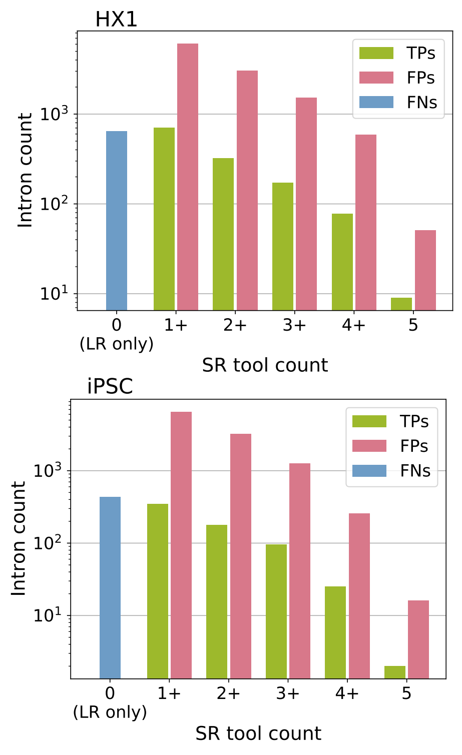
We tested performance (precision, recall, and F1-score) of RI detection by five short-read tools, comparing sets of called RIs against persistent introns identified from long read data (defined as $P_i \geq 0.1$). Overall tool performance was poor in all cases (Fig. 4.2a, Table D.2). Many persistent introns (55% and 48% in iPSC and HX1, respectively, Fig. D.10) were not called by any short-read tool, and the majority of called RIs were neither identified among persistent introns in long-read data nor consistently called between short-read tools (Figs. 4.2c and D.10). In HX1 and iPSC, respectively, 54% and 49% of called RIs were not called by more than one tool (52.4% overall). IRFinder-S had the best performance across most metrics, possibly due to the similarity between the properties of its called RIs and properties of persistent introns. By contrast, iREAD demonstrated the lowest recall across all tools, likely due to its sparse calling of RIs (Fig. D.11). Performance metrics for IntEREst and KMA were very similar across both samples (Fig. 4.2b).



(a)



(b)



(c)

Figure 4.2: (a) Short-read tool performance across different thresholds of intron persistence. Each panel displays tool performance along the y-axis (measured by one of precision, recall, or F1-score as labeled) for a set of introns defined by the indicated threshold for intron persistence along the x-axis. Data for HX1 and iPSC are shown at left and right, respectively, with each tool's per-sample performance depicted in a different color (IRFinder-S [red], superintronic [yellow], iREAD [green], IntEREst [purple], and KMA [blue]). (b) Variation in short-read tool performance across intron persistence thresholds for potential vs. called RIs. Each panel displays tool performance as measured by precision (left), recall (middle), and F1-score (right) for HX1 (top) and iPSC (bottom) samples. The performances for each tool's potential RIs and called RIs are shown along the x- and y-axes, respectively, with centroid and whiskers denoting, respectively, the median and interquartile range of tool performance across intron persistence thresholds. Each tool's performance is depicted in a different color (IRFinder-S [red], superintronic [yellow], iREAD [green], IntEREst [purple], and KMA [blue]). Reference lines are shown with slope of 1. (c) Varying degrees of consensus of retained intron calls among short-read tools. Bar plots depict the number of TP (green), FP (pink), and FN (blue) intron calls (y-axis) consistent across a specified number of short-read tools (x-axis). Upper and lower panels depict HX1 and iPSC data, respectively. LR denotes long-read data, SR denotes short-read data.

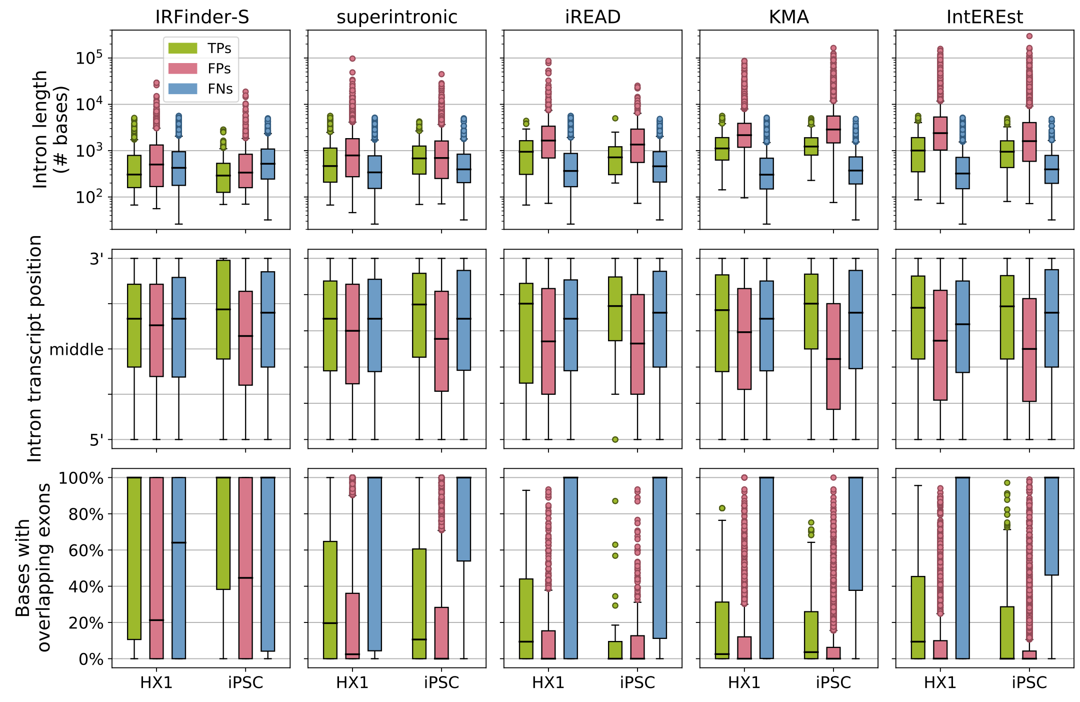
To address sensitivity in persistent intron identification, we also considered short-read tool performance on subsets of long-read introns with increasing minimum thresholds of intron persistence ($P_i \geq 0.1$ to 0.9 in increments of 0.1). We found that overall performance remained poor across all levels of intron persistence, with uniformly worse precision, recall and F1-score as intron persistence increased (Figs. 4.2a and D.12). While individual tool performance varied significantly, IRFinder-S and superintronic were consistently best performers, albeit interchangeably depending on the sample, metric assessed, and intron persistence threshold. For instance, IRFinder-S demonstrated highest recall in HX1 at the lowest cutoff values ($P_i \geq 0.1$ to 0.4), while superintronic demonstrated higher recall across higher thresholds in HX1 and for all cutoffs in iPSC (Table D.2).

Finally, since each tool is capable of calling RIs with different levels of stringency, we evaluated tool performance on a raw set of all potential RIs (all expressed introns detected by that tool) vs. the corresponding subset of introns called as RIs by that tool. Rather than improving overall performance by retaining persistent RIs and removing false positive ones, stringency filters improved precision at the expense of recall, with a slight corresponding improvement in F1-score across tools (Fig. 4.2b, Supplementary File [`supplementary_performance_data.xlsx`]).

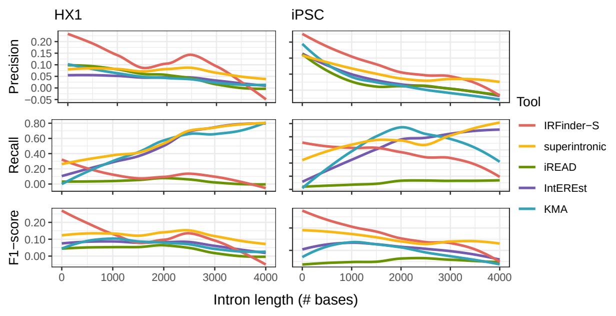
4.3.4 Short introns and introns that do not overlap exons are more reliably called

We next compared the distributions of six intronic properties (length, position, exonic overlap, splice site motifs, U2- vs. U12-type spliceosomes, and uniformity of coverage by mapped reads) between the sets of TP, FP, and FN RIs for each tool. Every tool except IRFinder-S had difficulty identifying shorter RIs (< 600 bases) (Figs. 4.3a and 4.3b). FPs tended to be longer than either TPs or FNs, and were distributed more centrally within a transcript compared to persistent introns (both TPs and FNs) across all tools (Figs. 4.3a and D.14). Further, there was a relative 3' bias for the small subset of FPs that were shared across all short-read tools, potentially reflective of the minimum coverage filters for most tools combined with sequencing coverage bias [322] (Fig. D.14). As expected, the overwhelming majority of introns across all tools had canonical GT-AG splice motifs and splicing by the U2 spliceosome, while FNs showed increased frequencies of other motifs and spliceosome types relative to FPs and TPs (Fig. D.15).

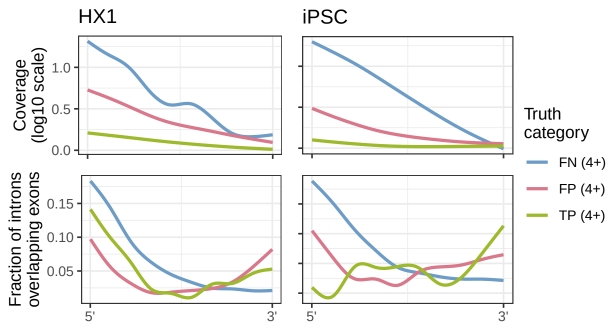
We also probed how much distributional uniformity of mapped read coverage across an intron (coverage "flatness" [184, 294]) and incidence of overlapping exons differed among TPs, FPs, and FNs. Coverage of FPs and to a greater degree FNs was nonuniform, where coverage decreased roughly monotonically from 5' to 3' intron ends. Coverage of TPs was comparatively uniform, where coverage was in general substantially lower than for FPs and FNs (Fig. 4.3c, top two plots). Closer to their 5' ends, FNs were distinguished by their tendency to overlap exons (Fig. 4.3c, bottom two plots). Indeed, for superintronic, iREAD, KMA, and IntEREst, the majority of FNs appear to be accounted for by overlapping exons (Fig. 4.3a). Overlapping exons may thus be a key obstacle to improving recall of many short-read RI detection tools.



(a)



(b)



(c)

Figure 4.3: (a) Distributions of properties of TP, FP, and FN RIs across short-read detection tools. Each panel displays the boxplot distributions of intron length (top, log scale), relative position in transcript (middle), and % of intron bases with overlapping annotated exons (bottom) for the output from each of five short-read tools (from left to right: IRFinder-S, superintronic, iREAD, KMA, IntEREst). Y-axes correspond to intron properties as labeled, with each boxplot along the x-axis corresponding to the TP (green, left boxes), FP (pink, middle boxes), and FN (blue, right boxes) calls for HX1 (left) and iPSC (right). (b) Short-read tool performance as a function of intron length. Each panel depicts the LOESS-smoothed precision (top), recall (middle), or F1-score (bottom) in either the HX1 (left) or iPSC (right) sample across overlapping, sliding window intron length ranges (Section 4.2). Smooths are grouped and colored by five short-read tools (red = IRFinder-S, yellow = superintronic, green = iREAD, purple = IntEREst, blue = KMA). (c) Read coverage and exon overlap as a function of position within an intron. LOESS-smoothed short-read data (see Section 4.2) show the median log10-scaled coverage (top row, y-axes) and fractions of introns with overlapping exons (bottom row, y-axes) as a function of position (x-axis, 5’ $\rightarrow$ 3’ on positive strand) for HX1 (left column) and iPSC (right column). Introns were grouped by truth category membership for at least 4/5 tools (colors, blue = FN, pink = FP, green = TP).

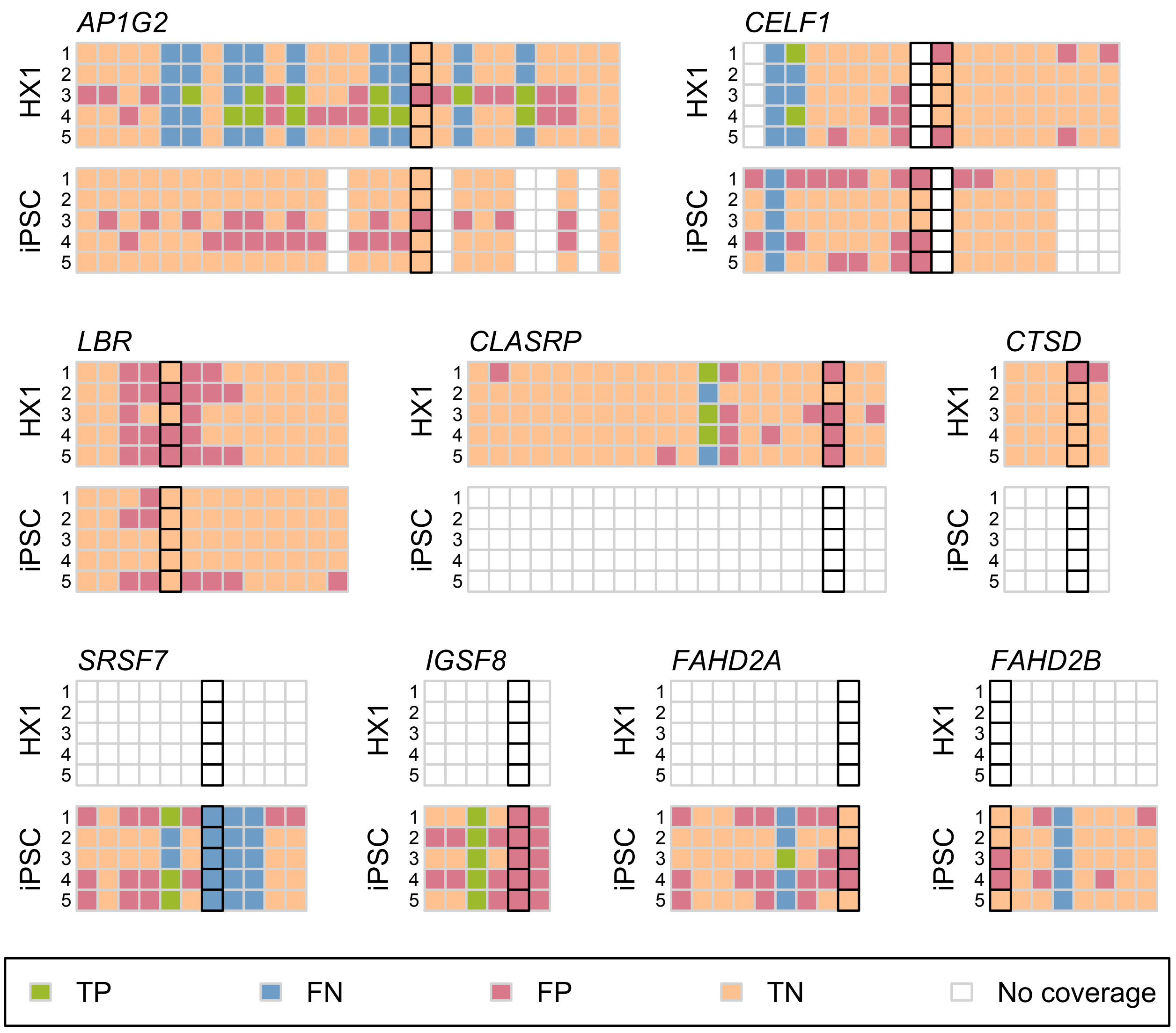


Figure 4.4: Short-read tool performance across nine genes with experimentally validated RIs. Comparison of short-read tool called RIs with introns detected in long-read data are shown as a pair of matrices for each of nine genes (*AP1G2*, *CELF1*, *LBR*, *CLASRP*, *CTSD*, *SRSF7*, *IGSF8*, *FAHD2A*, and *FAHD2B*). The rows in each matrix correspond to the results from each of five short-read tools (from top to bottom: 1: IntEREst, 2: iREAD, 3:IRFinder-S, 4: superintronic, 5: KMA) applied to either HX1 (top) or iPSC (bottom) data; columns correspond to all introns found across all annotated transcript isoforms of the indicated gene, ordered by left and then right genomic coordinates. Each cell in the matrix depicts the presence or absence of an intron in short-read and/or long-read data as a TP (green), FN (blue), FP (pink), and TN (peach) assessment; white boxes indicate introns found only in transcripts with < 5 assigned long reads. Black outlines indicate the experimentally validated RI(s) in each gene.

4.3.5 Persistent introns or called RIs occur in genes with experimentally validated IR

Finally, we searched the literature and third-party resources for independent evidence of persistent introns appearing in the HX1 and iPSC samples studied here. We examined RI presence in 9 genes (5 in HX1 and 7 in iPSC) that have experimentally validated IR from a variety of cell types and tissues (Table D.3) [272, 310–312]. We found that intron retention across these 9 genes varied substantially by sample (no TP introns were observed in both HX1 and iPSC) (Fig. 4.4). We also found significant variation between the set of RIs in these genes called by different short-read tools, with only a single TP intron in *IGSF8* identified across all tools for iPSC (Fig. D.16). Interestingly, the genes *SRSF7* [279, 323] and *AP1G253* [324] appear to be generally enriched for persistent introns, potentially consistent with post-transcriptional splicing [261, 325].

4.4 Discussion and Conclusion

This is the first study to evaluate the quality of short-read RI detection using short- and long-read RNA-seq data from the same biological specimen. This study also establishes a novel metric capturing the persistence of an intron in a transcript as it is processed using deep long read RNA-seq, and it is the first to interrogate the potential effects of splicing progression during transcript processing and spurious sources of intronic sequence. We find that short-read tools detect IR with poor recall and even worse precision, calling into question the completeness and validity of a large percentage of putatively retained introns called by commonly used methods. While our results indicate that it may be possible to improve precision slightly by applying expression filters to potential RIs, this appears to come at significant expense to recall.

This work raises fundamental questions regarding how results from short-read RI detection tools should be interpreted. We have taken IR to mean the persistence of an intron in a transcript after processing is complete, in alignment with the biological literature on IR. Short-read RI detection tools are commonly thought to identify such retained introns, with the assumption that poly(A) selection is sufficient to guarantee fully spliced and mature transcripts for sequencing; however, these tools are not inherently designed to distinguish intron retention from contaminating events such as partial transcript processing. This disconnect between how tool developers and tool users employ the same language may be responsible for false assertions in the published literature about which introns are retained. We note, for instance, that the prediction of putative neoepitopes arising from IR [286–290] requires confidence in the detection of stable, persistent IR with a high likelihood of translation and a low likelihood of undergoing NMD, none of which is assured by short-read RI detection tools.

Limitations of this work include the small number of biological specimens with matched short and deep long read RNA-seq available in the public domain, the lack of replicates of short-read RNA-seq data in this setting, and the limited depth of the long read sequencing data. As a result, we were unable to study the patterns of IR across tissue type and other distinguishing sample characteristics. We confined attention to introns that occur in genes with high coverage in both short and long read data, and did not address either confidence in IR as a function of read depth or systematic biases in gene coverage as a function of sequencing platform. While an improvement, our intron persistence metric only partially accounts for admixed splicing patterns from different cell types in a mixed-cell sample such as HX1. Like other RI detection studies [270–272, 280, 286, 289, 297], our approach is explicitly linked to annotation (here, GENCODE v35) and therefore reports IR only relative to annotated transcripts, ignoring potential unannotated transcripts. We also did not explore the entanglement of biological and technical effects in the length of persistent introns: shorter introns are more likely to be retained [297, 326, 327], but the length limit of PacBio Iso-Seq reads of up to 10 kilobases means that any molecules with longer persistent introns were not considered in this study. Furthermore, we calculated length-weighted median expression to harmonize short-read tool outputs to long-read intron ranges (Fig. D.17), and this stringent approach may have inflated false negative rates in regions returning high expression magnitudes and variances. Finally, we were only able to evaluate a small subset of the tools available for short read-based RI detection, as many of these tools harbor substantial software implementation and reproducibility challenges.

While there is evidence for cytoplasmic splicing, the phenomenon is rare in many tissues and cell types [264–267]. It may be worth exploring the extent to which sequencing only cytoplasmic RNAs focuses attention on fully processed RNA transcripts in future work.

5 Conclusions and future directions

5.1 Key findings from analyses of public omics data

We analyzed public DNAm array data and RNA-seq data in novel cross-study analyses, and this lead us to several useful insights. In Section 2.3.10 we reported CpG probes with low DNAm variances across tissues, and these could be filtered to boost power in future EWAS studying differential DNAm across seven normal tissues. Further, in Section 2.3.11 we reported CpG probes with high tissue-specific variances, and these could be important for showing tissue-specific DNAm patterns in future EWAS. Further, Section 3.3.2 studied the impact of two types of linear adjustments on study ID bias across tens of thousands of simulations. We found a uniform study ID bias adjustment across all available studies is justified, reduced total explained variance, and increased fractional explained variances from biological and demographic variables.

We used public sample-matched short-read and long-read RNA-seq data to study the reliability of RI's called from short-read data. We developed and published `conda` [183] environments (Appendix A.5) which allowed us to run five tools for RI detection: IntEREst, KMA, iREAD, superintronic, and IRFinder-S. We further described intron characteristics which help explain whether they are reliably called in both short-read and long-read data, including length and exon overlap (Section 4.3.2). These findings can inform the fine-tuning and design of RI detection tools for short-read data in the future. In summary, the above findings show applications of omics data from the SRA and GEO, which can inform future experiment planning and provide and novel cross-study analyses.

5.2 Resources for future analyses

This dissertation presents several resources which can aid experiment design and future analyses of public omics datasets. First, for DNAm arrays, we compiled and uniformly processed all public samples with available raw data for HM450K and EPIC, the two most prevalent DNAm array platform types. These compilations feature uniformly processed and harmonized sample metadata with DNAm array data provided as raw color intensity signals, normalized M and U signals, and normalized Beta-value fractions, respectively. We provided compiled data in the HDF5 file format, which uses chunking and compression and has broad support across many programming environments. We also provided compiled data as `HDF5-SummarizedExperiment` hybrid objects, which combined the benefits of `HDF5` and `SummarizedExperiment` filetypes for the benefit of R/Bioconductor users.

We supported use of these comprehensive DNAm array data compilations with the `recountmethylation` [38] Bioconductor package. Early versions of this package provided key functions to download and query large DNAm array data compilation files. We initially compiled data from tens of thousands of samples run using the HM450K platform. With periodic updates over the course of almost two years, we expanded compiled data to include more recent samples and to add samples processed using the newer EPIC platform. We provided vignettes showing key cross-study analyses, how to perform simulation-based power analyses using the `pwrEWAS` [181, 182] method, how to infer genetic ancestry from DNAm data using the `GLINT` [32] software, and more. We also added support for preprocessing, including functions supporting Illumina's BeadArray tests [74] as well as signal threshold cutoffs [27], and we made several lists of cross-reactive probes available to use in probe quality filters [47, 328].

5.3 Analysis extensions

While we identified high-confidence biomarker candidates of differential DNAm for sex and gestational age, these remain to be experimentally and orthogonally validated such as with targeted methylite or pyrosequencing assays [130–132]. These integrative analyses can help elucidate the underlying mechanisms and potential biological impact of these differentially methylated sites [92].

The `recountmethylation` Bioconductor package and associated data compilations can be expanded on in several ways. First, key analyses from Section 3 performed in blood samples can be explored for other prevalent tissues such as brain. These analyses should also be pursued for disease tissues such as cancer, for which specific normalization and quality assurance practices likely necessary. Further, support for DNAm array data compilations can be extended to include WGBS and other bisulfite sequencing data types made available in the SRA. Bisulfite sequencing data is typically higher-resolution than DNAm array platforms, and it has the added benefit of showing the phased states of CpGs sharing a single DNA strand. Bisulfite sequencing data could either be analyzed asynchronously with arrays (e.g. with separate discovery and validation experiments) or harmonized with arrays and analyzed together as a larger data compilation.

The reliability of RI detection tools can be further studied with new sample-matched RNA-seq data as it becomes available in the SRA. It is possible that additional sample-matched data is available in the SRA but not adequately identifiable from metadata, and thus improved methods of mapping, harmonizing, and querying SRA metadata could lead to expanded sample sizes for future analyses. Further, our results and approach to analyze sample-matched data could be used to fine-tune and improve the capabilities of existing RI detection tools or to inform development of altogether new tools. If recent progress developing the IRFinder-S algorithm is any indication, future resources will make greater use of machine learning methods [184].

5.4 Future applications

It will be important for future cross-study analyses to approach hypothesis testing systematically and with a mind to prior work. While it may appear at the outset that there is an extremely large set of possible studies that can be performed, a small subset of possible approaches are probably useful in practice. This is partly because it can be challenging to correct for noise, artifacts, and technical bias across studies, and there is need for expanded efforts to benchmark bias correction practices for cross-study analyses. It will further be important to weigh the relative cost and benefit of more numerous samples from heterogeneous sources, although it seems likely that cross-study analyses should be considered as an alternative to meta-analyses as more raw data becomes available, methods for metadata harmonization improve, and awareness of the utility of public omics data becomes more widespread.

The concept of reproducible research is central to science. As with other computational endeavors, future analyses of public omics data will continue to benefit from tools and practices that promote reproducibility. Tools such as virtual environments, containers, workflows, and programming notebooks will be used to ensure code is runnable by computers. Thorough methods reporting, transparent methods, open-access code, and verbose comments and docstrings will further help to ensure code is parsable by humans. More dynamic ways of approaching computational problems are likely to save considerable time and effort between projects. The concept of continuous results was recently proposed as a novel research approach [329]. Continuous results is a dynamic idea that considers hypothesis testing as continual process rather than static practice. This shift in focus could be key to driving new insights in a more automated, and therefore reproducible, manner. Thus reproducible research will endure as a useful standard to strive for and formalize to the ultimate benefit of researchers and the greater community alike.

References

[1] C. M. Micheel *et al.*, *Omics-Based Clinical Discovery: Science, Technology, and Applications*, en. National Academies Press (US), Mar. 2012. [Online]. Available: `https://www.ncbi.nlm.nih.gov/books/NBK202165/` (visited on 04/06/2022).

[2] E. S. Lander *et al.*, "Initial sequencing and analysis of the human genome," en, *Nature*, vol. 409, no. 6822, pp. 860–921, Feb. 2001, ISSN: 1476-4687. DOI: `10.1038/35057062`. [Online]. Available: `https://www.nature.com/articles/35057062` (visited on 04/19/2022).

[3] S. K. Maden, R. F. Thompson, K. D. Hansen, and A. Nellore, "Human methylome variation across Infinium 450K data on the Gene Expression Omnibus," *NAR Genomics and Bioinformatics*, vol. 3, no. 2, Jun. 2021, ISSN: 2631-9268. DOI: `10.1093/nargab/lqab025`. [Online]. Available: `https://doi.org/10.1093/nargab/lqab025` (visited on 11/04/2021).

[4] S. K. Maden, B. Walsh, K. Ellrott, K. D. Hansen, R. F. Thompson, and A. Nellore, "Recountmethylation enables flexible analysis of public blood DNA methylation array data," en, bioRxiv, Tech. Rep., May 2022, Section: New Results Type: article, p. 2022.05.19.492680. DOI: `10.1101/2022.05.19.492680`. [Online]. Available: `https://www.biorxiv.org/content/10.1101/2022.05.19.492680v1` (visited on 05/23/2022).

[5] J. K. David, S. K. Maden, M. A. Wood, R. F. Thompson, and A. Nellore, "Retained introns in long RNA-seq reads are not reliably detected in sample-matched short reads," en, bioRxiv, Tech. Rep., Mar. 2022, p. 2022.03.11.484016. DOI: `10.1101/2022.03.11.484016`. [Online]. Available: `https://www.biorxiv.org/content/10.1101/2022.03.11.484016v1` (visited on 04/19/2022).

[6] E. El Marabti and I. Younis, "The Cancer Spliceome: Reprograming of Alternative Splicing in Cancer," *Frontiers in Molecular Biosciences*, vol. 5, 2018, ISSN: 2296-889X. [Online]. Available: `https://www.frontiersin.org/article/10.3389/fmolb.2018.00080` (visited on 04/26/2022).

[7] A. Athar *et al.*, "Arrayexpress update–from bulk to single-cell expression data," *Nucleic acids research*, vol. 47, no. D1, pp. D711–D715, 2019.

[8] R. Edgar, M. Domrachev, and A. E. Lash, "Gene Expression Omnibus: NCBI gene expression and hybridization array data repository," *Nucleic Acids Research*, vol. 30, no. 1, pp. 207–210, Jan. 2002, ISSN: 0305-1048. DOI: `10.1093/nar/30.1.207`. [Online]. Available: `https://doi.org/10.1093/nar/30.1.207` (visited on 02/27/2021).

[9] T. Barrett *et al.*, "Ncbi geo: Archive for functional genomics data sets—update," *Nucleic acids research*, vol. 41, no. D1, pp. D991–D995, 2012.

[10] E. P. Consortium *et al.*, "An integrated encyclopedia of dna elements in the human genome," *Nature*, vol. 489, no. 7414, pp. 57–74, 2012.

[11] C. A. Davis *et al.*, "The encyclopedia of dna elements (encode): Data portal update," *Nucleic acids research*, vol. 46, no. D1, pp. D794–D801, 2018.

[12] J. Barretina *et al.*, "The Cancer Cell Line Encyclopedia enables predictive modelling of anticancer drug sensitivity," eng, *Nature*, vol. 483, no. 7391, pp. 603–607, Mar. 2012, ISSN: 1476-4687. DOI: 10.1038/nature11003.

[13] L. J. Carithers and H. M. Moore, "The Genotype-Tissue Expression (GTEx) Project," eng, *Biopreservation and Biobanking*, vol. 13, no. 5, pp. 307–308, Oct. 2015, ISSN: 1947-5543. DOI: 10.1089/bio.2015.29031.hmm.

[14] M. C. Ward and Y. Gilad, "Cracking the regulatory code," en, *Nature*, vol. 550, no. 7675, pp. 190–191, Oct. 2017, ISSN: 1476-4687. DOI: 10.1038/550190a. [Online]. Available: https://www.nature.com/articles/550190a (visited on 04/19/2022).

[15] J. N. Weinstein *et al.*, "The cancer genome atlas pan-cancer analysis project," *Nature genetics*, vol. 45, no. 10, p. 1113, 2013.

[16] R. Leinonen, H. Sugawara, M. Shumway, and International Nucleotide Sequence Database Collaboration, "The sequence read archive," eng, *Nucleic Acids Research*, vol. 39, no. Database issue, Jan. 2011, ISSN: 1362-4962. DOI: 10.1093/nar/gkq1019.

[17] M. N. Bernstein, A. Doan, C. N. Dewey, and J. Wren, "MetaSRA: Normalized human sample-specific metadata for the Sequence Read Archive," en, *Bioinformatics*, vol. 33, no. 18, pp. 2914–2923, Sep. 2017, ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btx334. [Online]. Available: https://academic.oup.com/bioinformatics/article/33/18/2914/3848915 (visited on 08/30/2018).

[18] *Entrez Programming Utilities Help*, en. National Center for Biotechnology Information (US), 2010. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK25501/ (visited on 04/06/2022).

[19] *Ncbi sra toolkit*. [Online]. Available: https://trace.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?cmd=show&f=software&m=software&s=software (visited on 04/19/2022).

[20] M. S. Abu-Asab *et al.*, "Biomarkers in the Age of Omics: Time for a Systems Biology Approach," *OMICS : a Journal of Integrative Biology*, vol. 15, no. 3, pp. 105–112, Mar. 2011, ISSN: 1536-2310. DOI: 10.1089/omi.2010.0023. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3060038/ (visited on 04/06/2022).

[21] W. J. Locke *et al.*, "DNA Methylation Cancer Biomarkers: Translation to the Clinic," *Frontiers in Genetics*, vol. 10, p. 1150, 2019, ISSN: 1664-8021. DOI: 10.3389/fgene.2019.01150. [Online]. Available: https://www.frontiersin.org/article/10.3389/fgene.2019.01150 (visited on 01/12/2022).

[22] L. Li *et al.*, "DNA methylation in peripheral blood: A potential biomarker for cancer molecular epidemiology," eng, *Journal of Epidemiology*, vol. 22, no. 5, pp. 384–394, 2012, ISSN: 1349-9092. DOI: 10.2188/jea.je20120003.

[23] T. Mikeska and J. M. Craig, "DNA Methylation Biomarkers: Cancer and Beyond," *Genes*, vol. 5, no. 3, pp. 821–864, Sep. 2014, ISSN: 2073-4425. DOI: 10.3390/genes5030821. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4198933/ (visited on 01/12/2022).

[24] T. Willmer, R. Johnson, J. Louw, and C. Pheiffer, “Blood-Based DNA Methylation Biomarkers for Type 2 Diabetes: Potential for Clinical Applications,” *Frontiers in Endocrinology*, vol. 9, p. 744, 2018, ISSN: 1664-2392. DOI: 10.3389/fendo.2018.00744. [Online]. Available: https://www.frontiersin.org/article/10.3389/fendo.2018.00744 (visited on 01/12/2022).

[25] R. S. Gonçalves and M. A. Musen, “The variable quality of metadata about biological samples used in biomedical experiments,” en, *Scientific Data*, vol. 6, no. 1, p. 190 021, Feb. 2019, ISSN: 2052-4463. DOI: 10.1038/sdata.2019.21. [Online]. Available: https://www.nature.com/articles/sdata201921 (visited on 04/06/2022).

[26] C. Wilks *et al.*, “Recount3: Summaries and queries for large-scale RNA-seq expression and splicing,” *Genome Biology*, vol. 22, no. 1, p. 323, Nov. 2021, ISSN: 1474-760X. DOI: 10.1186/s13059-021-02533-6. [Online]. Available: https://doi.org/10.1186/s13059-021-02533-6 (visited on 03/24/2022).

[27] M. J. Aryee *et al.*, “Minfi: A flexible and comprehensive bioconductor package for the analysis of infinium dna methylation microarrays,” *Bioinformatics*, vol. 30, no. 10, pp. 1363–1369, 2014. DOI: 10.1093/bioinformatics/btu049.

[28] R. Lowe, “Marmal-aid - a database for Infinium HumanMethylation450,” *BMC Bioinformatics*, vol. 14, no. 1, p. 359, Dec. 2013, ISSN: 1471-2105. DOI: 10.1186/1471-2105-14-359. [Online]. Available: https://doi.org/10.1186/1471-2105-14-359 (visited on 08/16/2018).

[29] V. S. Malladi *et al.*, “Ontology application and use at the ENCODE DCC,” *Database: The Journal of Biological Databases and Curation*, vol. 2015, Mar. 2015, ISSN: 1758-0463. DOI: 10.1093/database/bav010. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4360730/ (visited on 11/08/2019).

[30] S. Horvath, “DNA methylation age of human tissues and cell types,” *Genome Biology*, vol. 14, no. 10, R115, 2013, ISSN: 1465-6906. DOI: 10.1186/gb-2013-14-10-r115. [Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4015143/ (visited on 07/13/2015).

[31] E. A. Houseman *et al.*, “DNA methylation arrays as surrogate measures of cell mixture distribution,” en, *BMC Bioinformatics*, vol. 13, no. 1, p. 86, May 2012, ISSN: 1471-2105. DOI: 10.1186/1471-2105-13-86. [Online]. Available: http://www.biomedcentral.com/1471-2105/13/86/abstract (visited on 07/06/2015).

[32] E. Rahmani *et al.*, “Genome-wide methylation data mirror ancestry information,” *Epigenetics & Chromatin*, vol. 10, no. 1, p. 1, Jan. 2017, ISSN: 1756-8935. DOI: 10.1186/s13072-016-0108-y. [Online]. Available: https://doi.org/10.1186/s13072-016-0108-y (visited on 08/01/2019).

[33] M. Li *et al.*, “EWAS Atlas: A curated knowledgebase of epigenome-wide association studies,” en, *Nucleic Acids Research*, vol. 47, no. D1, pp. D983–D988, Jan. 2019, ISSN: 0305-1048. DOI: 10.1093/nar/gky1027. [Online]. Available: https://academic.oup.com/nar/article/47/D1/D983/5144953 (visited on 10/29/2019).

[34] Z. Xiong *et al.*, “EWAS Data Hub: A resource of DNA methylation array data and metadata,” en, *Nucleic Acids Research*, 2019. DOI: `10.1093/nar/gkz840`. [Online]. Available: `https://academic.oup.com/nar/advance-article/doi/10.1093/nar/gkz840/5580903` (visited on 10/29/2019).

[35] D. Liu *et al.*, “EWASdb: Epigenome-wide association study database,” eng, *Nucleic Acids Research*, vol. 47, no. D1, pp. D989–D993, Jan. 2019, ISSN: 1362-4962. DOI: `10.1093/nar/gky942`.

[36] R. Li *et al.*, “MethBank 3.0: A database of DNA methylomes across a variety of species,” eng, *Nucleic Acids Research*, vol. 46, no. D1, pp. D288–D295, 2018, ISSN: 1362-4962. DOI: `10.1093/nar/gkx1139`.

[37] Y. Xiong *et al.*, “DiseaseMeth version 2.0: A major expansion and update of the human disease methylation database,” *Nucleic Acids Research*, vol. 45, no. Database issue, pp. D888–D895, Jan. 2017, ISSN: 0305-1048. DOI: `10.1093/nar/gkw1123`. [Online]. Available: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5210584/` (visited on 10/29/2019).

[38] S. K. Maden, R. F. Thompson, K. D. Hansen, and A. Nellore, *Recountmethylation: Access and analyze DNA methylation database compilations*, 2022. DOI: `10.18129/B9.bioc.recountmethylation`. [Online]. Available: `https://bioconductor.org/packages/recountmethylation/` (visited on 04/22/2022).

[39] W. Huber *et al.*, “Orchestrating high-throughput genomic analysis with Bioconductor,” en, *Nature Methods*, vol. 12, no. 2, pp. 115–121, Feb. 2015, ISSN: 1548-7105. DOI: `10.1038/nmeth.3252`. [Online]. Available: `https://www.nature.com/articles/nmeth.3252` (visited on 06/18/2020).

[40] A. C. Frazee, B. Langmead, and J. T. Leek, “ReCount: A multi-experiment resource of analysis-ready RNA-seq gene count datasets,” *BMC Bioinformatics*, vol. 12, no. 1, p. 449, Nov. 2011, ISSN: 1471-2105. DOI: `10.1186/1471-2105-12-449`. [Online]. Available: `https://doi.org/10.1186/1471-2105-12-449` (visited on 03/24/2022).

[41] L. Collado-Torres *et al.*, “Reproducible RNA-seq analysis using recount2,” en, *Nature Biotechnology*, vol. 35, no. 4, pp. 319–321, Apr. 2017, ISSN: 1546-1696. DOI: `10.1038/nbt.3838`. [Online]. Available: `https://www.nature.com/articles/nbt.3838` (visited on 03/24/2022).

[42] A. Razmara *et al.*, “Recount-brain: A curated repository of human brain RNA-seq datasets metadata,” en, bioRxiv, Tech. Rep., Apr. 2019, p. 618 025. DOI: `10.1101/618025`. [Online]. Available: `https://www.biorxiv.org/content/10.1101/618025v1` (visited on 03/24/2022).

[43] L. Collado-Torres *et al.*, *Recount: Explore and download data from the recount project*, 2017. [Online]. Available: `https://bioconductor.org/packages/recount`.

[44] L. Collado-Torres, *Recount3: Explore and download data from the recount3 project*, 2020. [Online]. Available: `https://bioconductor.org/packages/recount3`.

[45] M. Bibikova *et al.*, “High density DNA methylation array with single CpG site resolution,” en, *Genomics*, New Genomic Technologies and Applications, vol. 98, no. 4, pp. 288–295, Oct. 2011, ISSN: 0888-7543. DOI: `10.1016/j.ygeno.2011.07.007`. [Online]. Available: `http://www.sciencedirect.com/science/article/pii/S0888754311001807` (visited on 11/07/2020).

[46] J. Sandoval *et al.*, “Validation of a DNA methylation microarray for 450,000 CpG sites in the human genome,” eng, *Epigenetics*, vol. 6, no. 6, pp. 692–702, Jun. 2011, ISSN: 1559-2308. DOI: 10.4161/epi.6.6.16196.

[47] R. Pidsley *et al.*, “Critical evaluation of the Illumina MethylationEPIC BeadChip microarray for whole-genome DNA methylation profiling,” *Genome Biology*, vol. 17, Oct. 2016, ISSN: 1474-7596. DOI: 10.1186/s13059-016-1066-1. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5055731/ (visited on 04/19/2019).

[48] F. Asenius *et al.*, “The DNA methylome of human sperm is distinct from blood with little evidence for tissue-consistent obesity associations,” en, *PLOS Genetics*, vol. 16, no. 10, e1009035, Oct. 2020, ISSN: 1553-7404. DOI: 10.1371/journal.pgen.1009035. [Online]. Available: https://journals.plos.org/plosgenetics/article?id=10.1371/journal.pgen.1009035 (visited on 02/02/2021).

[49] K. L. Haftorn *et al.*, “An EPIC predictor of gestational age and its application to newborns conceived by assisted reproductive technologies,” *Clinical Epigenetics*, vol. 13, no. 1, p. 82, Apr. 2021, ISSN: 1868-7083. DOI: 10.1186/s13148-021-01055-z. [Online]. Available: https://doi.org/10.1186/s13148-021-01055-z (visited on 11/02/2021).

[50] “Diagnostic Utility of Genome-wide DNA Methylation Testing in Genetically Unsolved Individuals with Suspected Hereditary Conditions,” eng, *American Journal of Human Genetics*, vol. 104, no. 4, pp. 685–700, Apr. 2019, ISSN: 1537-6605. DOI: 10.1016/j.ajhg.2019.03.008.

[51] “Clinical epigenomics: Genome-wide DNA methylation analysis for the diagnosis of Mendelian disorders,” en, *Genetics in Medicine*, vol. 23, no. 6, pp. 1065–1074, Jun. 2021, ISSN: 1530-0366. DOI: 10.1038/s41436-020-01096-4. [Online]. Available: https://www.nature.com/articles/s41436-020-01096-4 (visited on 04/18/2022).

[52] M. Krassowski, V. Das, S. K. Sahu, and B. B. Misra, “State of the Field in Multi-Omics Research: From Computational Needs to Data Mining and Sharing,” *Frontiers in Genetics*, vol. 11, p. 610 798, Dec. 2020, ISSN: 1664-8021. DOI: 10.3389/fgene.2020.610798. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7758509/ (visited on 04/06/2022).

[53] G. Drouard *et al.*, “Multi-Omics Integration in a Twin Cohort and Predictive Modeling of Blood Pressure Values,” *OMICS : a Journal of Integrative Biology*, vol. 26, no. 3, pp. 130–141, Mar. 2022, ISSN: 1536-2310. DOI: 10.1089/omi.2021.0201. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8978565/ (visited on 04/06/2022).

[54] M. D. Luecken *et al.*, “Benchmarking atlas-level data integration in single-cell genomics,” *Nature Methods*, vol. 19, no. 1, pp. 41–50, 2022, ISSN: 1548-7091. DOI: 10.1038/s41592-021-01336-8. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8748196/ (visited on 04/24/2022).

[55] S. L. Amarasinghe, S. Su, X. Dong, L. Zappia, M. E. Ritchie, and Q. Gouil, “Opportunities and challenges in long-read sequencing data analysis,” *Genome Biology*, vol. 21, p. 30, Feb. 2020, ISSN: 1474-7596. DOI:

10.1186/s13059-020-1935-5. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7006217/ (visited on 04/20/2022).

[56] Z. Zeng, Y. Li, Y. Li, and Y. Luo, "Statistical and machine learning methods for spatially resolved transcriptomics data analysis," *Genome Biology*, vol. 23, no. 1, p. 83, Mar. 2022, ISSN: 1474-760X. DOI: 10.1186/s13059-022-02653-7. [Online]. Available: https://doi.org/10.1186/s13059-022-02653-7 (visited on 04/14/2022).

[57] N. Schaduangrat, S. Lampa, S. Simeon, M. P. Gleeson, O. Spjuth, and C. Nantasenamat, "Towards reproducible computational drug discovery," *Journal of Cheminformatics*, vol. 12, p. 9, Jan. 2020, ISSN: 1758-2946. DOI: 10.1186/s13321-020-0408-x. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6988305/ (visited on 04/07/2022).

[58] J. M. Alston and J. A. Rick, "A Beginner's Guide to Conducting Reproducible Research," en, *The Bulletin of the Ecological Society of America*, vol. 102, no. 2, e01801, 2021, ISSN: 2327-6096. DOI: 10.1002/bes2.1801. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/bes2.1801 (visited on 04/07/2022).

[59] J. Leipzig, D. Nüst, C. T. Hoyt, K. Ram, and J. Greenberg, "The role of metadata in reproducible computational research," *Patterns*, vol. 2, no. 9, p. 100 322, Sep. 2021, ISSN: 2666-3899. DOI: 10.1016/j.patter.2021.100322. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8441584/ (visited on 04/07/2022).

[60] S. Kanwal, F. Z. Khan, A. Lonie, and R. O. Sinnott, "Investigating reproducibility and tracking provenance – A genomic workflow case study," *BMC Bioinformatics*, vol. 18, p. 337, Jul. 2017, ISSN: 1471-2105. DOI: 10.1186/s12859-017-1747-0. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5508699/ (visited on 04/24/2022).

[61] S. Maden, R. Thompson, K. Hansen, and A. Nellore, *Recountmethylation_instance*, 2022. [Online]. Available: https://github.com/metamaden/recountmethylation%5C_instance.

[62] M. Bostick, J. K. Kim, P.-O. Estève, A. Clark, S. Pradhan, and S. E. Jacobsen, "UHRF1 plays a role in maintaining DNA methylation in mammalian cells," eng, *Science (New York, N.Y.)*, vol. 317, no. 5845, pp. 1760–1764, Sep. 2007, ISSN: 1095-9203. DOI: 10.1126/science.1147939.

[63] E. Li and Y. Zhang, "DNA Methylation in Mammals," *Cold Spring Harbor Perspectives in Biology*, vol. 6, no. 5, a019133, May 2014, ISSN: 1943-0264. DOI: 10.1101/cshperspect.a019133. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3996472/ (visited on 04/23/2022).

[64] M. Damelin and T. H. Bestor, "Biological Functions of DNA Methyltransferase 1 Require Its Methyltransferase Activity," *Molecular and Cellular Biology*, vol. 27, no. 11, pp. 3891–3899, Jun. 2007, ISSN: 0270-7306. DOI: 10.1128/MCB.00036-07. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1900033/ (visited on 04/23/2022).

[65] L. Gao *et al.*, "Comprehensive structure-function characterization of DNMT3B and DNMT3A reveals distinctive de novo DNA methylation mechanisms," *Nature Communications*, vol. 11, p. 3355, Jul. 2020, ISSN: 2041-1723. DOI: 10.1038/s41467-020-17109-4. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7335073/ (visited on 04/23/2022).

[66] J. Charlton *et al.*, "TETs compete with DNMT3 activity in pluripotent cells at thousands of methylated somatic enhancers," *Nature genetics*, vol. 52, no. 8, pp. 819–827, Aug. 2020, ISSN: 1061-4036. DOI: 10.1038/s41588-020-0639-9. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7415576/ (visited on 04/23/2022).

[67] F. Neri *et al.*, "Intragenic DNA methylation prevents spurious transcription initiation," eng, *Nature*, vol. 543, no. 7643, pp. 72–77, Mar. 2017, ISSN: 1476-4687. DOI: 10.1038/nature21373.

[68] G. Lev Maor, A. Yearim, and G. Ast, "The alternative role of DNA methylation in splicing regulation," eng, *Trends in genetics: TIG*, vol. 31, no. 5, pp. 274–280, May 2015, ISSN: 0168-9525. DOI: 10.1016/j.tig.2015.03.002.

[69] P. A. Jones, "Functions of DNA methylation: Islands, start sites, gene bodies and beyond," en, *Nature Reviews Genetics*, vol. 13, no. 7, pp. 484–492, Jul. 2012, ISSN: 1471-0064. DOI: 10.1038/nrg3230. [Online]. Available: https://www.nature.com/articles/nrg3230 (visited on 06/18/2019).

[70] J. T. Bell *et al.*, "DNA methylation patterns associate with genetic and gene expression variation in HapMap cell lines," eng, *Genome Biology*, vol. 12, no. 1, R10, 2011, ISSN: 1474-760X. DOI: 10.1186/gb-2011-12-1-r10.

[71] Z. Wang *et al.*, "Complex impact of DNA methylation on transcriptional dysregulation across 22 human cancer types," en, *Nucleic Acids Research*, vol. 48, no. 5, pp. 2287–2302, Mar. 2020, ISSN: 0305-1048. DOI: 10.1093/nar/gkaa041. [Online]. Available: https://academic.oup.com/nar/article/48/5/2287/5718258 (visited on 10/26/2020).

[72] R. A. Irizarry *et al.*, "The human colon cancer methylome shows similar hypo- and hypermethylation at conserved tissue-specific CpG island shores," en, *Nature Genetics*, vol. 41, no. 2, pp. 178–186, Feb. 2009, ISSN: 1546-1718. DOI: 10.1038/ng.298. [Online]. Available: https://www.nature.com/articles/ng.298 (visited on 10/02/2020).

[73] M. Gardiner-Garden and M. Frommer, "CpG islands in vertebrate genomes," eng, *Journal of Molecular Biology*, vol. 196, no. 2, pp. 261–282, Jul. 1987, ISSN: 0022-2836. DOI: 10.1016/0022-2836(87)90689-9.

[74] Illumina, *Illumina Genome Studio Methylation Module v1.8*, Nov. 2010. [Online]. Available: https://support.illumina.com/content/dam/illumina-support/documents/documentation/software_documentation/genomestudio/genomestudio-2011-1/genomestudio-methylation-v1-8-user-guide-11319130-b.pdf (visited on 08/20/2019).

[75] A. M. Deaton and A. Bird, "CpG islands and the regulation of transcription," *Genes & Development*, vol. 25, no. 10, pp. 1010–1022, May 2011, ISSN: 0890-9369. DOI: 10.1101/gad.2037511. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3093116/ (visited on 11/23/2021).

[76] E. G. Luebeck *et al.*, "Identification of a key role of widespread epigenetic drift in Barrett's esophagus and esophageal adenocarcinoma," *Clinical Epigenetics*, vol. 9, no. 1, p. 113, Oct. 2017, ISSN: 1868-7083. DOI: 10.1186/s13148-017-0409-4. [Online]. Available: https://doi.org/10.1186/s13148-017-0409-4 (visited on 04/18/2022).

[77] D. Takai and P. A. Jones, "Comprehensive analysis of CpG islands in human chromosomes 21 and 22," eng, *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. 6, pp. 3740–3745, Mar. 2002, ISSN: 0027-8424. DOI: 10.1073/pnas.052410099.

[78] S. F. Wolf, D. J. Jolly, K. D. Lunnen, T. Friedmann, and B. R. Migeon, "Methylation of the hypoxanthine phosphoribosyltransferase locus on the human X chromosome: Implications for X-chromosome inactivation," eng, *Proceedings of the National Academy of Sciences of the United States of America*, vol. 81, no. 9, pp. 2806–2810, May 1984, ISSN: 0027-8424. DOI: 10.1073/pnas.81.9.2806.

[79] N. Brockdorff and B. M. Turner, "Dosage Compensation in Mammals," *Cold Spring Harbor Perspectives in Biology*, vol. 7, no. 3, a019406, Mar. 2015, ISSN: 1943-0264. DOI: 10.1101/cshperspect.a019406. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4355265/ (visited on 04/23/2022).

[80] C. Walsh, J. Chaillet, and T. Bestor, "Transcription of IAP endogenous retroviruses is constrained by cytosine methylation," *Nature Genetics*, vol. 20, no. 2, pp. 116–117, Oct. 1998, ISSN: 1061-4036. DOI: 10.1038/2413.

[81] D. Jähner *et al.*, "De novo methylation and expression of retroviral genomes during mouse embryogenesis," en, *Nature*, vol. 298, no. 5875, pp. 623–628, Aug. 1982, ISSN: 1476-4687. DOI: 10.1038/298623a0. [Online]. Available: https://www.nature.com/articles/298623a0 (visited on 04/23/2022).

[82] O. Bogdanović and R. Lister, "DNA methylation and the preservation of cell identity," eng, *Current Opinion in Genetics & Development*, vol. 46, pp. 9–14, Oct. 2017, ISSN: 1879-0380. DOI: 10.1016/j.gde.2017.06.007.

[83] Y. Zeng and T. Chen, "DNA Methylation Reprogramming during Mammalian Development," en, *Genes*, vol. 10, no. 4, p. 257, Apr. 2019, ISSN: 2073-4425. DOI: 10.3390/genes10040257. [Online]. Available: https://www.mdpi.com/2073-4425/10/4/257 (visited on 04/23/2022).

[84] C. Ambrosi, M. Manzo, and T. Baubec, "Dynamics and Context-Dependent Roles of DNA Methylation," eng, *Journal of Molecular Biology*, vol. 429, no. 10, pp. 1459–1475, May 2017, ISSN: 1089-8638. DOI: 10.1016/j.jmb.2017.02.008.

[85] C. Tomasetti and B. Vogelstein, "Variation in cancer risk among tissues can be explained by the number of stem cell divisions," *Science (New York, N.Y.)*, vol. 347, no. 6217, pp. 78–81, Jan. 2015, ISSN: 0036-8075. DOI:

10.1126/science.1260825. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4446723/ (visited on 04/23/2022).

[86] Z. Yang *et al.*, “Correlation of an epigenetic mitotic clock with cancer risk,” *Genome Biology*, vol. 17, Oct. 2016, ISSN: 1474-7596. DOI: 10.1186/s13059-016-1064-3. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5046977/ (visited on 03/15/2018).

[87] C. G. Bell *et al.*, “DNA methylation aging clocks: Challenges and recommendations,” *Genome Biology*, vol. 20, p. 249, Nov. 2019, ISSN: 1474-7596. DOI: 10.1186/s13059-019-1824-y. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6876109/ (visited on 04/23/2022).

[88] E. M. Michalak, M. L. Burr, A. J. Bannister, and M. A. Dawson, “The roles of DNA, RNA and histone methylation in ageing and cancer,” eng, *Nature Reviews. Molecular Cell Biology*, vol. 20, no. 10, pp. 573–589, Oct. 2019, ISSN: 1471-0080. DOI: 10.1038/s41580-019-0143-1.

[89] K. Bacos *et al.*, “Blood-based biomarkers of age-associated epigenetic changes in human islets associate with insulin secretion and diabetes,” *Nature Communications*, vol. 7, p. 11 089, Mar. 2016, ISSN: 2041-1723. DOI: 10.1038/ncomms11089. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4821875/ (visited on 01/12/2022).

[90] T. Dayeh *et al.*, “DNA methylation of loci within ABCG1 and PHOSPHO1 in blood DNA is associated with future type 2 diabetes risk,” *Epigenetics*, vol. 11, no. 7, pp. 482–488, May 2016, ISSN: 1559-2294. DOI: 10.1080/15592294.2016.1178418. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4939923/ (visited on 01/12/2022).

[91] M. Samblas, F. I. Milagro, and A. Martínez, “DNA methylation markers in obesity, metabolic syndrome, and weight loss,” *Epigenetics*, vol. 14, no. 5, pp. 421–444, Mar. 2019, ISSN: 1559-2294. DOI: 10.1080/15592294.2019.1595297. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6557553/ (visited on 01/12/2022).

[92] M. Yu *et al.*, “Subtypes of Barrett’s oesophagus and oesophageal adenocarcinoma based on genome-wide methylation analysis,” eng, *Gut*, Jun. 2018, ISSN: 1468-3288. DOI: 10.1136/gutjnl-2017-314544.

[93] Z. Guan, H. Yu, K. Cuk, Y. Zhang, and H. Brenner, “Whole-Blood DNA Methylation Markers in Early Detection of Breast Cancer: A Systematic Literature Review,” en, *Cancer Epidemiology and Prevention Biomarkers*, vol. 28, no. 3, pp. 496–505, Mar. 2019, ISSN: 1055-9965, 1538-7755. DOI: 10.1158/1055-9965.EPI-18-0378. [Online]. Available: https://cebp.aacrjournals.org/content/28/3/496 (visited on 01/12/2022).

[94] C. S. Danstrup, M. Marcussen, I. S. Pedersen, H. Jacobsen, K. Dybkær, and M. Gaihede, “DNA methylation biomarkers in peripheral blood of patients with head and neck squamous cell carcinomas. A systematic review,” en, *PLOS ONE*, vol. 15, no. 12, e0244101, Dec. 2020, ISSN: 1932-6203. DOI: 10.1371/journal.pone.0244101. [Online]. Available: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0244101 (visited on 01/12/2022).

[95] S. D. Henriksen and O. Thorlacius-Ussing, "Cell-Free DNA Methylation as Blood-Based Biomarkers for Pancreatic Adenocarcinoma—A Literature Update," en, *Epigenomes*, vol. 5, no. 2, p. 8, Jun. 2021. DOI: 10.3390/epigenomes5020008. [Online]. Available: https://www.mdpi.com/2075-4655/5/2/8 (visited on 01/12/2022).

[96] S. Ø. Jensen *et al.*, "Novel DNA methylation biomarkers show high sensitivity and specificity for blood-based detection of colorectal cancer—a clinical biomarker discovery and validation study," *Clinical Epigenetics*, vol. 11, no. 1, p. 158, Nov. 2019, ISSN: 1868-7083. DOI: 10.1186/s13148-019-0757-3. [Online]. Available: https://doi.org/10.1186/s13148-019-0757-3 (visited on 01/12/2022).

[97] L. Dong and H. Ren, "Blood-based DNA Methylation Biomarkers for Early Detection of Colorectal Cancer," *Journal of proteomics & bioinformatics*, vol. 11, no. 6, pp. 120–126, 2018, ISSN: 0974-276X. DOI: 10.4172/jpb.1000477. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6054487/ (visited on 01/12/2022).

[98] M. Alizadeh-Sedigh, M. S. Fazeli, H. Mahmoodzadeh, S. B. Sharif, and L. Teimoori-Toolabi, "Methylation of FBN1, SPG20, ITF2, RUNX3, SNCA, MLH1, and SEPT9 genes in circulating cell-free DNA as biomarkers of colorectal cancer," eng, *Cancer Biomarkers: Section A of Disease Markers*, Dec. 2021, ISSN: 1875-8592. DOI: 10.3233/CBM-210315.

[99] W.-H. Lin *et al.*, "Circulating tumor DNA methylation marker MYO1-G for diagnosis and monitoring of colorectal cancer," eng, *Clinical Epigenetics*, vol. 13, no. 1, p. 232, Dec. 2021, ISSN: 1868-7083. DOI: 10.1186/s13148-021-01216-0.

[100] "Discovery of methylated circulating DNA biomarkers for comprehensive non-invasive monitoring of treatment response in metastatic colorectal cancer," vol. 67, pp. 1995–2005, 2018, ISSN: 0017-5749. DOI: 10.1136/gutjnl-2016-313372. [Online]. Available: https://gut.bmj.com/content/67/11/1995.

[101] Y. N. Lamb and S. Dhillon, "Epi proColon® 2.0 CE: A Blood-Based Screening Test for Colorectal Cancer," en, *Molecular Diagnosis & Therapy*, vol. 21, no. 2, pp. 225–232, Apr. 2017, ISSN: 1179-2000. DOI: 10.1007/s40291-017-0259-y. [Online]. Available: https://doi.org/10.1007/s40291-017-0259-y (visited on 04/07/2022).

[102] T. Nagasaka *et al.*, "Analysis of fecal DNA methylation to detect gastrointestinal neoplasia," eng, *Journal of the National Cancer Institute*, vol. 101, no. 18, pp. 1244–1258, Sep. 2009, ISSN: 1460-2105. DOI: 10.1093/jnci/djp265.

[103] Y. Hashimoto, T. J. Zumwalt, and A. Goel, "DNA methylation patterns as noninvasive biomarkers and targets of epigenetic therapies in colorectal cancer," *Epigenomics*, vol. 8, no. 5, pp. 685–703, May 2016, ISSN: 1750-1911. DOI: 10.2217/epi-2015-0013. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4928499/ (visited on 04/07/2022).

[104] B. Planterose Jiménez, M. Kayser, and A. Vidaki, "Revisiting genetic artifacts on DNA methylation microarrays exposes novel biological implications," *Genome Biology*, vol. 22, no. 1, p. 274, Sep. 2021, ISSN: 1474-760X.

DOI: 10.1186/s13059-021-02484-y. [Online]. Available: https://doi.org/10.1186/s13059-021-02484-y (visited on 04/18/2022).

[105] H. Pohl, B. Sirovich, and H. G. Welch, "Esophageal adenocarcinoma incidence: Are we reaching the peak?" eng, *Cancer Epidemiology, Biomarkers & Prevention: A Publication of the American Association for Cancer Research, Cosponsored by the American Society of Preventive Oncology*, vol. 19, no. 6, pp. 1468–1470, Jun. 2010, ISSN: 1538-7755. DOI: 10.1158/1055-9965.EPI-10-0012.

[106] M. F. Buas and T. L. Vaughan, "Epidemiology and risk factors for gastroesophageal junction tumors: Understanding the rising incidence of this disease," eng, *Seminars in Radiation Oncology*, vol. 23, no. 1, pp. 3–9, Jan. 2013, ISSN: 1532-9461. DOI: 10.1016/j.semradonc.2012.09.008.

[107] C.-A. J. Ong, P. Lao-Sirieix, and R. C. Fitzgerald, "Biomarkers in Barrett's esophagus and esophageal adenocarcinoma: Predictors of progression and prognosis," *World Journal of Gastroenterology : WJG*, vol. 16, no. 45, pp. 5669–5681, Dec. 2010, ISSN: 1007-9327. DOI: 10.3748/wjg.v16.i45.5669. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2997982/ (visited on 04/18/2022).

[108] H. Chettouh *et al.*, "Methylation panel is a diagnostic biomarker for Barrett's oesophagus in endoscopic biopsies and non-endoscopic cytology specimens," eng, *Gut*, vol. 67, no. 11, pp. 1942–1949, Nov. 2018, ISSN: 1468-3288. DOI: 10.1136/gutjnl-2017-314026.

[109] M. A. Alvi *et al.*, "DNA methylation as an adjunct to histopathology to detect prevalent, inconspicuous dysplasia and early-stage neoplasia in Barrett's esophagus," eng, *Clinical Cancer Research: An Official Journal of the American Association for Cancer Research*, vol. 19, no. 4, pp. 878–888, Feb. 2013, ISSN: 1557-3265. DOI: 10.1158/1078-0432.CCR-12-2880.

[110] Z. Wang *et al.*, "Methylation Biomarker Panel Performance in EsophaCap Cytology Samples for Diagnosing Barrett's Esophagus: A Prospective Validation Study," eng, *Clinical Cancer Research: An Official Journal of the American Association for Cancer Research*, vol. 25, no. 7, pp. 2127–2135, Apr. 2019, ISSN: 1557-3265. DOI: 10.1158/1078-0432.CCR-18-3696.

[111] P. G. Iyer *et al.*, "Highly Discriminant Methylated DNA Markers for the Non-endoscopic Detection of Barrett's Esophagus," eng, *The American Journal of Gastroenterology*, vol. 113, no. 8, pp. 1156–1166, Aug. 2018, ISSN: 1572-0241. DOI: 10.1038/s41395-018-0107-7.

[112] H. R. Moinova *et al.*, "Identifying DNA methylation biomarkers for non-endoscopic detection of Barrett's esophagus," eng, *Science Translational Medicine*, vol. 10, no. 424, eaao5848, Jan. 2018, ISSN: 1946-6242. DOI: 10.1126/scitranslmed.aao5848.

[113] G. Hannum *et al.*, "Genome-wide Methylation Profiles Reveal Quantitative Views of Human Aging Rates," English, *Molecular Cell*, vol. 49, no. 2, pp. 359–367, Jan. 2013, ISSN: 1097-2765. DOI: 10.1016/j.molcel.2012.10.016. [Online]. Available: https://www.cell.com/molecular-cell/abstract/S1097-2765(12)00893-3 (visited on 02/07/2020).

[114] T. Wang *et al.*, "Dysfunctional epigenetic aging of the normal colon and colorectal cancer risk," *Clinical Epigenetics*, vol. 12, no. 1, p. 5, Jan. 2020, ISSN: 1868-7083. DOI: 10.1186/s13148-019-0801-3. [Online]. Available: https://doi.org/10.1186/s13148-019-0801-3 (visited on 02/07/2020).

[115] A. K. Knight *et al.*, "An epigenetic clock for gestational age at birth based on blood methylation data," *Genome Biology*, vol. 17, no. 1, p. 206, Oct. 2016, ISSN: 1474-760X. DOI: 10.1186/s13059-016-1068-z. [Online]. Available: https://doi.org/10.1186/s13059-016-1068-z (visited on 03/01/2022).

[116] J. Bohlin *et al.*, "Prediction of gestational age based on genome-wide differentially methylated regions," *Genome Biology*, vol. 17, no. 1, p. 207, Oct. 2016, ISSN: 1474-760X. DOI: 10.1186/s13059-016-1063-4. [Online]. Available: https://doi.org/10.1186/s13059-016-1063-4 (visited on 03/01/2022).

[117] Y. Lee *et al.*, "Placental epigenetic clocks: Estimating gestational age using placental DNA methylation levels," *Aging (Albany NY)*, vol. 11, no. 12, pp. 4238–4253, Jun. 2019, ISSN: 1945-4589. DOI: 10.18632/aging.102049. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6628997/ (visited on 03/01/2022).

[118] R. Fung *et al.*, "Achieving accurate estimates of fetal gestational age and personalised predictions of fetal growth based on data from an international prospective cohort study: A population-based machine learning study," *The Lancet. Digital Health*, vol. 2, no. 7, e368–e375, Jun. 2020, ISSN: 2589-7500. DOI: 10.1016/S2589-7500(20)30131-X. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7323599/ (visited on 03/10/2022).

[119] S. K. Merid *et al.*, "Epigenome-wide meta-analysis of blood DNA methylation in newborns and children identifies numerous loci related to gestational age," *Genome Medicine*, vol. 12, Mar. 2020, ISSN: 1756-994X. DOI: 10.1186/s13073-020-0716-9. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7050134/ (visited on 08/25/2020).

[120] Z. V. Ogneva, A. S. Dubrovina, and K. V. Kiselev, "Age-associated alterations in DNA methylation and expression of methyltransferase and demethylase genes in Arabidopsis thaliana," en, *Biologia Plantarum*, vol. 60, no. 4, pp. 628–634, Dec. 2016, ISSN: 1573-8264. DOI: 10.1007/s10535-016-0638-y. [Online]. Available: https://doi.org/10.1007/s10535-016-0638-y (visited on 04/18/2022).

[121] G. S. Wilkinson *et al.*, "DNA methylation predicts age and provides insight into exceptional longevity of bats," en, *Nature Communications*, vol. 12, no. 1, p. 1615, Mar. 2021, ISSN: 2041-1723. DOI: 10.1038/s41467-021-21900-2. [Online]. Available: https://www.nature.com/articles/s41467-021-21900-2 (visited on 04/18/2022).

[122] M. Frommer *et al.*, "A genomic sequencing protocol that yields a positive display of 5-methylcytosine residues in individual DNA strands.," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 89, no. 5, pp. 1827–1831, Mar. 1992, ISSN: 0027-8424. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC48546/ (visited on 04/25/2022).

[123] C. A. Leontiou, M. D. Hadjidaniel, P. Mina, P. Antoniou, M. Ioannides, and P. C. Patsalis, "Bisulfite Conversion of DNA: Performance Comparison of Different Kits and Methylation Quantitation of Epigenetic Biomarkers that Have the Potential to Be Used in Non-Invasive Prenatal Testing," *PLoS ONE*, vol. 10, no. 8, e0135058, Aug. 2015, ISSN: 1932-6203. DOI: 10.1371/journal.pone.0135058. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4527772/ (visited on 04/25/2022).

[124] F. K. Crary-Dooley, M. E. Tam, K. W. Dunaway, I. Hertz-Picciotto, R. J. Schmidt, and J. M. LaSalle, "A comparison of existing global DNA methylation assays to low-coverage whole-genome bisulfite sequencing for epidemiological studies," *Epigenetics*, vol. 12, no. 3, pp. 206–214, Jan. 2017, ISSN: 1559-2294. DOI: 10.1080/15592294.2016.1276680. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5406214/ (visited on 04/18/2022).

[125] S. G. Zhao *et al.*, "The DNA methylation landscape of advanced prostate cancer," eng, *Nature Genetics*, vol. 52, no. 8, pp. 778–789, Aug. 2020, ISSN: 1546-1718. DOI: 10.1038/s41588-020-0648-8.

[126] M. J. Ziller, K. D. Hansen, A. Meissner, and M. J. Aryee, "Coverage recommendations for methylation analysis by whole-genome bisulfite sequencing," eng, *Nature Methods*, vol. 12, no. 3, 230–232, 1 p following 232, Mar. 2015, ISSN: 1548-7105. DOI: 10.1038/nmeth.3152.

[127] Q. Li, P. J. Hermanson, and N. M. Springer, "Detection of DNA Methylation by Whole-Genome Bisulfite Sequencing," eng, *Methods in Molecular Biology (Clifton, N.J.)*, vol. 1676, pp. 185–196, 2018, ISSN: 1940-6029. DOI: 10.1007/978-1-4939-7315-6_11.

[128] J. Guo *et al.*, "Dysregulation of CXCL14 promotes malignant phenotypes of esophageal squamous carcinoma cells via regulating SRC and EGFR signaling," eng, *Biochemical and Biophysical Research Communications*, vol. 609, pp. 75–83, Apr. 2022, ISSN: 1090-2104. DOI: 10.1016/j.bbrc.2022.03.144.

[129] M. Good *et al.*, "Selective hypermethylation is evident in small intestine samples from infants with necrotizing enterocolitis," eng, *Clinical Epigenetics*, vol. 14, no. 1, p. 49, Apr. 2022, ISSN: 1868-7083. DOI: 10.1186/s13148-022-01266-y.

[130] J. Tost and I. G. Gut, "DNA methylation analysis by pyrosequencing," eng, *Nature Protocols*, vol. 2, no. 9, pp. 2265–2275, 2007, ISSN: 1750-2799. DOI: 10.1038/nprot.2007.314.

[131] C. A. Eads *et al.*, "MethyLight: A high-throughput assay to measure DNA methylation," *Nucleic Acids Research*, vol. 28, no. 8, e32, Apr. 2000, ISSN: 0305-1048. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC102836/ (visited on 04/18/2022).

[132] H.-C. Wu, Q. Wang, L. Delgado-Cruzata, R. M. Santella, and M. B. Terry, "Genomic Methylation Changes Over Time in Peripheral Blood Mononuclear Cell DNA: Differences by Assay Type and Baseline Values," *Cancer epidemiology, biomarkers & prevention : a publication of the American Association for Cancer Research, cosponsored by the American Society of Preventive Oncology*, vol. 21, no. 8, pp. 1314–1318, Aug. 2012, ISSN: 1055-9965. DOI: 10.1158/1055-9965.EPI-12-0300. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4032622/ (visited on 04/18/2022).

[133] A. Martisova, J. Holcakova, N. Izadi, R. Sebuyoya, R. Hrstka, and M. Bartosik, “DNA Methylation in Solid Tumors: Functions and Methods of Detection,” *International Journal of Molecular Sciences*, vol. 22, no. 8, p. 4247, Apr. 2021, ISSN: 1422-0067. DOI: 10.3390/ijms22084247. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8073724/ (visited on 04/18/2022).

[134] S. Li and T. O. Tollefsbol, “DNA methylation methods: Global DNA methylation and methylomic analyses.,” *Methods (San Diego, Calif.)*, vol. 187, pp. 28–43, Mar. 2021, ISSN: 1046-2023. DOI: 10.1016/j.ymeth.2020.10.002. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7914139/ (visited on 04/18/2022).

[135] R. Halabian, V. Arshad, A. Ahmadi, P. Saeedi, S. Azimzadeh Jamalkandi, and M. R. Alivand, “Laboratory methods to decipher epigenetic signatures: A comparative review,” *Cellular & Molecular Biology Letters*, vol. 26, p. 46, Nov. 2021, ISSN: 1425-8153. DOI: 10.1186/s11658-021-00290-9. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8582164/ (visited on 04/18/2022).

[136] K. D. Siegmund, “Statistical approaches for the analysis of DNA methylation microarray data,” eng, *Human Genetics*, vol. 129, no. 6, pp. 585–595, Jun. 2011, ISSN: 1432-1203. DOI: 10.1007/s00439-011-0993-x.

[137] Y.-a. Chen *et al.*, “Discovery of cross-reactive probes and polymorphic CpGs in the Illumina Infinium HumanMethylation450 microarray,” *Epigenetics*, vol. 8, no. 2, pp. 203–209, Feb. 2013, ISSN: 1559-2294. DOI: 10.4161/epi.23470. [Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3592906/ (visited on 07/11/2015).

[138] M. Inoshita *et al.*, “Sex differences of leukocytes DNA methylation adjusted for estimated cellular proportions,” *Biology of Sex Differences*, vol. 6, no. 1, p. 11, Jun. 2015, ISSN: 2042-6410. DOI: 10.1186/s13293-015-0029-7. [Online]. Available: https://doi.org/10.1186/s13293-015-0029-7 (visited on 07/08/2021).

[139] L. A. Salas *et al.*, “Enhanced cell deconvolution of peripheral blood using DNA methylation for high-resolution immune profiling,” en, *Nature Communications*, vol. 13, no. 1, p. 761, Feb. 2022, ISSN: 2041-1723. DOI: 10.1038/s41467-021-27864-7. [Online]. Available: https://www.nature.com/articles/s41467-021-27864-7 (visited on 02/22/2022).

[140] T. J. Triche, D. J. Weisenberger, D. Van Den Berg, P. W. Laird, and K. D. Siegmund, “Low-level processing of Illumina Infinium DNA Methylation BeadArrays,” en, *Nucleic Acids Research*, vol. 41, no. 7, e90–e90, Apr. 2013, ISSN: 0305-1048. DOI: 10.1093/nar/gkt090. [Online]. Available: https://academic.oup.com/nar/article/41/7/e90/1070878 (visited on 02/15/2019).

[141] J.-P. Fortin *et al.*, “Functional normalization of 450k methylation array data improves replication in large cancer studies,” *Genome biology*, vol. 15, no. 11, p. 503, 2014.

[142] L. M. McEwen *et al.*, “Systematic evaluation of DNA methylation age estimation with common preprocessing methods and the Infinium MethylationEPIC BeadChip array,” *Clinical Epigenetics*, vol. 10, p. 123, Oct. 2018, ISSN: 1868-7075. DOI: 10.1186/s13148-018-0556-2. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6192219/ (visited on 04/18/2022).

[143] Z. Xu, L. Niu, and J. A. Taylor, "The ENmix DNA methylation analysis pipeline for Illumina BeadChip and comparisons with seven other preprocessing pipelines," *Clinical Epigenetics*, vol. 13, p. 216, Dec. 2021, ISSN: 1868-7075. DOI: 10.1186/s13148-021-01207-1. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8662917/ (visited on 04/18/2022).

[144] J. Liu and K. D. Siegmund, "An evaluation of processing methods for HumanMethylation450 BeadChip data," eng, *BMC genomics*, vol. 17, p. 469, Jun. 2016, ISSN: 1471-2164. DOI: 10.1186/s12864-016-2819-7.

[145] M. Knoll, J. Debus, and A. Abdollahi, "cnAnalysis450k: An R package for comparative analysis of 450k/EPIC Illumina methylation array derived copy number data," eng, *Bioinformatics (Oxford, England)*, vol. 33, no. 15, pp. 2266–2272, Aug. 2017, ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btx156.

[146] C. Jiao *et al.*, "Positional effects revealed in Illumina methylation array and the impact on analysis," eng, *Epigenomics*, vol. 10, no. 5, pp. 643–659, May 2018, ISSN: 1750-192X. DOI: 10.2217/epi-2017-0105.

[147] S. Behjati and P. S. Tarpey, "What is next generation sequencing?" *Archives of Disease in Childhood. Education and Practice Edition*, vol. 98, no. 6, pp. 236–238, Dec. 2013, ISSN: 1743-0585. DOI: 10.1136/archdischild-2013-304340. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3841808/ (visited on 04/23/2022).

[148] R. Kamps *et al.*, "Next-Generation Sequencing in Oncology: Genetic Diagnosis, Risk Prediction and Cancer Classification," *International Journal of Molecular Sciences*, vol. 18, no. 2, p. 308, Jan. 2017, ISSN: 1422-0067. DOI: 10.3390/ijms18020308. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5343844/ (visited on 04/23/2022).

[149] Y. O. Alekseyev *et al.*, "A Next-Generation Sequencing Primer—How Does It Work and What Can It Do?" *Academic Pathology*, vol. 5, p. 2 374 289 518 766 521, May 2018, ISSN: 2374-2895. DOI: 10.1177/2374289518766521. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5944141/ (visited on 04/23/2022).

[150] S. Davis, *Awesome-single-cell*, Apr. 2022. [Online]. Available: https://github.com/seandavi/awesome-single-cell (visited on 04/21/2022).

[151] *Long-read-tools*, en. [Online]. Available: https://long-read-tools.org/ (visited on 04/21/2022).

[152] F. Marini, *Awesome-expression-browser*, Apr. 2022. [Online]. Available: https://github.com/federicomarini/awesome-expression-browser (visited on 04/21/2022).

[153] Dobin, Alexander and Davis, Carrie A and Schlesinger, Felix and Drenkow, Jorg and Zaleski, Chris and Jha, Sonali and Batut, Philippe and Chaisson, Mark and Gingeras, Thomas R, "STAR: Ultrafast universal RNA-seq aligner," *Bioinformatics*, vol. 29, no. 1, pp. 15–21, 2013.

[154] M. Mahmoud, N. Gobet, D. I. Cruz-Dávalos, N. Mounier, C. Dessimoz, and F. J. Sedlazeck, "Structural variant calling: The long and the short of it," *Genome Biology*, vol. 20, no. 1, p. 246, Nov. 2019, ISSN: 1474-760X. DOI: 10.1186/s13059-019-1828-7. [Online]. Available: https://doi.org/10.1186/s13059-019-1828-7 (visited on 04/21/2022).

[155] A. Nellore *et al.*, “Human splicing diversity and the extent of unannotated splice junctions across human RNA-seq samples on the Sequence Read Archive,” en, *Genome Biology*, vol. 17, no. 1, p. 266, Dec. 2016, ISSN: 1474-760X. DOI: 10.1186/s13059-016-1118-6. [Online]. Available: https://doi.org/10.1186/s13059-016-1118-6 (visited on 04/26/2022).

[156] A. Rhoads and K. F. Au, “PacBio Sequencing and Its Applications,” *Genomics, Proteomics & Bioinformatics*, vol. 13, no. 5, pp. 278–289, Oct. 2015, ISSN: 1672-0229. DOI: 10.1016/j.gpb.2015.08.002. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4678779/ (visited on 04/20/2022).

[157] R. I. Kuo *et al.*, “Illuminating the dark side of the human transcriptome with long read transcript sequencing,” *BMC Genomics*, vol. 21, p. 751, Oct. 2020, ISSN: 1471-2164. DOI: 10.1186/s12864-020-07123-7. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7596999/ (visited on 04/20/2022).

[158] Frankish, Adam and Diekhans, Mark and Ferreira, Anne-Maud and Johnson, Rory and Jungreis, Irwin and Loveland, Jane and Mudge, Jonathan M and Sisu, Cristina and Wright, James and Armstrong, Joel and others, “GENCODE reference annotation for the human and mouse genomes,” *Nucleic Acids Research*, vol. 47, no. D1, pp. D766–D773, 2019.

[159] *IlluminaHumanMethylationEPICmanifest*, en-US. [Online]. Available: http://bioconductor.org/packages/IlluminaHumanMethylationEPICmanifest/ (visited on 04/21/2022).

[160] *IlluminaHumanMethylation450kmanifest*, en-US. [Online]. Available: http://bioconductor.org/packages/IlluminaHumanMethylation450kmanifest/ (visited on 04/21/2022).

[161] *Infinium MethylationEPIC BeadChip Product Files*. [Online]. Available: https://support.illumina.com/array/array_kits/infinium-methylationepic-beadchip-kit/downloads.html (visited on 04/21/2022).

[162] J. Navarro Gonzalez *et al.*, “The UCSC Genome Browser database: 2021 update,” *Nucleic Acids Research*, vol. 49, no. D1, pp. D1046–D1057, Jan. 2021, ISSN: 0305-1048. DOI: 10.1093/nar/gkaa1070. [Online]. Available: https://doi.org/10.1093/nar/gkaa1070 (visited on 04/21/2022).

[163] A. S. Hinrichs *et al.*, “The UCSC Genome Browser Database: Update 2006,” *Nucleic Acids Research*, vol. 34, no. suppl_1, pp. D590–D598, Jan. 2006, ISSN: 0305-1048. DOI: 10.1093/nar/gkj144. [Online]. Available: https://doi.org/10.1093/nar/gkj144 (visited on 04/21/2022).

[164] M. E. Ritchie *et al.*, “A comparison of background correction methods for two-colour microarrays,” *Bioinformatics*, vol. 23, no. 20, pp. 2700–2707, Oct. 2007, ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btm412. [Online]. Available: https://doi.org/10.1093/bioinformatics/btm412 (visited on 04/25/2022).

[165] A. N. Gorban and I. Y. Tyukin, “Blessing of dimensionality: Mathematical foundations of the statistical physics of data,” *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, vol. 376, no. 2118, p. 20 170 237, Apr. 2018, ISSN: 1364-503X. DOI: 10.1098/rsta.2017.0237. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5869543/ (visited on 04/18/2022).

[166] B. Verhulst, J. N. Pritikin, J. Clifford, and E. Prom-Wormley, "Using Genetic Marginal Effects to Study Gene-Environment Interactions with GWAS Data," en, *Behavior Genetics*, vol. 51, no. 3, pp. 358–373, May 2021, ISSN: 1573-3297. DOI: 10.1007/s10519-021-10058-8. [Online]. Available: https://doi.org/10.1007/s10519-021-10058-8 (visited on 04/19/2022).

[167] Y. Benjamini and Y. Hochberg, "Controlling the false discovery rate: A practical and powerful approach to multiple testing," *JSRRB*, vol. 57, pp. 289–300, 1995.

[168] K. B. Michels *et al.*, "Recommendations for the design and analysis of epigenome-wide association studies," eng, *Nature Methods*, vol. 10, no. 10, pp. 949–955, Oct. 2013, ISSN: 1548-7105. DOI: 10.1038/nmeth.2632.

[169] A. N. Gorban, V. A. Makarov, and I. Y. Tyukin, "High-Dimensional Brain in a High-Dimensional World: Blessing of Dimensionality," *Entropy*, vol. 22, no. 1, p. 82, Jan. 2020, ISSN: 1099-4300. DOI: 10.3390/e22010082. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7516518/ (visited on 04/18/2022).

[170] V. Kreinovich and O. Kosheleva, "Limit Theorems as Blessing of Dimensionality: Neural-Oriented Overview," *Entropy*, vol. 23, no. 5, p. 501, Apr. 2021, ISSN: 1099-4300. DOI: 10.3390/e23050501. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8145334/ (visited on 04/18/2022).

[171] S. Holmes and W. Huber, *Modern Statistics for Modern Biology*. Cambridge, UK; New York, NY: Cambridge University Press, 2019.

[172] J. Shlens, "A Tutorial on Principal Component Analysis," *arXiv:1404.1100 [cs, stat]*, Apr. 2014. [Online]. Available: http://arxiv.org/abs/1404.1100 (visited on 04/20/2022).

[173] C. R. Shalizi, *Advanced data analysis from an elementary point of view*, Aug. 2014. [Online]. Available: https://www.stat.cmu.edu/~cshalizi/ADAfaEPoV/ADAfaEPoV.pdf.

[174] K. Weinberger, A. Dasgupta, J. Attenberg, J. Langford, and A. Smola, "Feature Hashing for Large Scale Multitask Learning," *arXiv:0902.2206 [cs]*, Feb. 2010. [Online]. Available: http://arxiv.org/abs/0902.2206 (visited on 10/29/2019).

[175] Q. Shi *et al.*, "Hash kernels," in *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, D. van Dyk and M. Welling, Eds., ser. Proceedings of Machine Learning Research, vol. 5, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA: PMLR, 2009, pp. 496–503.

[176] The HDF Group. "Hierarchical data format version 5." (2010), [Online]. Available: http://www.hdfgroup.org/HDF5.

[177] H. Pagès, *Hdf5array: Hdf5 backend for delayedarray objects*, 2021. [Online]. Available: https://bioconductor.org/packages/HDF5Array.

[178] B. Fischer, M. Smith, and G. Pau, *Rhdf5: R interface to hdf5*, 2020. [Online]. Available: https://github.com/grimbough/rhdf5.

[179] A. Saffari *et al.*, "Estimation of a significance threshold for epigenome-wide association studies," eng, *Genetic Epidemiology*, vol. 42, no. 1, pp. 20–33, Feb. 2018, ISSN: 1098-2272. DOI: 10.1002/gepi.22086.

[180] P.-C. Tsai and J. T. Bell, “Power and sample size estimation for epigenome-wide association scans to detect differential DNA methylation,” *International Journal of Epidemiology*, vol. 44, no. 4, pp. 1429–1441, Aug. 2015, ISSN: 0300-5771. DOI: 10.1093/ije/dyv041. [Online]. Available: https://doi.org/10.1093/ije/dyv041 (visited on 04/24/2022).

[181] S. Graw, R. Henn, J. A. Thompson, and D. C. Koestler, “pwrEWAS: A user-friendly tool for comprehensive power estimation for epigenome wide association studies (EWAS),” *BMC Bioinformatics*, vol. 20, no. 1, p. 218, Apr. 2019, ISSN: 1471-2105. DOI: 10.1186/s12859-019-2804-7. [Online]. Available: https://doi.org/10.1186/s12859-019-2804-7 (visited on 03/07/2021).

[182] S. Graw, *pwrEWAS: A user-friendly tool for comprehensive power estimation for epigenome wide association studies (EWAS)*, 2022. DOI: 10.18129/B9.bioc.pwrEWAS. [Online]. Available: https://bioconductor.org/packages/pwrEWAS/ (visited on 04/24/2022).

[183] *Anaconda software distribution*, version Vers. 2-2.4.0, 2020. [Online]. Available: https://docs.anaconda.com/.

[184] Lorenzi, Claudio and Barriere, Sylvain and Arnold, Katharina and Luco, Reini F and Oldfield, Andrew J and Ritchie, William, “IRFinder-S: A comprehensive suite to discover and explore intron retention,” *Genome Biology*, vol. 22, no. 1, pp. 1–13, 2021.

[185] S. Baichoo *et al.*, “Developing reproducible bioinformatics analysis workflows for heterogeneous computing environments to support African genomics,” *BMC Bioinformatics*, vol. 19, p. 457, Nov. 2018, ISSN: 1471-2105. DOI: 10.1186/s12859-018-2446-1. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6264621/ (visited on 04/24/2022).

[186] S. Orjuela, R. Huang, K. M. Hembach, M. D. Robinson, and C. Soneson, “ARMOR: An Automated Reproducible MOdular Workflow for Preprocessing and Differential Analysis of RNA-seq Data,” *G3: Genes|Genomes|Genetics*, vol. 9, no. 7, pp. 2089–2096, May 2019, ISSN: 2160-1836. DOI: 10.1534/g3.119.400185. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6643886/ (visited on 04/24/2022).

[187] X. Zhang and I. Jonassen, “RASflow: An RNA-Seq analysis workflow with Snakemake,” *BMC Bioinformatics*, vol. 21, p. 110, Mar. 2020, ISSN: 1471-2105. DOI: 10.1186/s12859-020-3433-x. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7079470/ (visited on 04/24/2022).

[188] A. E. Ahmed *et al.*, “Design considerations for workflow management systems use in production genomics research and the clinic,” *Scientific Reports*, vol. 11, p. 21 680, Nov. 2021, ISSN: 2045-2322. DOI: 10.1038/s41598-021-99288-8. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8569008/ (visited on 04/24/2022).

[189] P. Amstutz *et al.*, *Portable workflow and tool descriptions with the cwl (common workflow language)*, 2015. DOI: 10.7490/f1000research.1110021.1. [Online]. Available: http://dx.doi.org/10.7490/f1000research.1110021.1.

[190] E. Afgan *et al.*, "The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update," *Nucleic Acids Research*, vol. 46, no. W1, W537–W544, Jul. 2018, ISSN: 0305-1048. DOI: 10.1093/nar/gky379. [Online]. Available: https://doi.org/10.1093/nar/gky379 (visited on 04/24/2022).

[191] L. de la Garza *et al.*, "From the desktop to the grid: Scalable bioinformatics via workflow conversion," *BMC Bioinformatics*, vol. 17, p. 127, Mar. 2016, ISSN: 1471-2105. DOI: 10.1186/s12859-016-0978-9. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4788856/ (visited on 04/24/2022).

[192] F. Mölder *et al.*, "Sustainable data analysis with Snakemake," en, *F1000Research*, vol. 10, no. 10:33, 33 Jan. 2021. DOI: 10.12688/f1000research.29032.1. [Online]. Available: https://f1000research.com/articles/10-33 (visited on 11/24/2021).

[193] S. Maden, R. Thompson, K. Hansen, and A. Nellore, *Recountmethylation_server*, 2021. [Online]. Available: https://github.com/metamaden/recountmethylation%5C_server.

[194] ——, *Recountmethylation.pipeline*, 2021. [Online]. Available: https://github.com/metamaden/recountmethylation.pipeline.

[195] A. Lun and V. Carey, *Basilisk: Freezing python dependencies inside bioconductor packages*, 2022. DOI: 10.18129/B9.bioc.basilisk. [Online]. Available: https://bioconductor.org/packages/basilisk/ (visited on 04/22/2022).

[196] K. Ushey *et al.*, *Reticulate: Interface to 'python'*, 2022. [Online]. Available: https://CRAN.R-project.org/package=reticulate (visited on 04/22/2022).

[197] X. Du, J. J. Aristizabal-Henao, T. J. Garrett, M. Brochhausen, W. R. Hogan, and D. J. Lemas, "A Checklist for Reproducible Computational Analysis in Clinical Metabolomics Research," *Metabolites*, vol. 12, no. 1, p. 87, Jan. 2022, ISSN: 2218-1989. DOI: 10.3390/metabo12010087. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8779534/ (visited on 04/07/2022).

[198] A. P. Feinberg and B. Tycko, "The history of cancer epigenetics," eng, *Nature Reviews. Cancer*, vol. 4, no. 2, pp. 143–153, 2004, ISSN: 1474-175X. DOI: 10.1038/nrc1279.

[199] M. J. Ziller *et al.*, "Charting a dynamic DNA methylation landscape of the human genome," en, *Nature*, vol. 500, no. 7463, pp. 477–481, Aug. 2013, ISSN: 1476-4687. DOI: 10.1038/nature12433. [Online]. Available: https://www.nature.com/articles/nature12433 (visited on 11/30/2018).

[200] K. Lokk *et al.*, "DNA methylome profiling of human tissues identifies global and tissue-specific methylation patterns," *Genome Biology*, vol. 15, no. 4, r54, 2014, ISSN: 1465-6906. DOI: 10.1186/gb-2014-15-4-r54. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4053947/ (visited on 06/10/2019).

[201] H.-M. Byun *et al.*, "Epigenetic profiling of somatic tissues from human autopsy specimens identifies tissue- and individual-specific DNA methylation patterns," eng, *Human Molecular Genetics*, vol. 18, no. 24, pp. 4808–4817, Dec. 2009, ISSN: 1460-2083. DOI: 10.1093/hmg/ddp445.

[202] H. Heyn *et al.*, "Epigenomic analysis detects aberrant super-enhancer DNA methylation in human cancer," *Genome Biology*, vol. 17, 2016, ISSN: 1474-7596. DOI: 10.1186/s13059-016-0879-2. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4728783/ (visited on 03/09/2018).

[203] J.-P. Issa, "CpG island methylator phenotype in cancer," eng, *Nature Reviews. Cancer*, vol. 4, no. 12, pp. 988–993, Dec. 2004, ISSN: 1474-175X. DOI: 10.1038/nrc1507.

[204] T. J. Peters *et al.*, "De novo identification of differentially methylated regions in the human genome," *Epigenetics & Chromatin*, vol. 8, Jan. 2015, ISSN: 1756-8935. DOI: 10.1186/1756-8935-8-6. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4429355/ (visited on 04/25/2018).

[205] R. Edgar, M. Domrachev, and A. E. Lash, "Gene expression omnibus: Ncbi gene expression and hybridization array data repository," *Nucleic acids research*, vol. 30, no. 1, pp. 207–210, 2002.

[206] J. A. Heiss and A. C. Just, "Identifying mislabeled and contaminated DNA methylation microarray data: An extended quality control toolset with examples from GEO," *Clinical Epigenetics*, vol. 10, Jun. 2018, ISSN: 1868-7075. DOI: 10.1186/s13148-018-0504-1. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5984806/ (visited on 09/12/2019).

[207] W. Huber *et al.*, "Orchestrating high-throughput genomic analysis with Bioconductor," *Nature Methods*, vol. 12, no. 2, pp. 115–121, 2015. [Online]. Available: http://www.nature.com/nmeth/journal/v12/n2/full/nmeth.3252.html.

[208] Illumina, *BeadArray Controls Reporter Software Guide*, Oct. 2015. [Online]. Available: https://support.illumina.com/content/dam/illumina-support/documents/documentation/chemistry_documentation/infinium_assays/infinium_hd_methylation/beadarray-controls-reporter-user-guide-1000000004009-00.pdf (visited on 08/20/2019).

[209] D. M. Kane and J. Nelson, "Sparser johnson-lindenstrauss transforms," *Journal of the ACM (JACM)*, vol. 61, no. 1, pp. 1–23, 2014.

[210] Z. Gu, R. Eils, and M. Schlesner, "Complex heatmaps reveal patterns and correlations in multidimensional genomic data," en, *Bioinformatics*, vol. 32, no. 18, pp. 2847–2849, Sep. 2016, ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btw313. [Online]. Available: https://academic.oup.com/bioinformatics/article/32/18/2847/1743594 (visited on 11/01/2020).

[211] H. Wickham, *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016, ISBN: 978-3-319-24277-4. [Online]. Available: https://ggplot2.tidyverse.org.

[212] M. J. Bonder *et al.*, "Genetic and epigenetic regulation of gene expression in fetal and adult human livers," *BMC Genomics*, vol. 15, no. 1, p. 860, Oct. 2014, ISSN: 1471-2164. DOI: 10.1186/1471-2164-15-860. [Online]. Available: https://doi.org/10.1186/1471-2164-15-860 (visited on 09/07/2020).

[213] S. Horvath and B. R. Ritz, "Increased epigenetic age and granulocyte counts in the blood of Parkinson's disease patients," eng, *Aging*, vol. 7, no. 12, pp. 1130–1142, Dec. 2015, ISSN: 1945-4589. DOI: 10.18632/aging.100859.

[214] D. T. Butcher *et al.*, "Charge and kabuki syndromes: Gene-specific dna methylation signatures identify epigenetic mechanisms linking these clinically overlapping conditions," English, *The American Journal of Human Genetics*, vol. 100, no. 5, pp. 773–788, May 2017, ISSN: 0002-9297, 1537-6605. DOI: 10.1016/j.ajhg.2017.04.004. (visited on 09/07/2020).

[215] A. C. Espinal *et al.*, "A methodological study of genome-wide DNA methylation analyses using matched archival formalin-fixed paraffin embedded and fresh frozen breast tumors," *Oncotarget*, vol. 8, no. 9, pp. 14 821–14 829, Jan. 2017, ISSN: 1949-2553. DOI: 10.18632/oncotarget.14739. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5362446/ (visited on 01/07/2020).

[216] E. M. Siegel *et al.*, "Expanding Epigenomics to Archived FFPE Tissues: An Evaluation of DNA Repair Methodologies," en, *Cancer Epidemiology and Prevention Biomarkers*, vol. 23, no. 12, pp. 2622–2631, Dec. 2014, ISSN: 1055-9965, 1538-7755. DOI: 10.1158/1055-9965.EPI-14-0464. [Online]. Available: https://cebp.aacrjournals.org/content/23/12/2622 (visited on 01/07/2020).

[217] T. Kling, A. Wenger, S. Beck, and H. Carén, "Validation of the MethylationEPIC BeadChip for fresh-frozen and formalin-fixed paraffin-embedded tumours," *Clinical Epigenetics*, vol. 9, no. 1, p. 33, Apr. 2017, ISSN: 1868-7083. DOI: 10.1186/s13148-017-0333-7. [Online]. Available: https://doi.org/10.1186/s13148-017-0333-7 (visited on 01/07/2020).

[218] N. Acevedo *et al.*, "Age-associated DNA methylation changes in immune genes, histone modifiers and chromatin remodeling factors within 5 years after birth in human blood leukocytes," eng, *Clinical Epigenetics*, vol. 7, p. 34, 2015, ISSN: 1868-7075. DOI: 10.1186/s13148-015-0064-6.

[219] G. C. Sharp *et al.*, "Distinct DNA methylation profiles in subtypes of orofacial cleft," *Clinical Epigenetics*, vol. 9, no. 1, p. 63, Jun. 2017, ISSN: 1868-7083. DOI: 10.1186/s13148-017-0362-2. [Online]. Available: https://doi.org/10.1186/s13148-017-0362-2 (visited on 10/31/2019).

[220] J. I. Orozco, A. O. Manughian-Peter, M. P. Salomon, and D. M. Marzese, "Epigenetic classifiers for precision diagnosis of brain tumors," *Epigenetics Insights*, vol. 12, p. 2 516 865 719 840 284, 2019.

[221] M. P. Salomon *et al.*, "Brain metastasis dna methylomes, a novel resource for the identification of biological and clinical features," *Scientific data*, vol. 5, no. 1, pp. 1–8, 2018.

[222] X. Wang *et al.*, "Medulloblastoma subgroups remain stable across primary and metastatic compartments," eng, *Acta Neuropathologica*, vol. 129, no. 3, pp. 449–457, Mar. 2015, ISSN: 1432-0533. DOI: 10.1007/s00401-015-1389-0.

[223] K. D. Hansen *et al.*, "Increased methylation variation in epigenetic domains across cancer types," *Nature genetics*, vol. 43, no. 8, p. 768, 2011.

[224] A. E. Teschendorff *et al.*, "Epigenetic variability in cells of normal cytology is associated with the risk of future morphological transformation," eng, *Genome Medicine*, vol. 4, no. 3, p. 24, 2012, ISSN: 1756-994X. DOI: 10.1186/gm323.

[225] J. Hyun and Y. Jung, “DNA Methylation in Nonalcoholic Fatty Liver Disease,” *International Journal of Molecular Sciences*, vol. 21, no. 21, p. 8138, Oct. 2020, ISSN: 1422-0067. DOI: 10.3390/ijms21218138. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7662478/ (visited on 01/12/2022).

[226] A.-A. Hudon Thibeault and C. Laprise, “Cell-Specific DNA Methylation Signatures in Asthma,” *Genes*, vol. 10, no. 11, p. 932, Nov. 2019, ISSN: 2073-4425. DOI: 10.3390/genes10110932. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6896152/ (visited on 01/12/2022).

[227] P. D. Fransquet *et al.*, “Blood DNA methylation signatures to detect dementia prior to overt clinical symptoms,” en, *Alzheimer’s & Dementia: Diagnosis, Assessment & Disease Monitoring*, vol. 12, no. 1, e12056, 2020, ISSN: 2352-8729. DOI: 10.1002/dad2.12056. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/dad2.12056 (visited on 01/12/2022).

[228] Y.-T. Huang *et al.*, “Epigenome-wide profiling of DNA methylation in paired samples of adipose tissue and blood,” *Epigenetics*, vol. 11, no. 3, pp. 227–236, Mar. 2016, ISSN: 1559-2294. DOI: 10.1080/15592294.2016.1146853. [Online]. Available: https://doi.org/10.1080/15592294.2016.1146853 (visited on 08/25/2020).

[229] N. Parveen and S. Dhawan, “DNA Methylation Patterning and the Regulation of Beta Cell Homeostasis,” *Frontiers in Endocrinology*, vol. 12, p. 651 258, May 2021, ISSN: 1664-2392. DOI: 10.3389/fendo.2021.651258. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8137853/ (visited on 01/12/2022).

[230] B. T. Mayne, S. Y. Leemaqz, A. K. Smith, J. Breen, C. T. Roberts, and T. Bianco-Miotto, “Accelerated placental aging in early onset preeclampsia pregnancies identified by DNA methylation,” *Epigenomics*, vol. 9, no. 3, pp. 279–289, Mar. 2017, ISSN: 1750-1911. DOI: 10.2217/epi-2016-0103. [Online]. Available: https://www.futuremedicine.com/doi/full/10.2217/epi-2016-0103 (visited on 03/01/2022).

[231] D. C. Koestler *et al.*, “Improving cell mixture deconvolution by identifying optimal DNA methylation libraries (IDOL),” *BMC Bioinformatics*, vol. 17, Mar. 2016, ISSN: 1471-2105. DOI: 10.1186/s12859-016-0943-7. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4782368/ (visited on 10/26/2018).

[232] D. R. Masser *et al.*, “Sexually divergent DNA methylation patterns with hippocampal aging,” *Aging Cell*, vol. 16, no. 6, pp. 1342–1352, Dec. 2017, ISSN: 1474-9718. DOI: 10.1111/acel.12681. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5676057/ (visited on 01/14/2022).

[233] E. Hall *et al.*, “Sex differences in the genome-wide DNA methylation pattern and impact on gene expression, microRNA levels and insulin secretion in human pancreatic islets,” *Genome Biology*, vol. 15, no. 12, p. 522, Dec. 2014, ISSN: 1474-760X. DOI: 10.1186/s13059-014-0522-z. [Online]. Available: https://doi.org/10.1186/s13059-014-0522-z (visited on 01/14/2022).

[234] C. L. Nino, G. F. Perez, N. Isaza, M. J. Gutierrez, J. L. Gomez, and G. Nino, "Characterization of Sex-Based Dna Methylation Signatures in the Airways During Early Life," *Scientific Reports*, vol. 8, p. 5526, Apr. 2018, ISSN: 2045-2322. DOI: 10.1038/s41598-018-23063-5. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5882800/ (visited on 01/14/2022).

[235] M. Maschietto *et al.*, "Sex differences in DNA methylation of the cord blood are related to sex-bias psychiatric diseases," en, *Scientific Reports*, vol. 7, no. 1, p. 44 547, Mar. 2017, ISSN: 2045-2322. DOI: 10.1038/srep44547. [Online]. Available: https://www.nature.com/articles/srep44547 (visited on 01/14/2022).

[236] O. A. Grant, Y. Wang, M. Kumari, N. R. Zabet, and L. Schalkwyk, "Characterising sex differences of autosomal DNA methylation in whole blood using the Illumina EPIC array," en, *bioRxiv*, p. 2021.09.02.458717, Sep. 2021. DOI: 10.1101/2021.09.02.458717. (visited on 01/14/2022).

[237] H. Pagès, P. Hickey, and A. Lun, *Delayedarray: A unified framework for working transparently with on-disk and in-memory array-like datasets*, 2021. [Online]. Available: https://bioconductor.org/packages/DelayedArray.

[238] J.-P. Fortin, T. J. Triche, and K. D. Hansen, "Preprocessing, normalization and integration of the illumina humanmethylationepic array with minfi," *Bioinformatics*, vol. 33, no. 4, 2017. DOI: 10.1093/bioinformatics/btw691.

[239] M. E. Ritchie *et al.*, "limma powers differential expression analyses for RNA-sequencing and microarray studies," *Nucleic Acids Research*, vol. 43, no. 7, e47, 2015. DOI: 10.1093/nar/gkv007.

[240] R. Williams, "A new algorithm for optimal constraint satisfaction and its implications," en, p. 13,

[241] Y. A. Malkov and D. A. Yashunin, "Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs," *arXiv:1603.09320 [cs]*, Aug. 2018. [Online]. Available: http://arxiv.org/abs/1603.09320 (visited on 03/23/2021).

[242] M. Aumüller, E. Bernhardsson, and A. Faithfull, "ANN-Benchmarks: A Benchmarking Tool for Approximate Nearest Neighbor Algorithms," *arXiv:1807.05614 [cs]*, Jul. 2018. [Online]. Available: http://arxiv.org/abs/1807.05614 (visited on 03/28/2021).

[243] C. R. Harris *et al.*, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. DOI: 10.1038/s41586-020-2649-2. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2.

[244] J. T. Leek *et al.*, *Sva: Surrogate variable analysis*, 2021.

[245] R Core Team, *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, 2021. [Online]. Available: https://www.R-project.org/.

[246] Python Core Team, *Python: A dynamic, open source programming language*, Python version 3.7, Python Software Foundation, 2019. [Online]. Available: https://www.python.org/.

[247] N. Gehlenborg, *Upsetr: A more scalable alternative to venn and euler diagrams for visualizing intersecting sets*, 2019. [Online]. Available: https://CRAN.R-project.org/package=UpSetR.

[248] J. R. Murray and M. S. Rajeevan, "Evaluation of DNA extraction from granulocytes discarded in the separation medium after isolation of peripheral blood mononuclear cells and plasma from whole blood," *BMC Research Notes*, vol. 6, p. 440, Nov. 2013, ISSN: 1756-0500. DOI: `10.1186/1756-0500-6-440`. [Online]. Available: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3818442/` (visited on 02/07/2022).

[249] G. Mansell *et al.*, "Guidance for DNA methylation studies: Statistical insights from the Illumina EPIC array," *BMC Genomics*, vol. 20, May 2019, ISSN: 1471-2164. DOI: `10.1186/s12864-019-5761-7`. [Online]. Available: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6518823/` (visited on 04/04/2021).

[250] *Field Guide to Methylation Methods*, Jun. 2016. [Online]. Available: `https://www.illumina.com/content/dam/illumina-marketing/documents/products/other/field_guide_methylation.pdf` (visited on 11/23/2021).

[251] A. Bird, "DNA methylation patterns and epigenetic memory," en, *Genes & Development*, vol. 16, no. 1, pp. 6–21, Jan. 2002, ISSN: 0890-9369, 1549-5477. DOI: `10.1101/gad.947102`. (visited on 11/23/2021).

[252] B. J. Heil, M. M. Hoffman, F. Markowetz, S.-I. Lee, C. S. Greene, and S. C. Hicks, "Reproducibility standards for machine learning in the life sciences," en, *Nature Methods*, vol. 18, no. 10, pp. 1132–1135, Oct. 2021, ISSN: 1548-7105. DOI: `10.1038/s41592-021-01256-7`. [Online]. Available: `https://www.nature.com/articles/s41592-021-01256-7` (visited on 11/24/2021).

[253] B. K. Beaulieu-Jones and C. S. Greene, "Reproducibility of computational workflows is automated using continuous analysis," *Nature biotechnology*, vol. 35, no. 4, pp. 342–346, Apr. 2017, ISSN: 1087-0156. DOI: `10.1038/nbt.3780`. [Online]. Available: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6103790/` (visited on 03/12/2021).

[254] A. J. Noble *et al.*, "A validation of Illumina EPIC array system with bisulfite-based amplicon sequencing," eng, *PeerJ*, vol. 9, e10762, 2021, ISSN: 2167-8359. DOI: `10.7717/peerj.10762`.

[255] T. Wang *et al.*, "A systematic study of normalization methods for Infinium 450K methylation data using whole-genome bisulfite sequencing data," eng, *Epigenetics: official journal of the DNA Methylation Society*, vol. 10, no. 7, pp. 662–669, Jul. 2015, ISSN: 1559-2308. DOI: `10.1080/15592294.2015.1057384`.

[256] Herzel, Lydia and Ottoz, Diana SM and Alpert, Tara and Neugebauer, Karla M, "Splicing and transcription touch base: co-transcriptional spliceosome assembly and function," *Nature Reviews Molecular Cell Biology*, vol. 18, no. 10, pp. 637–650, 2017.

[257] Wan, Yihan and Anastasakis, Dimitrios G and Rodriguez, Joseph and Palangat, Murali and Gudla, Prabhakar and Zaki, George and Tandon, Mayank and Pegoraro, Gianluca and Chow, Carson C and Hafner, Markus and others, "Dynamic imaging of nascent RNA reveals general principles of transcription dynamics and stochastic splice site selection," *Cell*, vol. 184, no. 11, pp. 2878–2895, 2021.

[258] Ameur, Adam and Zaghlool, Ammar and Halvardson, Jonatan and Wetterbom, Anna and Gyllensten, Ulf and Cavelier, Lucia and Feuk, Lars, "Total RNA sequencing reveals nascent transcription and widespread

co-transcriptional splicing in the human brain," *Nature Structural & Molecular Biology*, vol. 18, no. 12, pp. 1435–1440, 2011.

[259] Alpert, Tara and Herzel, Lydia and Neugebauer, Karla M, "Perfect timing: Splicing and transcription rates in living cells," *Wiley Interdisciplinary Reviews: RNA*, vol. 8, no. 2, e1401, 2017.

[260] Reimer, Kirsten A and Mimoso, Claudia and Adelman, Karen and Neugebauer, Karla M, "Rapid and efficient co-transcriptional splicing enhances mammalian gene expression," *bioRxiv*, pp. 2020–02, 2020.

[261] Girard, Cyrille and Will, Cindy L and Peng, Jianhe and Makarov, Evgeny M and Kastner, Berthold and Lemm, Ira and Urlaub, Henning and Hartmuth, Klaus and Lührmann, Reinhard, "Post-transcriptional spliceosomes are retained in nuclear speckles until splicing completion," *Nature Communications*, vol. 3, no. 1, pp. 1–12, 2012.

[262] Moyer, Devlin C and Larue, Graham E and Hershberger, Courtney E and Roy, Scott W and Padgett, Richard A, "Comprehensive database and evolutionary dynamics of U12-type introns," *Nucleic Acids Research*, vol. 48, no. 13, pp. 7066–7078, 2020.

[263] Zhang, Guohong and Taneja, Krishan L and Singer, Robert H and Green, Michael R, "Localization of pre-mRNA splicing in mammalian nuclei," *Nature*, vol. 372, no. 6508, pp. 809–812, 1994.

[264] Denis, Melvin M and Tolley, Neal D and Bunting, Michaeline and Schwertz, Hansjörg and Jiang, Huimiao and Lindemann, Stephan and Yost, Christian C and Rubner, Frederick J and Albertine, Kurt H and Swoboda, Kathryn J and others, "Escaping the nuclear confines: signal-dependent pre-mRNA splicing in anucleate platelets," *Cell*, vol. 122, no. 3, pp. 379–391, 2005.

[265] König, Harald and Matter, Nathalie and Bader, Rüdiger and Thiele, Wilko and Müller, Ferenc, "Splicing segregation: The minor spliceosome acts outside the nucleus and controls cell proliferation," *Cell*, vol. 131, no. 4, pp. 718–729, 2007.

[266] Buckley, Peter T and Khaladkar, Mugdha and Kim, Junhyong and Eberwine, James, "Cytoplasmic intron retention, function, splicing, and the sentinel RNA hypothesis," *Wiley Interdisciplinary Reviews: RNA*, vol. 5, no. 2, pp. 223–230, 2014.

[267] Uemura, Aya and Oku, Masaya and Mori, Kazutoshi and Yoshida, Hiderou, "Unconventional splicing of XBP1 mRNA occurs in the cytoplasm during the mammalian unfolded protein response," *Journal of Cell Science*, vol. 122, no. 16, pp. 2877–2886, 2009.

[268] Middleton, Robert and Gao, Dadi and Thomas, Aubin and Singh, Babita and Au, Amy and Wong, Justin JL and Bomane, Alexandra and Cosson, Bertrand and Eyras, Eduardo and Rasko, John EJ and others, "IRFinder: assessing the impact of intron retention on mammalian gene expression," *Genome Biology*, vol. 18, no. 1, pp. 1–11, 2017.

[269] Yap, Karen and Lim, Zhao Qin and Khandelia, Piyush and Friedman, Brad and Makeyev, Eugene V, "Coordinated regulation of neuronal mRNA steady-state levels through developmentally controlled intron retention," *Genes & Development*, vol. 26, no. 11, pp. 1209–1223, 2012.

[270] Edwards, Christopher R and Ritchie, William and Wong, Justin J-L and Schmitz, Ulf and Middleton, Robert and An, Xiuli and Mohandas, Narla and Rasko, John EJ and Blobel, Gerd A, “A dynamic intron retention program in the mammalian megakaryocyte and erythrocyte lineages,” *Blood, The Journal of the American Society of Hematology*, vol. 127, no. 17, e24–e34, 2016.

[271] Pimentel, Harold and Parra, Marilyn and Gee, Sherry L and Mohandas, Narla and Pachter, Lior and Conboy, John G, “A dynamic intron retention program enriched in RNA processing genes regulates gene expression during terminal erythropoiesis,” *Nucleic Acids Research*, vol. 44, no. 2, pp. 838–851, 2016.

[272] Wong, Justin J-L and Ritchie, William and Ebner, Olivia A and Selbach, Matthias and Wong, Jason WH and Huang, Yizhou and Gao, Dadi and Pinello, Natalia and Gonzalez, Maria and Baidya, Kinsha and others, “Orchestrated intron retention regulates normal granulocyte differentiation,” *Cell*, vol. 154, no. 3, pp. 583–595, 2013.

[273] Lareau, Liana F and Inada, Maki and Green, Richard E and Wengrod, Jordan C and Brenner, Steven E, “Unproductive splicing of SR genes associated with highly conserved and ultraconserved DNA elements,” *Nature*, vol. 446, no. 7138, pp. 926–929, 2007.

[274] Ge, Ying and Porse, Bo T, “The functional consequences of intron retention: alternative splicing coupled to NMD as a regulator of gene expression,” *Bioessays*, vol. 36, no. 3, pp. 236–243, 2014.

[275] Naro, Chiara and Jolly, Ariane and Di Persio, Sara and Bielli, Pamela and Setterblad, Niclas and Alberdi, Antonio J and Vicini, Elena and Geremia, Raffaele and De la Grange, Pierre and Sette, Claudio, “An orchestrated intron retention program in meiosis controls timely usage of transcripts during germ cell differentiation,” *Developmental Cell*, vol. 41, no. 1, pp. 82–93, 2017.

[276] Memon, Danish and Dawson, Keren and Smowton, Christopher SF and Xing, Wei and Dive, Caroline and Miller, Crispin J, “Hypoxia-driven splicing into noncoding isoforms regulates the DNA damage response,” *npj Genomic Medicine*, vol. 1, no. 1, pp. 1–7, 2016.

[277] Mauger, Oriane and Lemoine, Frédéric and Scheiffele, Peter, “Targeted intron retention and excision for rapid gene regulation in response to neuronal activity,” *Neuron*, vol. 92, no. 6, pp. 1266–1278, 2016.

[278] Ni, Ting and Yang, Wenjing and Han, Miao and Zhang, Yubo and Shen, Ting and Nie, Hongbo and Zhou, Zhihui and Dai, Yalei and Yang, Yanqin and Liu, Poching and others, “Global intron retention mediated gene regulation during CD4+ T cell activation,” *Nucleic Acids Research*, vol. 44, no. 14, pp. 6817–6829, 2016.

[279] Boutz, Paul L and Bhutkar, Arjun and Sharp, Phillip A, “Detained introns are a novel, widespread class of post-transcriptionally spliced introns,” *Genes & Development*, vol. 29, no. 1, pp. 63–80, 2015.

[280] Dvinge, Heidi and Bradley, Robert K, “Widespread intron retention diversifies most cancer transcriptomes,” *Genome Medicine*, vol. 7, no. 1, pp. 1–13, 2015.

[281] Zhang, Qu and Li, Hua and Jin, Hong and Tan, Huibiao and Zhang, Jun and Sheng, Sitong, “The global landscape of intron retentions in lung adenocarcinoma,” *BMC Medical Genomics*, vol. 7, no. 1, pp. 1–9, 2014.

[282] Eswaran, Jeyanthy and Horvath, Anelia and Godbole, Sucheta and Reddy, Sirigiri Divijendra and Mudvari, Prakriti and Ohshiro, Kazufumi and Cyanam, Dinesh and Nair, Sujit and Fuqua, Suzanne AW and Polyak, Kornelia and others, “RNA sequencing of cancer reveals novel splicing alterations,” *Scientific Reports*, vol. 3, no. 1, pp. 1–12, 2013.

[283] Monteuuis, Geoffray and Wong, Justin JL and Bailey, Charles G and Schmitz, Ulf and Rasko, John EJ, “The changing paradigm of intron retention: Regulation, ramifications and recipes,” *Nucleic Acids Research*, vol. 47, no. 22, pp. 11 497–11 513, 2019.

[284] de Lima Morais, David A and Harrison, Paul M, “Large-scale evidence for conservation of NMD candidature across mammals,” *PLoS One*, vol. 5, no. 7, e11695, 2010.

[285] Buckley, Peter T and Lee, Miler T and Sul, Jai-Yoon and Miyashiro, Kevin Y and Bell, Thomas J and Fisher, Stephen A and Kim, Junhyong and Eberwine, James, “Cytoplasmic intron sequence-retaining transcripts can be dendritically targeted via ID element retrotransposons,” *Neuron*, vol. 69, no. 5, pp. 877–884, 2011.

[286] Smart, Alicia C and Margolis, Claire A and Pimentel, Harold and He, Meng Xiao and Miao, Diana and Adeegbe, Dennis and Fugmann, Tim and Wong, Kwok-Kin and Van Allen, Eliezer M, “Intron retention is a source of neoepitopes in cancer,” *Nature Biotechnology*, vol. 36, no. 11, pp. 1056–1058, 2018.

[287] Kahles, André and Lehmann, Kjong-Van and Toussaint, Nora C and Hüser, Matthias and Stark, Stefan G and Sachsenberg, Timo and Stegle, Oliver and Kohlbacher, Oliver and Sander, Chris and Caesar-Johnson, Samantha J and others, “Comprehensive analysis of alternative splicing across tumors from 8,705 patients,” *Cancer Cell*, vol. 34, no. 2, pp. 211–224, 2018.

[288] Trincado, Juan L and Reixachs-Sole, Marina and Pérez-Granado, Judith and Fugmann, Tim and Sanz, Ferran and Yokota, Jun and Eyras, Eduardo, “ISOTOPE: ISOform-guided prediction of epiTOPEs in cancer,” *PLoS Computational Biology*, vol. 17, no. 9, e1009411, 2021.

[289] Dong, Chuanpeng and Cesarano, Annamaria and Bombaci, Giuseppe and Reiter, Jill L and Yu, Christina Y and Wang, Yue and Jiang, Zhaoyang and Zaid, Mohammad Abu and Huang, Kun and Lu, Xiongbin and others, “Intron retention-induced neoantigen load correlates with unfavorable prognosis in multiple myeloma,” *Oncogene*, vol. 40, no. 42, pp. 6130–6138, 2021.

[290] Dong, Chuanpeng and Reiter, Jill L and Dong, Edward and Wang, Yue and Lee, Kelvin P and Lu, Xiongbin and Liu, Yunlong, “Intron-Retention neoantigen load predicts favorable prognosis in pancreatic cancer,” *JCO Clinical Cancer Informatics*, vol. 6, e2100124, 2022.

[291] Szabo, Linda and Salzman, Julia, “Detecting circular RNAs: Bioinformatic and experimental challenges,” *Nature Reviews Genetics*, vol. 17, no. 11, pp. 679–692, 2016.

[292] Pimentel, Harold and Conboy, John G and Pachter, Lior, “Keep me around: Intron retention detection and analysis,” *arXiv preprint arXiv:1510.00696*, 2015.

[293] Oghabian, Ali and Greco, Dario and Frilander, Mikko J, “IntEREst: Intron-exon retention estimator,” *BMC Bioinformatics*, vol. 19, no. 1, pp. 1–10, 2018.

[294] Li, Hong-Dong and Funk, Cory C and Price, Nathan D, “iREAD: a tool for intron retention detection from RNA-seq data,” *BMC Genomics*, vol. 21, no. 1, pp. 1–11, 2020.

[295] Lee, Stuart and Zhang, Albert Y and Su, Shian and Ng, Ashley P and Holik, Aliaksei Z and Asselin-Labat, Marie-Liesse and Ritchie, Matthew E and Law, Charity W, “Covering all your bases: Incorporating intron signal from RNA-seq data,” *NAR Genomics and Bioinformatics*, vol. 2, no. 3, lqaa073, 2020.

[296] Tilgner, Hagen and Knowles, David G and Johnson, Rory and Davis, Carrie A and Chakrabortty, Sudipto and Djebali, Sarah and Curado, João and Snyder, Michael and Gingeras, Thomas R and Guigó, Roderic, “Deep sequencing of subcellular RNA fractions shows splicing to be predominantly co-transcriptional in the human genome but inefficient for lncRNAs,” *Genome Research*, vol. 22, no. 9, pp. 1616–1625, 2012.

[297] Braunschweig, Ulrich and Barbosa-Morais, Nuno L and Pan, Qun and Nachman, Emil N and Alipanahi, Babak and Gonatopoulos-Pournatzis, Thomas and Frey, Brendan and Irimia, Manuel and Blencowe, Benjamin J, “Widespread intron retention in mammals functionally tunes transcriptomes,” *Genome Research*, vol. 24, no. 11, pp. 1774–1786, 2014.

[298] Kronenberg, Zev N and Fiddes, Ian T and Gordon, David and Murali, Shwetha and Cantsilieris, Stuart and Meyerson, Olivia S and Underwood, Jason G and Nelson, Bradley J and Chaisson, Mark JP and Dougherty, Max L and others, “High-resolution comparative analysis of great ape genomes,” *Science*, vol. 360, no. 6393, eaar6343, 2018.

[299] Shi, Lingling and Guo, Yunfei and Dong, Chengliang and Huddleston, John and Yang, Hui and Han, Xiaolu and Fu, Aisi and Li, Quan and Li, Na and Gong, Siyi and others, “Long-read sequencing and de novo assembly of a Chinese genome,” *Nature Communications*, vol. 7, no. 1, pp. 1–10, 2016.

[300] Li, Heng and Handsaker, Bob and Wysoker, Alec and Fennell, Tim and Ruan, Jue and Homer, Nils and Marth, Gabor and Abecasis, Goncalo and Durbin, Richard, “The sequence alignment/map format and SAMtools,” *Bioinformatics*, vol. 25, no. 16, pp. 2078–2079, 2009.

[301] Bonfield, James K and Marshall, John and Danecek, Petr and Li, Heng and Ohan, Valeriu and Whitwham, Andrew and Keane, Thomas and Davies, Robert M, “HTSlib: C library for reading/writing high-throughput sequencing data,” *Gigascience*, vol. 10, no. 2, giab007, 2021.

[302] Langmead, Ben and Salzberg, Steven L, “Fast gapped-read alignment with Bowtie 2,” *Nature Methods*, vol. 9, no. 4, pp. 357–359, 2012.

[303] *pachterlab/kma: Keep Me Around: Intron retention detection*, `https://github.com/pachterlab/kma`.

[304] Roberts, Adam and Pachter, Lior, “Streaming fragment assignment for real-time analysis of sequencing experiments,” *Nature Methods*, vol. 10, no. 1, pp. 71–73, 2013.

[305] Roberts, Adam and Trapnell, Cole and Donaghey, Julie and Rinn, John L and Pachter, Lior, “Improving RNA-Seq expression estimates by correcting for fragment bias,” *Genome Biology*, vol. 12, no. 3, pp. 1–14, 2011.

[306] Li, Hong-Dong, "GTFtools: A Python package for analyzing various modes of gene models," *bioRxiv*, p. 263 517, 2018.

[307] Fleiss, Joseph L, "Measuring nominal scale agreement among many raters," *Psychological Bulletin*, vol. 76, no. 5, p. 378, 1971.

[308] *irr: Various Coefficients of Interrater Reliability and Agreement*, `https://CRAN.R-project.org/package=irr`.

[309] Glažar, Petar and Papavasileiou, Panagiotis and Rajewsky, Nikolaus, "circBase: A database for circular RNAs," *RNA*, vol. 20, no. 11, pp. 1666–1670, 2014.

[310] Jeong, Ji-Eun and Seol, Binna and Kim, Han-Seop and Kim, Jae-Yun and Cho, Yee-Sook, "Exploration of alternative splicing events in mesenchymal stem cells from human induced pluripotent stem cells," *Genes*, vol. 12, no. 5, p. 737, 2021.

[311] Lejeune, Fabrice and Cavaloc, Yvon and Stevenin, James, "Alternative splicing of intron 3 of the serine/arginine-rich protein 9G8 gene: Identification of flanking exonic splicing enhancers and involvement of 9G8 as a trans-acting factor," *Journal of Biological Chemistry*, vol. 276, no. 11, pp. 7850–7858, 2001.

[312] Li, Hong-Dong and Funk, Cory C and McFarland, Karen and Dammer, Eric B and Allen, Mariet and Carrasquillo, Minerva M and Levites, Yona and Chakrabarty, Paramita and Burgess, Jeremy D and Wang, Xue and others, "Integrative functional genomic analysis of intron retention in human and mouse brain with Alzheimer's disease," *Alzheimer's & Dementia*, vol. 17, no. 6, pp. 984–1004, 2021.

[313] Dumbović, Gabrijela and Braunschweig, Ulrich and Langner, Heera K and Smallegan, Michael and Biayna, Josep and Hass, Evan P and Jastrzebska, Katarzyna and Blencowe, Benjamin and Cech, Thomas R and Caruthers, Marvin H and others, "Nuclear compartmentalization of TERT mRNA and TUG1 lncRNA is driven by intron retention," *Nature Communications*, vol. 12, no. 1, pp. 1–19, 2021.

[314] Inoue, Daichi and Polaski, Jacob T and Taylor, Justin and Castel, Pau and Chen, Sisi and Kobayashi, Susumu and Hogg, Simon J and Hayashi, Yasutaka and Pineda, Jose Mario Bello and El Marabti, Ettaib and others, "Minor intron retention drives clonal hematopoietic disorders and diverse cancer predisposition," *Nature Genetics*, vol. 53, no. 5, pp. 707–718, 2021.

[315] Bell, Thomas J and Miyashiro, Kevin Y and Sul, Jai-Yoon and McCullough, Ronald and Buckley, Peter T and Jochems, Jeanine and Meaney, David F and Haydon, Phil and Cantor, Charles and Parsons, Thomas D and others, "Cytoplasmic BKCa channel intron-containing mRNAs contribute to the intrinsic excitability of hippocampal neurons," *Proceedings of the National Academy of Sciences*, vol. 105, no. 6, pp. 1901–1906, 2008.

[316] Bell, Thomas J and Miyashiro, Kevin Y and Sul, Jai-Yoon and Buckley, Peter T and Lee, Miler T and McCullough, Ron and Jochems, Jeanine and Kim, Junhyong and Cantor, Charles R and Parsons, Thomas D and others, "Intron retention facilitates splice variant diversity in calcium-activated big potassium channel populations," *Proceedings of the National Academy of Sciences*, vol. 107, no. 49, pp. 21 152–21 157, 2010.

[317] Chen, Zhe and Gore, Bryan B and Long, Hua and Ma, Le and Tessier-Lavigne, Marc, “Alternative splicing of the Robo3 axon guidance receptor governs the midline switch from attraction to repulsion,” *Neuron*, vol. 58, no. 3, pp. 325–332, 2008.

[318] Zhong, Xiaoli and Liu, Jinrong R and Kyle, John W and Hanck, Dorothy A and Agnew, William S, “A profile of alternative RNA splicing and transcript variation of CACNA1H, a human T-channel gene candidate for idiopathic generalized epilepsies,” *Human Molecular Genetics*, vol. 15, no. 9, pp. 1497–1512, 2006.

[319] Mansilla, Alicia and López-Sánchez, Carmen and De la Rosa, Enrique J and Garcıa-Martınez, Virginio and Martınez-Salas, Encarna and de Pablo, Flora and Hernández-Sánchez, Catalina, “Developmental regulation of a proinsulin messenger RNA generated by intron retention,” *EMBO Reports*, vol. 6, no. 12, pp. 1182–1187, 2005.

[320] Forrest, Scott T and Barringhaus, Kurt G and Perlegas, Demetra and Hammarskjold, Marie-Louise and McNamara, Coleen A, “Intron retention generates a novel Id3 isoform that inhibits vascular lesion formation,” *Journal of Biological Chemistry*, vol. 279, no. 31, pp. 32 897–32 903, 2004.

[321] *User bulletin: Guidelines for preparing cDNA libraries for isoform sequencing (Iso-Seq (TM))*, `https://www.pacb.com/wp-content/uploads/2015/09/User-Bulletin-Guidelines-for-Preparing-cDNA-Libraries-for-Isoform-Sequencing-Iso-Seq.pdf`.

[322] Khrameeva, Ekaterina E and Gelfand, Mikhail S, “Biases in read coverage demonstrated by interlaboratory and interplatform comparison of 117 mRNA and genome sequencing experiments,” *BMC Bioinformatics*, vol. 13, no. 6, pp. 1–7, 2012.

[323] Königs, Vanessa and de Oliveira Freitas Machado, Camila and Arnold, Benjamin and Blümel, Nicole and Solovyeva, Anfisa and Löbbert, Sinah and Schafranek, Michal and Ruiz De Los Mozos, Igor and Wittig, Ilka and McNicoll, Francois and others, “SRSF7 maintains its homeostasis through the expression of split-ORFs and nuclear body assembly,” *Nature Structural & Molecular Biology*, vol. 27, no. 3, pp. 260–273, 2020.

[324] Heinicke, Laurie A and Nabet, Behnam and Shen, Shihao and Jiang, Peng and van Zalen, Sebastiaan and Cieply, Benjamin and Russell, J Eric and Xing, Yi and Carstens, Russ P, “The RNA binding protein RBM38 (RNPC1) regulates splicing during late erythroid differentiation,” *PloS ONE*, vol. 8, no. 10, e78031, 2013.

[325] Bhatt, Dev M and Pandya-Jones, Amy and Tong, Ann-Jay and Barozzi, Iros and Lissner, Michelle M and Natoli, Gioacchino and Black, Douglas L and Smale, Stephen T, “Transcript dynamics of proinflammatory genes revealed by sequence analysis of subcellular RNA fractions,” *Cell*, vol. 150, no. 2, pp. 279–290, 2012.

[326] Schmitz, Ulf and Pinello, Natalia and Jia, Fangzhi and Alasmari, Sultan and Ritchie, William and Keightley, Maria-Cristina and Shini, Shaniko and Lieschke, Graham J and Wong, Justin JL and Rasko, John EJ, “Intron retention enhances gene regulatory complexity in vertebrates,” *Genome Biology*, vol. 18, no. 1, pp. 1–15, 2017.

[327] Sakabe, Noboru Jo and De Souza, Sandro José, “Sequence features responsible for intron retention in human,” *BMC Genomics*, vol. 8, no. 1, pp. 1–14, 2007.

[328] W. Chen, J. T. Wilson, S. Tyree, K. Q. Weinberger, and Y. Chen, “Compressing Neural Networks with the Hashing Trick,” *arXiv:1504.04788 [cs]*, Apr. 2015. [Online]. Available: `http://arxiv.org/abs/1504.04788` (visited on 04/24/2022).

[329] B. K. Beaulieu-Jones and C. S. Greene, “Reproducibility of computational workflows is automated using continuous analysis,” *Nature biotechnology*, vol. 35, no. 4, pp. 342–346, Apr. 2017, ISSN: 1087-0156. DOI: `10.1038/nbt.3780`. [Online]. Available: `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6103790/` (visited on 03/12/2021).

6 Appendices

A Introduction

A.1 Example `esearch` query

Example query using the `esearch` function provided in the Entrez Programming Utilities [18] software. This queries the GEO Data Sets API for records containing data from the EPIC array (database ID: GPL21145) platform, for which the supplemental files include raw IDAT images. The query uses `esearch`, which returns IDs for samples containing EPIC IDATs. The sample IDs are then piped to `efetch` and `xtract` to determine the number of unique IDs available.

```
esearch -db gds -query 'GPL21145'[ACCN] AND idat[suppFile] AND gsm[ETYP] |
  efetch -format docsum |
  xtract -pattern DocumentSummary -element ID Accession > ./
```

A.2 Example Python script using `esearch` utilities

This Python script calls `esearch`, provided in the Entrez Programming Utilities [18] software, in order to determine sample IDs and study IDs for specific platform IDs. It determines the total IDs and IDs for which an IDAT supplement is available. Note, this script calls `settings`, which defines several global variables including the platform ID, `settings.platformid`, which can be any valid platform ID. Thus this script is modular and can be applied to either of the Illumina Infinium BeadArray platforms. This script appears in the `recountmethylation_server` [193] resource, which is used in `recountmethylation_instance` to access data compiled in the data compilations for `recountmethylation`.

```python
#!/usr/bin/env python3

""" edirect_query.py

    Authors: Sean Maden, Abhi Nellore

    Get files of GSE and GSM ids from GEO via edirect query (NCBI entrez
    utilities function). IDs correspond to valid HM450k array experiments and
    samples.

    Notes:
        * Scheduling: New edirect queries should be scheduled periodically to
          check for latest experiment/sample/file info and to detect novel
          uploaded experiments/samples/files.
        * Filters: The edirect queries (GSE, GSM, and filtered query file) work
          together to form a filter on valid sample and experiment ids whose
          files are to be downloaded and preprocessed.

    Functions:
        * gse_query_diffs: Quickly detect and return differences between two
          edirect query files.
        * gsm_query: Get valid HM450k sample (GSM) IDs, based on presence of
          HM450k platform and availability of raw idat array files in
          supplement.
        * gse_query: Get valid HM450k experiment (GSE) IDs, based on
          specification of the correct platform accession (GPL13534) in the
          experiment annotation. Note, many experiments combine multiple
          platform ids, so it is necessary to filter out non-HM450k array
          samples from experiment GSM ID lists.
        * gsequery_filter: Generate a new edirect query filter file, containing
          valid GSE and GSM IDs for HM450k array experiments/samples with
          idats available in sample supplemental files.
"""

import subprocess, os, socket, struct, sys, time, tempfile, atexit, shutil
import glob, filecmp; from itertools import chain
sys.path.insert(0, os.path.join("recountmethylation_server","src"))
from utilities import gettime_ntp, querydict, getlatest_filepath
import settings
```

```
settings.init()

def gse_query_diffs(query1, query2, rstat=False):
    """ gse_query_diffs

        Compares two GSE query results, returning query file diffs or boolean.

        Arguments:
            * query1 (str) : first edirect query, filename
            * query2 (str) : second edirect query, filename
            * rstat (True/False, bool.) : whether to return boolean only,
                or else return list (default)

        Returns:
            * boolean (T/F) or query diffs (list of GSE IDs). Boolean is 'True'
                if query objects are the same, 'False' otherwise
    """
    difflist = []
    qd1 = querydict(query1) # eg. for first value: qd1[list(qd1.keys())[0]]
    qd2 = querydict(query2)
    # if gse doesn't exist in qd2
    for key in qd1:
        evalkey = ''
        if key in list(qd2.keys()):
            if not qd1[key]==qd2[key]:
                for item in qd1[key]:
                    if not item in qd2[key]:
                        evalkey = False
                for item in qd2[key]:
                    if not item in qd1[key]:
                        evalkey = False
        else:
            evalkey = None
        if not evalkey:
            difflist.append(key)
    for key in qd2:
        if not key in list(qd1.keys()):
            difflist.append(key)
    if rstat:
        if len(difflist)>0:
            return False
        else:
            return True
    else:
        return difflist

def gsm_query(validate=True, timestamp=gettime_ntp()):
    """ gsm_query
        Get GSM level query object, from edirect query.
        Arguments:
            * validate (True/False, bool.) : whether to validate the file after
                ownload.
            * timestamp (str) : NTP timestamp or function to retrieve it.
        Returns:
            * Error (str) or download object (dictionary).
    """
    # timestamp = str(gettime_ntp())
    eqdestpath = settings.equerypath
    temppath = settings.temppath
    os.makedirs(eqdestpath, exist_ok=True)
    os.makedirs(temppath, exist_ok=True)
    temp_make = tempfile.mkdtemp(dir=temppath)
    atexit.register(shutil.rmtree, temp_make)
    dldict = {}
    dldict['gsmquery'] = []
    dlfilename = ".".join(['gsm_edirectquery',timestamp])
    dldict['gsmquery'].append(dlfilename)
    subp_strlist1 = ["esearch","-db","gds","-query",
    "'"+settings.platformid+"[ACCN] AND idat[suppFile] AND gsm[ETYP]'"
    ]
    subp_strlist2 = ["efetch","-format","docsum"]
    subp_strlist3 = ["xtract","-pattern","DocumentSummary",
        "-element","Id Accession",">",
        os.path.join(temp_make,dlfilename)
        ]
    args = " | ".join([" ".join(subp_strlist1),
        " ".join(subp_strlist2),
        " ".join(subp_strlist3)])
    output=subprocess.check_output(args, shell=True)
    dldict['gsmquery'].append(output)
    if validate:
        gsmquery_filewritten = os.path.join(temp_make,dlfilename)
        gsmquery_old = glob.glob('.'.join([os.path.join(eqdestpath, '
   gsm_edirectquery'), '*',]))
        if gsmquery_old:
            if len(gsmquery_old)>1:
                gsmquery_old.sort(key=lambda x: int(x.split('.')[1]))
                gsmquery_old_mostrecent = gsmquery_old[-1]
            else:
                gsmquery_old_mostrecent = gsmquery_old[0]
            # filecmp should work (equesry file order preserved on reps)
            if filecmp.cmp(gsmquery_old_mostrecent,gsmquery_filewritten):
                print("Downloaded gsm query file same as most recent stored."+
                    " Removing..."
                    )
                os.remove(gsmquery_filewritten)
                dldict['gsmquery'].append(False)
            else:
                print("Downloaded file is new, moving to dest...")
                shutil.move(gsmquery_filewritten, os.path.join(
                                eqdestpath, os.path.basename(gsmquery_filewritten)
   )
                            )
                dldict['gsmquery'].append(True)
        else:
            print("Downloaded file is new, moving...")
            shutil.move(gsmquery_filewritten, os.path.join(
                eqdestpath, os.path.basename(gsmquery_filewritten))
                )
            dldict['gsmquery'].append(True)
    return dldict

def gse_query(validate=True, timestamp=gettime_ntp()):
    """ gse_query

        Get GSE level query object from edirect query.

        Arguments:
            * validate (True/False, bool) : Whether to validate the file after
                download.
            * timestamp (str) : NTP timestamp or function to retrieve it.

        Returns:
            * Error (str) or download object (dictionary).
    """
    eqdestpath = settings.equerypath
    os.makedirs(eqdestpath, exist_ok=True)
    temppath = settings.temppath
```

```python
    os.makedirs(temppath, exist_ok=True)
    temp_make = tempfile.mkdtemp(dir=temppath)
    atexit.register(shutil.rmtree, temp_make)
    dldict = {}
    dldict['gsequery'] = []
    dlfilename = ".".join(['gse_edirectquery',timestamp])
    dldict['gsequery'].append(dlfilename)
    subp_strlist1 = ["esearch","-db","gds","-query",
    "'"+settings.platformid+"[ACCN] AND idat[suppFile] AND gse[ETYP]'"
    ]
    subp_strlist2 = ["efetch","-format","docsum"]
    subp_strlist3 = ["xtract","-pattern","DocumentSummary",
        "-element","Id Accession",">",
        os.path.join(temp_make,dlfilename)
        ]
    args = " | ".join([" ".join(subp_strlist1),
        " ".join(subp_strlist2),
        " ".join(subp_strlist3)])
    output=subprocess.check_output(args, shell=True)
    dldict['gsequery'].append(output)
    if validate:
        gsequery_filewritten = os.path.join(temp_make,dlfilename)
        gsequery_old = glob.glob('.'.join([os.path.join(eqdestpath, '
    gse_edirectquery'), '*',]))
        if gsequery_old:
            if len(gsequery_old)>1:
                gsequery_old.sort(key=lambda x: int(x.split('.')[1]))
                gsequery_old_mostrecent = gsequery_old[-1]
            else:
                gsequery_old_mostrecent = gsequery_old[0]
            # get diffs manually (edirect can return id's in different order)
            diffs = gse_query_diffs(query1=gsequery_old_mostrecent,
                query2=gsequery_filewritten,
                rstat=True)
            if diffs:
                print("Downloaded gse query file same as most recent stored."+
                    " Removing..."
                    )
                os.remove(gsequery_filewritten)
                dldict['gsequery'].append(False)
            else:
                print("Downloaded file is new, moving to dest...")
                shutil.move(gsequery_filewritten, os.path.join(
                                eqdestpath, os.path.basename(gsequery_filewritten)
    )
                            )
                dldict['gsequery'].append(True)
        else:
            print("Downloaded file is new, moving...")
            shutil.move(gsequery_filewritten, os.path.join(
                eqdestpath, os.path.basename(gsequery_filewritten))
                )
            dldict['gsequery'].append(True)
    return dldict

def gsequery_filter(splitdelim='\t', timestamp=gettime_ntp()):
    """ gsequery_filter

        Prepare an edirect query file. Filter a GSE query file on its GSM
            membership.

        Arguments:
            * splitdelim (str) : Delimiter to split ids in querydict() call.
            * timestamp (str) : NTP timestamp or function to retrieve it.

        Returns:
            * gsequeryfiltered (list): Filtered GSE query object (list), writes
                filtered query file as side effect.
    """
    eqpath = settings.equerypath
    gsequerystr = settings.gsequerystr
    gsmquerystr = settings.gsmquerystr
    # get GSM list from gsm query file
    gsmqueryf_latestpath = getlatest_filepath(filepath=eqpath,
            filestr=gsmquerystr, embeddedpattern=True, tslocindex=1,
            returntype='returnlist'
        )
    if gsmqueryf_latestpath:
        print("Latest gsmquery file detected: "+str(gsmqueryf_latestpath))
    else:
        print("Error detecting latest gsmquery file! Returning...")
        return
    gsmlines = [line.rstrip('\n') for line in open(gsmqueryf_latestpath[0])]
    gsmlist = [line.split('\t')[1::][0] for line in gsmlines]
    # get GSE dictionary object
    gsequeryf_latestpath = getlatest_filepath(filepath=eqpath, filestr=gsequerystr
    ,
            embeddedpattern=True, tslocindex=1, returntype='returnlist'
        )
    if gsequeryf_latestpath:
        print("Latest gsequery file detected: "+str(gsequeryf_latestpath))
    else:
        print("Error detecting latest gsequery file! Returning...")
        return
    gsed_obj = querydict(querypath=gsequeryf_latestpath[0], splitdelim='\t')
    gsefiltl = []
    for gsekey in list(gsed_obj.keys()):
        samplelist_original = gsed_obj[gsekey]
        samplelist_filt = [sample for sample in samplelist_original
            if sample in gsmlist
        ]
        if samplelist_filt and len(samplelist_filt)>0:
            gsefiltl.append(' '.join([gsekey,' '.join(samplelist_filt)]))
    print('writing filt file...')
    if eqpath:
        filtfn = ".".join(["gsequery_filt",timestamp])
        with open(os.path.join(eqpath, filtfn), 'w') as filtfile:
            for item in gsefiltl:
                filtfile.write("%s\n" % item)
    return gsefiltl

if __name__ == "__main__":
    """ Run a new EDirect query

    Query the GEO DataSets API for valid data run using the target platform.

    """
    print("Beginning EDirect query...")
    equery_dest = settings.equerypath; temppath = settings.temppath
    gse_query(); gsm_query(); gsequery_filter()
```

A.3 Example metadata preprocessing script

This R script preprocesses metadata obtained from GEO prior metadata mapping using the postprocessing script (Appendix A.4). Briefly, this script recognizes key-value pairs from JSON-formatted files containing sample metadata extracted from downloaded SOFT-formatted files. This script is run as part of the metadata harmonization process for recountmethylation data compilation files. It is implemented in the recountmethylation.pipeline [194] R package.

```r
#!/usr/bin/env Rscript

# Author: Sean Maden
#
# Preprocess available sample metadata
#

#------------------------
# md preprocess functions
#------------------------

# define strings to search for variables

#' Terms vector for variable tissue
#'
#' Terms to seed regex pattern matching for md_preprocess().
#' @seealso md_preprocess(); md_postprocess()
#' @return Terms vector
#' @export
mdpre_vars_tissue <- function(){
  cname.tissue <- c("tissue", "tissu", "sample type", "sample_type",
                    "sample.type", "cell line", "cell_line", "cell.line",
                    "cell type", "cell_type", "cell.type", "tissue type",
                    "tissue_type", "tissue.type", "tissue region",
                    "tissue_region", "tissue.region", "histology",
                    "region", "brnum", "location")
  return(cname.tissue)
}

#' Terms vector for variable disease
#'
#' Terms to seed regex pattern matching for md_preprocess().
#' @seealso md_preprocess(); md_postprocess()
#' @return Terms vector
#' @export
mdpre_vars_disease <- function(){
  cname.disease_state <- c("disease state", "disease_state", "disease.state",
                           "subject status", "subject_status",
                           "subject.status", "sample group", "sample_group",
                           "sample.group", "diagnosis", "group", "condition",
                           "disease", "subgroup", "status")
  return(cname.disease_state)
}

#' Terms vector for variable info
#'
#' Terms to seed regex pattern matching for md_preprocess().
#' @seealso md_preprocess(); md_postprocess()
#' @return Terms vector
#' @export
mdpre_vars_info <- function(){
  cname.info <- c("state", "passages", "race", "individual",
                  "race", "stage", "condition", "risk", "twinid", "smok",
                  "cigarette", "pack year", "pack_year", "pack.year",
                  "pack yr", "pack_yr", "pack.yr", "drinks", "drink year",
                  "drink_year", "drink.year", "drink yr", "drink_yr",
                  "drink.yr", "drug use", "drug_use", "drug.use",
                  "alcohol", "treatment", "material", "run", "batch",
                  "plate", "developmental stage", "developmental_stage",
                  "developmental.stage", "source", "storage", "zygosity",
                  "family", "weight", "bmi", "drug", "intervention")
  return(cname.info)
```

```r
}

#' Terms vector for variable sex
#'
#' Terms to seed regex pattern matching for md_preprocess().
#' @seealso md_preprocess(); md_postprocess()
#' @return Terms vector
#' @export
mdpre_vars_sex <- function(){cname.sex <- c("sex", "gender");return(cname.sex)}

#' Terms vector for variable age
#'
#' Terms to seed regex pattern matching for md_preprocess().
#' @seealso md_preprocess(); md_postprocess()
#' @return Terms vector
#' @export
mdpre_vars_age <- function(){
  cname.age <- c("age", "passage", "age (years)", "age_(years)", "age.(years)")
  return(cname.age)
}

# get list of search strings

#' Get the regex patterns for variable mappings
#'
#' Gets the regex patterns from vectors of seed terms for each variable mapped
#' by md_preprocess().
#' @seealso md_preprocess(); md_postprocess()
#' @return Returns list containing regex patterns for each mapped variable.
#' @export
mdpre_vars <- function(){
  cl <- lapply(list(mdpre_vars_tissue(), mdpre_vars_disease(),
                    mdpre_vars_sex(), mdpre_vars_age(),
                    mdpre_vars_info()), get_pstr);
  names(cl) <- c("sample_type", "disease_state", "sex", "age", "info")
  return(cl)
}

#' Preprocess sample metadata
#'
#' Preprocess sample metadata by coercing JSON-formatted metadata files
#' into a flat matrix, with regex pattern matching to identify and
#' categorize various variable types. Additional data such as the sample
#' titles contained in the file declared by the titles.fn argument.
#'
#' @param ts Timestamp for the preprocessed metadata table to output
#' (integer or character).
#' @param mdpre.fn Name of preprocessed metadata table file to output
#' ("md_preprocess").
#' @param md.dname Name of directory, in files.dname, containing the instance
#' metadata files ("metadata).
#' @param titlesfn.str Name of sample/GSM titles file ("gsm_jsontitledf).
#' @param atablefn.str Name of study annotation tables file
#' ("geo_gse-atables_list")
#' @param files.dname Main recountmethylation instance files directory
#' ("recount-methylation-files").
#' @param verbose Whether to show status messages (TRUE).
#' @return NULL, produces the mdpre preprocessed metadata table.
#' @seealso md_postprocess()
#' @export
md_preprocess <- function(ts, mdpre.fn = "md_preprocess",
                          md.dname = "metadata",
                          titlesfn.str = "gsm_jsontitledf",
                          atablefn.str = "geo_gse-atables_list",
                          files.dname = "recount-methylation-files",
                          verbose = TRUE){
  if(verbose){message("Loading GSM data files...")}
  md.dpath <- file.path(files.dname, md.dname)
  md.lf<-list.files(md.dpath);md.lf<-md.lf[grepl(paste0(".*",ts,".*"),md.lf)]
  at.fname <- md.lf[grepl(paste0(atablefn.str, ".*"), md.lf)][1]
  titles.fname <- md.lf[grepl(paste0(titlesfn.str, ".*"), md.lf)][1]
  if(!file.exists(file.path(md.dpath, at.fname))){
    stop("Study anno. tables file not found at ", at.fpath)}
  if(!file.exists(file.path(md.dpath, titles.fname))){
    stop("GSM titles file not found at ",titles.fpath)}
  tls <- get(load(file.path(md.dpath, at.fname)))
  tdf <- get(load(file.path(md.dpath, titles.fname)))
  message("Processing GSE flat files...")
  colv <- c("gsm","gse","sample_type","disease_state","sex","age")
  gat.all <- matrix(nrow = 0, ncol = length(colv));colnames(gat.all) <- colv
  cl <- mdpre_vars(); message("Appending tables data...")
  for(gseid in names(tgse.list)){
    gsedat <- tgse.list[[gseid]]
    gati <- data.frame(gsm = gsedat[,1], gse=gsedat[,2],stringsAsFactors=FALSE)
    for(cn in names(cl)){
      cnvar <- rep("NA", nrow(gsedat))
      cn.dat <- rep("NA", nrow(gsedat));gsedat.rep <- gsedat;cn.cv <- cl[[cn]]
      cname.filt <- grepl(cn.cv, colnames(gsedat.rep))
      if(length(which(cname.filt)) > 0){
        if(cn %in% c("age", "info")){ # parse age mapping logic, excluding fp's
          cmv <- colnames(gsedat.rep)[cname.filt]
          if(verbose){message("Removing info matches (e.g. stage columns...)")}
          if(cn == "age"){
            cmv.info <- grepl(cl[["info"]], colnames(gsedat.rep))
            cmv <- colnames(gsedat.rep)[cname.filt & !cmv.info]}
          if(verbose){message("Appending colnames to row entries...")}
          for(colnamei in cmv){
            gsedat.rep[,colnamei]<-paste0(colnamei,":",gsedat.rep[,colnamei])}
        };gf <- gsedat.rep[, cname.filt, drop = FALSE]
        cnvar <- as.character(apply(gf,1,paste,collapse = ";"))
      };gati[,ncol(gati) + 1] <- cnvar; colnames(gati)[ncol(gati)] <- cn
    };gat.all <- rbind(gat.all, gati);message("Finished study: ", gseid)
  };gat.all <- gat.all[!duplicated(gat.all[,1]),]
  message("Appending GSM titles...")
  d1 <- gat.all; d2 <- gsmtitledf;gsm.all <- unique(c(d1[,1], d2[,1]))
  gsm1 <- gsm.all[!gsm.all %in% d1[,1]];gsm2 <- gsm.all[!gsm.all %in% d2[,1]]
  if(length(gsm1) > 0){
    nav <- rep(rep("NA", length(gsm1)), ncol(d1) - 1)
    mna <- matrix(c(gsm1, nav), nrow = length(gsm1), ncol = ncol(d1))
    colnames(mna) <- colnames(d1); d1 <- rbind(d1, mna)}
  if(length(gsm2) > 0){
    nav <- rep(rep("NA", length(gsm2)), ncol(d2) - 1)
    mna <- matrix(c(gsm2, nav), nrow = length(gsm2), ncol = ncol(d2))
    d2 <- rbind(d2, mna)}
  if(nrow(d2) > nrow(d1)){d2 <- d2[d2[,1] %in% d1[,1] & !duplicated(d2[,1]),]}
  match.gsm1 <- match(as.character(d1[,1]), as.character(d2[,1]))
  order.gsm1 <- order(match.gsm1);d1 <- d1[order.gsm1,]
  match.gsm2 <- match(as.character(d2[,1]), as.character(d1[,1]))
  order.gsm2 <- order(match.gsm2);d2 <- d2[order.gsm2,]
  cond <- identical(as.character(d2[,1]), as.character(d1[,1]))
  if(cond){
    d1 <- as.data.frame(d1, stringsAsFactors = FALSE)
    d1$gsm_title <- as.character(d2[,2])}
  mdpre.fpath <- file.path(md.dpath, paste0(mdpre.fn, "_", ts, ".rda"))
  message("Saving mdpre at ", mdpre.fpath, "...")
  mdpre <- d1; save(mdpre, file = mdpre.fpath); return(NULL)
}
```

A.4 Example metadata postprocessing script

This R script postprocesses metadata obtained from GEO after metadata preprocessing (Appendix A.3). Briefly, this script maps uniformly formatted terms under informative variables for tissue, disease, etc. It takes as input the flat files produced from preprocessing JSON files containing sample metadata. In addition to pattern recognition, there are several sections using logic to map specific attributes such as time units for mined ages. This script is run as part of the metadata harmonization process for `recountmethylation` data compilation files. It is implemented in the `recountmethylation.pipeline` [194] R package.

```
#!/usr/bin/env Rscript

# Author: Sean Maden
#
# Preprocess available sample metadata
#

# require(data.table); require(rjson)

#-------------------------
# md postprocess functions
#-------------------------
# disease terms (general, non-cancer)
#' Disease terms for disease variable
#'
#' List containing search terms and string queries for regex pattern matching.
#'
#' @return List of terms (names) and string queries (values).
#' @export
md_post_disease <- function(){
  dxl <- list("acute" = c("acute"), "syndrome" = c("syndrome"),
              "disorder" = c("disorder"), "inflam"= c("inflammation"),
              "cancer" = c("cancer"), "case" = c("case"),
              "normal" = c("normal"),
              "healthy" = c("healthy"), "replicate" = c("replicate"),
              "control" = c("control", "CONTROL", "ctl", "CTL", "ctrl",
                            "CTRL"),
              "psychosis" = c("psychosis", "psychotic"),
              "schizophrenia" = c("schizophrenia"),
              "arthritis" = c("osteoarthritis", "arthritis", "rheumatoid",
                              "psoriatic", "fibromyalgia", "gout"),
              "rheumatoid_arthritis" = c("rheumatoid arthritis"),
              "osteoarthritis" = c("osteoarthritis"),
              "psoriatic_arthritis" = c("psoriatic arthritis"),
              "genetic_disorder" = c("fragile x", "cystic fibrosis", "duane",
                                     "polycystic", "chrons", "hemophelia",
                                     "haemophelia", "hemochromatosis",
                                     "huntington's", "huntingtons",
                                     "thalassemia", "tay sachs", "tay sach",
                                     "parkinson's", "parkinsons",
                                     "sickle cell", "marfan"),
              "parkinsons" = c("parkinsons", "parkinson's"),
              "sickle_cell" = c("sickle cell"),
              "anemia" = c("anemia", "sickle cell"),
              "alzheimers" = c("alzheimer", "alzheimer"),
              "dementia" = c("dementia"), "lewy_body" = c("lewy bod"),
              "cystic_fibrosis" = c("cystic fibrosis"),
              "scoliosis" = c("scoliosis"),
              "obese" = c("obese"),
              "irritable_bowel_disease" = c("IBD", "ibd", "irritable bowel"),
              "lesions" = c("lesions"),
              "insulin_resistance" = c("insulin resist"),
              "autism" = c("autism", "autistic"), "patient" = c("patient"))
  return(dxl)
}

```

```r
# cancer tissue terms/generic cancer terms
#' Cancer terms for tissue variable
#'
#' List containing search terms and string queries for regex pattern matching.
#'
#' @return List of terms (names) and string queries (values).
#' @export
md_post_cancer <- function(){
  cxl <- list("cancer" = c("tumor", "tumour", "metasta", "carcinoma",
                            "sarcoma", "neoplas", "adenoma", "cancroid"),
              "tumor" = c("tumor", "tumour"),"metastasis" = c("metasta"),
              "carcinoma" = c("carcinoma"),"sarcoma" = c("sarcoma"),
              "neoplasia" = c("neoplas"), "adenoma" = c("adenoma"),
              "adenocarcinoma" = c("adenocarcinoma"), "lepidic" = c("lepidic"),
              "blastoma" = c("blastoma"), "benign" = c("benign"))
  return(cxl)
}

# cancer types by tissue/location
#' Cancer terms list for tissue variable
#'
#' List containing search terms and string queries for regex pattern matching.
#'
#' @return List of terms (names) and string queries (values).
#' @export
md_post_cancertype <- function(){
  cxsubl <- list("skin_cancer" = c("melanoma", "skin cancer"),
                 "brain_cancer" = c("glioblastoma", "astrocytoma",
                                    "brain cancer",
                                    "medulloblastoma"),
                 "medulloblastoma" = c("medulloblastoma"),
                 "glioblastoma" = c("glioblastoma"),
                 "breast_cancer" = c("breast lobular carcinoma",
                                     "breast ductal carcinoma",
                                     "breast cancer", "triple negative"),
                 "colorectal_cancer" = c("colorectal adeno", "colon cancer",
                                         "colorectal cancer", "rectal cancer"),
                 "stomach_cancer" = c("stomach adeno", "stomach cancer",
                                      "gastric cancer", "gastric adeno"),
                 "esophageal_cancer" = c("esophageal carcinoma",
                                         "esophageal adeno",
                                         "esophageal squamous cell carcinoma",
                                         "oesophageal carcinoma",
                                         "oesophageal adeno",
                                         "oesophageal squamous cell carcinoma",
                                         "esophageal cancer",
                                         "oesophageal cancer",
                                         " EAC$"),
                 "nerve_cell_cancer" = c("paraganglioma", "ependymoma",
                                         "nerve cancer",
                                         "nerve cell cancer",
                                         "schwannoma"),
                 "paraganglioma" = c("paraganglioma"),
                 "ovarian_cancer" = c("ovarian serous carcinoma",
                                      "ovarian cancer",
                                      "endometrioid",
                                      "ovarian epithelial cancer"),
                 "uterine_cancer" = c("uterine carcinosarcoma",
                                      "uterine cancer",
                                      "uterine corpus endometrial carcinoma",
                                      "endometrial carcinoma",
                                      "uterine serous carcinoma",
                                      "uterine papillary serous carcinoma"),
                 "kidney_cancer" = c("oncocytoma",
                                     "clear cell renal cell carcinoma",
                                     "chromophobe renal cell carcinoma",
                                     "renal cancer", "kidney cancer",
                                     "kidney papillary carcinoma"),
                 "thyroid_cancer" = c("thyroid carcinoma", "thyroid cancer"),
                 "lung_cancer" = c("lung adenocarcinoma",
                                   "lung squamous cell carcinoma",
                                   "lung cancer",
                                   "non-mucinous bronchoalveolar carcinoma",
                                   "lepidic-predominant adenocarcinoma",
                                   "LPA"),
                 "bladder_cancer" = c("invasive urothelial bladder cancer",
                                      "bladder cancer"),
                 "prostate_cancer" = c("prostate adenocarcinoma",
                                       "prostate cancer"),
                 "liver_cancer" = c("liver hepatocellular carcinoma",
                                    "hepatoblastoma",
                                    "cholangiocarcinoma",
                                    "liver angiosarcoma",
                                    "liver cancer"),
                 "thymus_gland_cancer" = c("thymoma", "thymus cancer",
                                           "thymus gland cancer",
                                           "thymic cancer"),
                 "testicular_cancer" = c("testicular germ cell cancer",
                                         "testicular cancer"),
                 "pancreatic_cancer" = c("pancreatic ductal adenocarcinoma",
                                         "pancreatic cancer"),
                 "cervical_cancer" = c("cervical squamous cell carcinoma",
                                       "cervical cancer",
                                       "cervical squamous cell adenocarcinoma"
                                       ),
                 "eye_cancer" = c("uveal melanoma", "uveal lymphoma",
                                  "intraocular cancer", "retinoblastoma",
                                  "retinal cancer"))
  return(cxsubl)
}

# leukemia disease terms
#' Leukemia terms list for disease variable
#'
#' List containing search terms and string queries for regex pattern matching.
#'
#' @return List of terms (names) and string queries (values).
#' @export
md_post_leukemia <- function(){
  leukl <- list("leukemia" = c("leukemia", "chronic leuk", "chronic myelo",
                               "acute leuk", "acute lympho", "acute myel",
                               "cml", "CML", "aml", "AML", "ALL"),
                "acute_leukemia" = c("acute leuk","acute lympho", "acute myel",
                                     "aml", "AML", "ALL"),
                "acute_myeloid_leukemia" = c("acute myel", "aml", "AML"),
                "acute_lymphoblastic_leukemia" = c("acute lympho", "ALL"))
  return(leukl)
}

# tissue terms, blood and related
#' Blood terms list for tissue variable
#'
#' List containing search terms and string queries for regex pattern matching.
#'
#' @return List of terms (names) and string queries (values).
#' @export
md_post_tissue_blood <- function(){
  txl <- list("blood" = c("blood", "hematopoiet", "haematopoiet","lymphoid",
```

```r
                          "myeloid", "natural killer", "( |^)nk( |$)",
                          "( |^)NK( |$)", "erythrocyte", "mast cell",
                          "myeloblast", "plasma","monocyte", "lymphocyte",
                          "eosinophil", "neutrophil","basophil", "macrophage",
                          "megakaryocyte", "thrombocyte","wbc", "WBC", "rbc",
                          "RBC","bcell", "b cell", "tcell", "t cell", "cd4",
                          "cd5", "cd8", "cd34", "CD4", "CD5", "CD8", "CD34",
                          "cytotoxic", "helper", "peripheral blood leukocytes",
                          "( |^)PBL( |$|:)", "pbmc", "( |^)PBMC( |$|:)",
                          "buffy", "blood spot", "blood punch", "granulocyte",
                          "white blood cell"),
              "buffy_coat" = c("buffy"),
              "whole_blood" = c("whole blood"),
              "peripheral_blood" = c("peripheral blood"),
              "cord_blood" = c("cord blood"), "blood_spot" = c("blood spot"),
              "white_blood_cell" = c("wbc", "WBC", "white blood cell",
                                     "monocyte", "lymphocyte", "eosinophil",
                                     "neutrophil", "basophil","bcell",
                                     "b cell", "tcell", "t cell", "cd4",
                                     "cd5", "cd8", "cd34", "granulocyte",
                                     "CD4", "CD5", "CD8", "CD34", "cytotoxic",
                                     "helper", "pbmc", "PBMC"),
              "peripheral_blood_mononuclear_cells" = c("pbmc", "PBMC"),
              "peripheral_blood_leukocytes" = c("peripheral blood leukocytes",
                                                "( |^)PBL( |$|:)"),
              "cd4" = c("cd4", "CD4"), "cd5" = c("cd5", "CD5"),
              "cd8" = c("cd8", "CD8"), "cd34" = c("cd34", "CD34"),
              "granulocyte" = c("granulocyte"), "monocyte" = c("monocyte"),
              "lymphocyte" = c("lymphocyte"), "neutrophil" = c("neutrophil"),
              "eosinophil" = c("eosinophil"), "basophil" = c("basophil"),
              "t_cell" = c("tcell", "t cell", "cd4", "cd5", "cd8",
                           "cd34", "cytotoxic", "helper"))
  return(txl)
}

# tissue terms, general, not blood
#' Non-blood terms list for tissue variable
#'
#' List containing search terms and string queries for regex pattern matching.
#'
#' @return List of terms (names) and string queries (values).
#' @export
md_post_tissue <- function(){
  which.var <- c("gsm_title", "sample_type")
  txl <- list("adjacent" = c("adjacent"), "matched" = c("match"),
              "distal" = c("distal"), "medullary" = c("medullary"),
              "cultured" = c("cultured"), "paired" = c("paired"),
              "explant" = c("explant"), "biopsy" = c("biopsy"),
              "clone" = c("clone", "clonal"),
              "subclone" = c("subclone", "subclonal"),
              "resection" = c("resection"), "xenograft" = c("xenograft"),
              "cells" = c("cells"), "cell_line" = c("cell line"),
              "peripheral" = c("peripheral"), "whole" = c("whole"),
              "organoid" = c("organoid"),
              "parenchyma" = c("parenchyma", "parenchyme"),
              "mesenchyme" = c("mesenchyme", "mesoderm"),
              "mesoderm" = c("mesoderm"),
              "ectoderm" = c("ectoderm"), "endoderm" = c("endoderm"),
              "gland" = c("gland"), "acinus" = c("acinus", "acinary"),
              "muscle" = c("muscle", "brachii", "ulnaris", "minimus",
                           "gemellus", "gluteus", "bicep", "rhomboids",
                           "tongue", "splenius", "capitis"),
              "smooth_muscle" = c("smooth muscle", "smoothe muscle"),
              "skeletal_muscle" = c("skeletal muscle"),
              "bone" = c("bone", "femur", "patella", "tibia", "fibula",
                         "clavicle", "scapula", "humeral", "flexor",
                         "supinator", "radius", "ulna", "humerus",
                         "carpus", "metacarpus", "phalanges", "marrow"),
              "marrow" = c("marrow"), "knee" = c("knee"), "head" = c("head"),
              "leg" = c("leg"), "arm" = c("arm"),
              "thorax" = c("thorax"), "chest" = c("chest"),
              "spine" = c("spine", "spinal"), "foot" = c("foot"),
              "hand" = c("hand"),
              "skin" = c("cutaneous", "skin", "melanocyte"),
              "kidney" = c("kidney", "renal", "abdominal gland", "nephron"),
              "corpuscule" = c("corpuscule"), "tubule" = c("tubule"),
              "gallbladder" = c("gallbladder", "gall bladder", "gall-bladder"),
              "saliva" = c("saliva", "sputum"), "sputum" = c("sputum"),
              "mucus" = c("mucus"), "mucinous" = c("mucinous"),
              "fiber" = c("fiber", "fibrous"), "cartilage" = c("cartilage"),
              "joint" = c("joint"),
              "heart" = c("cardiac", "heart", "superior vena cava", "aorta",
                          "pulmonary artery", "pulmonary vein", "atrium",
                          "pulmonary valve", "ticuspid valve",
                          "inferior vena cava", "mitral valve",
                          "aortic valve", "ventricle"),
              "esophagus" = c("esophag", "oesophag"),
              "stomach" = c("stomach", "gastric"),
              "barretts" = c("( |^)BE( |$|:)", "barretts", "barrett's"),
              "dysplasia" = c("( |^)(H|L)GD( |$|:)", "dysplasia"),
              "low_grade" = c("low grade"), "high_grade" = c("high_grade"),
              "squamous" = c("( |^)SQ( |$|:)", "squamous"),
              "colorectal" = c("colorec"), "intestine" = c("colorec"),
              "colon" = c("colon", "colorec", "large intestine", "cecum"),
              "intestine" = c("colon", "colorec", "large intestine", "cecum"),
              "rectum" = c("colorec", "rectal", "rectum", "anus"),
              "respiratory_system" = c("lung", "bronchi", "alveol",
                                       "interstiti", "pleura",
                                       "trachea", "windpipe", "wind pipe",
                                       "bronchi", "airway"),
              "lung" = c("lung", "bronchi", "alveol", "interstiti", "pleura"),
              "alveolar" = c("alveolar"), "lepidic" = c("lepidic"),
              "windpipe" = c("trachea", "windpipe", "wind pipe", "bronchi",
                             "airway"),
              "nervous_system" = c("astrocyte", "oligodendrocyte", "ependymal",
                                   "schwann", "satellite cell", "glia"),
              "liver" = c("liver", "hepato", "kupff"),
              "bladder" = c("bladder", "urothel"),
              "brain" = c("brain", "cerebrum", "cerebral", "cerebellum",
                          "cerebelli", "dorsolat", "medulla", "lobe",
                          "prefront", "occipital", "falx", "meningeal",
                          "supratentorial", "fossa", "sellar", "grey matter",
                          "gray matter", "white matter", "tentorium",
                          "tentorial", "cortex", "hippocampus"),
              "frontal_lobe" = c("frontal lobe"),
              "frontal_cortex" = c("frontal cortex"),
              "parietal_lobe" =c("parietal lobe"),
              "occipital_lobe" = c("occipital lobe"),
              "prefrontal_lobe" = c("prefront"),
              "brainstem" = c("brain stem", "brainstem"),
              "hippocampus" =c("hippocampus"),
              "cerebellum" = c("cerebellum"),
              "cerebrum" = c("cerebrum", "cerebral"),
              "cortex" = c("cortex"), "white_matter" = c("white matter"),
              "gray_matter" = c("gray matter", "grey matter"),
              "cortex" = c("cortex"),
              "occipital" = c("occipital"),
              "placenta" = c("chorion", "villus", "placent"),
```

```r
              "umbilical_cord" = c("umbilical", "cord blood"),
              "uterus" = c("uterus", "uteri", "endometr"),
              "ovary" = c("ovary", "ovari", "endometrium", "endometrioid"),
              "fallopian_tube" = c("fallop"), "prostate" = c("prostate"),
              "neck" = c("neck", "thyroid"), "thyroid_gland" = c("thyroid"),
              "adrenal" = c("adrenal"),
              "eye" = c("eye", "uvea", "optic nerve", "cone", "rod", "retina"),
              "endocrine_system" = c("endocrine", "pineal", "pituitary",
                                     "pancreas", "pancreat", "adren",
                                     "thyroid", "hypothalamus",
                                     "adrenal cortex", "adreno",
                                     "paraganglioma", "paraganglioma",
                                     "pheochromocytoma", "zona",
                                     "glomerulosa", "fasciculata",
                                     "reticularis", "ovary", "ovari",
                                     "testic", "teste"),
              "pancreas" = c("pancreas", "pancreat"),
              "skin" = c("skin", "epidermis", "keratinocyt"),
              "keratinocyte" = c("keratinocyt"), "breast" = c("breast"),
              "lymphatic_system" = c("lymph", "spleen", "thymus"),
              "oral" = c("mouth", "buccal", "labial", "lip", "tongue",
                         "lingual", "throat", "masticatory"),
              "throat" = c("throat"),
              "buccal" = c("buccal", "cheek swab", "mouth swab"),
              "neuron" = c("neuro", "neural", "nerve", "dendrite", "axon"),
              "glia" = c("glia"), "epithelial" = c("epithel"),
              "endothelium" = c("endothel"),
              "stem_cell" = c("stem cell", "pluripot", "ipsc", "iPSC"),
              "induced_pluripotent_stem_cell" = c("ipsc", "iPSC"),
              "fibroblast" = c("fibroblast"),
              "crypt" = c("crypt"),"ectoderm" = c("ectoderm"),
              "mucosa" = c("mucosa"),
              "primed" = c("primed"),
              "nasal" = c("nasal", "nose", "septum", "sinus"),
              "sperm" = c("sperm", "semen"), "gamete" = c("gamete"),
              "adipose" = c("adipose", "fat", "visceral")); return(txl)
}

# storage condition terms
#' Storage condition terms for storageinfo variable
#'
#' List containing search terms and string queries for regex pattern matching.
#'
#' @return List of terms (names) and string queries (values).
#' @export
md_post_storage <- function(){
  sll <- list("frozen" = c("FF$", "frozen", "frzn", "fzn"),
              "fresh_frozen" = c("FF$", "frozen", "frzn", "fzn"),
              "FF" = c("FF$", "frozen", "frzn", "fzn"),
              "formalin_fixed_paraffin_embedded" = c("FFPE", "formalin"),
              "FFPE" = c("FFPE", "formalin")); return(sll)
}

#' Age terms to seed regex queries
#'
#' Age terms to seed regex queries, called by md_post_handle_age().
#'
#' @return List of terms (names) and string queries (values).
#' @export
md_age_infol <- function(){
  ageinfol <- list("adult" = c("adult", "old", "senior"),
                   "young" = c("neonatal", "pediatric", "prepubescent",
                               "youth","young","child","infant","postnate"),
                   "prenatal"=c("embryo","embryonic","prenatal","prenate"),
                   "fetal" = c("fetal", "foetal", "fetus"),
                   "neonatal" = c("neonate", "neonatal"),
                   "maternal" = c("maternal", "mother"));return(ageinfol)
}

#' Age unit terms to seed regex queries
#'
#' Age unit terms to seed regex queries, called by md_post_handle_age().
#'
#' @return List of unit term labels (names) and string query seed terms
#' (values).
#' @export
md_age_unitl <- function(){
  ageunitl <- list("years" = c("year", "yr", "y ", "(0-9)y.*"),
                   "months" = c("month", "mo"), "weeks" = c("week", "wk"),
                   "days" = c("day", "dy"), "passage" = c("passage"))
  return(ageunitl)
}

# age terms and logic handling
#' Handle age term mappings for metadata postprocessing
#'
#' This function is called by md_postprocess() in order to handle the logic of
#' age term mappings from preprocessed metadata.
#'
#' @param mdpre Table of preprocessed metadata.
#' @param mdpost Table of postprocessed metadata.
#' @param mdpost.vl List mapping term categories (names) to column
#' names in the mdpost postprocessed metadata matrix to be generated.
#' @param mdpre.vl List mapping term categories (names) to column
#' names in the mdpre preprocessed metadata matrix.
#' @param verbose Whether to show status messages (TRUE).
#' @seealso md_postprocess(); md_preprocess()
#' @return Postprocessed metadata table with a new column of mapped age
#' info, values.
#' @export
md_post_handle_age <- function(mdpre, mdpost, mdpost.vl = list("age" = "age"),
                               mdpre.vl = list("sample_id" = "gsm",
                                               "sample_title" = "gsm_title",
                                               "info" = "info","age" = "age",
                                               "age_temp" = "age_temp"),
                               verbose = TRUE){
  ap <- mdpre[,mdpre.vl[["age"]]]
  if(verbose){message("Getting filtered, formatted ages...")}
  av <- unlist(lapply(ap, function(x){
    xsplit.num.form <- "NA"
    if(!x == "NA"){
      xval <- unlist(strsplit(x, ";")) # split values
      xvf <- xval[grepl("[0-9]", xval)][1] # catch first numeric value
      xsplit = unlist(strsplit(xvf, "[a-zA-Z]+"))
      xsplit.num <- xsplit[grepl("[0-9]+", xsplit)][1] # catch 1st numeric
      symv <- "[a-zA-Z]| |:|\\)|\\(|/|!|?|\\_|+|," # replace remaining symbols
      xsplit.num.form <- gsub(symv, "", xsplit.num)};return(xsplit.num.form)
  })); av[av == ""] <- "NA"
  mdpost[,mdpost.vl[["age"]]] <- paste0("age_val:", as.character(av))
  if(verbose){message("Getting filtered age data as new variable...")}
  age.val.filt <- unlist(lapply(ap, function(x){
    xvf <- "NA"
    if(!x == "NA"){
      xval <- unlist(strsplit(x, ";")); cond1 <- grepl("[0-9]", xval)
      cond2 <- grepl("age", xval) & !grepl("stage", xval) # catch age tag
      xvf <- paste(xval[(cond1|cond2)], collapse = ";")};return(xvf)}))
  agetemp.cname <- mdpre.vl[["age_temp"]]; mdpre[,ncol(mdpre) + 1] <- "NA"
  colnames(mdpre)[ncol(mdpre)] <- agetemp.cname
```

```r
  if(!is.null(age.val.filt)){mdpre[,agetemp.cname] <- age.val.filt}
  if(verbose){message("Adding age metadata...")};ageunitl <- md_age_unitl()
  gf.run <- rep(FALSE, nrow(mdpre))
  for(term in names(ageunitl)){ # allows first units match
    gf.rep <- get_filt(v = get_pstr(v = ageunitl[[term]]), m = mdpre,
                       varl = agetemp.cname)
    mdpost[,mdpost.vl[["age"]]] <- appendvar(var=mdpost.vl[["age"]],
                                              val=paste0("age_units:",term),
                                               filtv=gf.rep & !gf.run,
                                               m = mdpost)
    gf.run <- gf.run|gf.rep};if(verbose){message("Adding age group info...")}
  ageinfol <- md_age_infol(); lgf = list()
  age.cname <- mdpre.vl[["age"]]; title.cname <- mdpre.vl[["sample_title"]]
  which.var <- c(mdpre.vl[["age"]], mdpre.vl[["sample_title"]])
  for(term in names(ageinfol)){
    age.var <- get_pstr(v = ageinfol[[term]])
    lgf[[term]][[age.cname]]<-get_filt(v=age.var,m=mdpre,varl=which.var)
    title.var <- get_pstr(v = ageinfol[[term]])
    lgf[[term]][[title.cname]]<-get_filt(v=title.var,m=mdpre,varl=which.var)}
  if(verbose){message("Handling age info map logic...")};termv<-names(ageinfol)
  for(r in seq(nrow(mdpost))){
    sv.term <- "NA"; bool.term.age <- bool.term.title <- c()
    for(t in termv){
      bool.term.age <- c(bool.term.age, lgf[[t]][[age.cname]][r])
      bool.term.title <- c(bool.term.title, lgf[[t]][[title.cname]][r])}
    tf.age<-termv[which(bool.term.age)];tf.title<-termv[which(bool.term.title)]
    sv.term <- ifelse(length(tf.age) == 1, tf.age,
                      ifelse(length(tf.title) == 1, tf.title, "NA"))
    info.val <- paste0("age_info:", sv.term)
    mdpost[,mdpost.vl[["age"]]][r] <- paste0(mdpost[,mdpost.vl[["age"]]][r],
                                             ";", info.val)}
  return(mdpost)
}

# main postprocess function

#' Postprocess metadata prepared using md_preprocess()
#'
#' Perform postprocessing of previously prepreocessed sample metadata. This
#' produces the new harmonized variables (specified by arg mdpost.vl),
#' where harmonization means terms are mapped, lowercase, and "_" separated.
#' Variable entries can include multiple terms separated by ";". The args
#' mdpre.vl and mdpost.vl specify the various variable titles in
#' the preprocess and postprocess dataset. The vars disease.search.vars,
#' tissue.search.vars, and storage.info.vars specify the mdpre variables to
#' search for disease, tissue, and storage info mappings.
#'
#' @param ts The timestamp for this run.
#' @param mdpre The matrix containing preprocessed metadata (returned from
#' md_preprocess()).
#' @param mdpost.fname Filename for newly mapped postprocessed metadata.
#' @param md.dpath Path to the directory containing the preprocessed
#' metadata matrix, where newly postprocessed metadata will be stored.
#' @param mdpre.vl List mapping term categories (names) to column
#' names in the mdpre preprocessed metadata matrix.
#' @param mdpost.vl List mapping term categories (names) to column
#' names in the mdpost postprocessed metadata matrix to be generated.
#' @param disease.search.vars Term categories to search in preprocessed
#' metadata for the disease term mappings
#' @param tissue.search.vars Term categories in preprocessed metadata to
#' search for tissue term mappings.
#' @param storage.info.vars Term categories in preprocessed metadata to
#' search for storage information term mappings.
#' @param verbose Whether to show status messages (TRUE).
#' @seealso md_preprocess()
#' @return Postprocessed metadata table.
#' @export
md_postprocess <- function(ts, mdpre, mdpost.fname = "md_postprocess",
                            md.dpath = file.path("recount-methylation-files",
                                               "metadata"),
                            mdpre.vl=list("study_id"="gse","sample_id"="gsm",
                                          "sample_title" = "gsm_title",
                                          "disease" = "disease_state",
                                          "sample_type" = "sample_type",
                                          "sex" = "sex", "info" = "info",
                                          "age" = "age",
                                          "age_temp" = "age_temp"),
                            mdpost.vl = list("tissue" = "tissue",
                                             "disease" = "disease",
                                             "age" = "age", "sex" = "sex",
                                             "storageinfo" = "storageinfo"),
                            disease.search.vars = c("sample_title", "disease"),
                            tissue.search.vars = c("sample_type",
                                                   "sample_title"),
                            storage.info.vars = c("sample_type",
                                                  "sample_title", "info"),
                            verbose = TRUE){
  mdpost.fn <- paste0(mdpost.fname, "_", ts, ".rda")
  mdpost.fpath <- file.path(md.dpath, mdpost.fn)
  if(verbose){message("Will save mdpost data to ", mdpost.fpath, "...")}
  mdpost <- mdpre[,c(mdpre.vl[["sample_id"]], mdpre.vl[["study_id"]],
                     mdpre.vl[["sample_title"]])]
  mdpost[,mdpost.vl[["tissue"]]] <- mdpost[,mdpost.vl[["disease"]]] <- "NA"
  mdpost[,mdpost.vl[["age"]]] <- mdpost[,mdpost.vl[["sex"]]] <- "NA"
  mdpost[,mdpost.vl[["storageinfo"]]] <- "NA"
  if(verbose){message("Getting disease status...")}; dxl <- md_post_disease()
  which.var <- unlist(mdpre.vl[names(mdpre.vl) %in% disease.search.vars])
  for(dx in names(dxl)){
    ssv <- dxl[[dx]]; pstr <- get_pstr(v = ssv)
    gfilt <- get_filt(v = pstr, m = mdpre, ntfilt = ssv, varl = which.var)
    dx.var <- appendvar(var = mdpost.vl[["disease"]], val = dx,
                        filtv = gfilt, m = mdpost)
    mdpost[,mdpost.vl[["disease"]]] <- dx.var}
  if(verbose){message("Getting disease terms for cancers by type/location...")}
  which.var <- unlist(mdpre.vl[names(mdpre.vl) %in% disease.search.vars])
  cxsubl <- md_post_cancertype()
  for(cxsub in names(cxsubl)){
    ssv <- cxsubl[[cxsub]]; pstr <- get_pstr(v = ssv)
    gfilt <- get_filt(v = pstr, m = mdpre, varl = which.var)
    dxvar1 <- appendvar(var = mdpost.vl[["disease"]], val = cxsub,
                        filtv = gfilt, m = mdpost)
    mdpost[,mdpost.vl[["disease"]]] <- dxvar1
    dxvar2 <- appendvar(var = mdpost.vl[["disease"]], val = "cancer",
                        filtv = gfilt, m = mdpost)
    mdpost[,mdpost.vl[["disease"]]] <- dxvar2}
  if(verbose){message("Getting leukemia disease terms...")};leukl <- md_post_
    leukemia()
  which.var <- unlist(mdpre.vl[names(mdpre.vl) %in% disease.search.vars])
  for(leuk in names(leukl)){
    pstr <- suppressMessages(get_pstr(v = leukl[[leuk]]))
    gfilt <- suppressMessages(get_filt(v = pstr, m = mdpre, ntfilt = pstr,
                                      varl = which.var))
    dxvar1 <- appendvar(var = mdpost.vl[["disease"]], val = leuk,
                        filtv = gfilt, m = mdpost)
    mdpost[,mdpost.vl[["disease"]]] <- dxvar1
    dxvar2 <- appendvar(var = mdpost.vl[["disease"]], val = "cancer",
                        filtv = gfilt, m = mdpost)
    mdpost[,mdpost.vl[["disease"]]] <- dxvar2}
  if(verbose){message("Getting tissue and disease for cancer subtypes...")}
  which.var <- c(mdpre.vl[["sample_title"]], mdpre.vl[["sample_type"]],
                 mdpre.vl[["disease"]]);cxl <- md_post_cancer()
  for(cx in names(cxl)){
    ssv <- cxl[[cx]]; pstr <- get_pstr(v = ssv)
    gfilt<-suppressMessages(get_filt(v=pstr,m=mdpre,ntfilt=ssv,varl=which.var))
    txvar <- appendvar(var = mdpost.vl[["tissue"]], val = cx,
                       filtv = gfilt, m = mdpost)
    mdpost[,mdpost.vl[["tissue"]]] <- txvar
    dxvar <- appendvar(var = mdpost.vl[["disease"]], val = "cancer",
                       filtv = gfilt, m = mdpost)
    mdpost[,mdpost.vl[["disease"]]] <- dxvar}
  if(verbose){message("Getting tissue terms for cancers by type/location...")}
  which.var <- unlist(mdpre.vl[names(mdpre.vl) %in% tissue.search.vars])
  for(cxsub in names(cxsubl)){
    ssv <- cxsubl[[cxsub]]; pstr <- get_pstr(v = ssv)
    gfilt <- get_filt(v = pstr, m = mdpre, varl = which.var);
    txvar1 <- appendvar(var = mdpost.vl[["tissue"]], val = cxsub,
                        filtv = gfilt, m = mdpost)
    mdpost[,mdpost.vl[["tissue"]]] <- txvar1
    txvar2 <- appendvar(var = mdpost.vl[["tissue"]], val = "cancer",
                        filtv = gfilt, m = mdpost)
    mdpost[,mdpost.vl[["tissue"]]] <- txvar2}
  if(verbose){message("Getting leukemia tissue terms...")}
  which.var <- unlist(mdpre.vl[names(mdpre.vl) %in% tissue.search.vars])
  for(leuk in names(leukl)){
    pstr <- get_pstr(v = leukl[[leuk]])
    gfilt <- get_filt(v = pstr, m = mdpre, varl = which.var)
    txvar <- appendvar(var = mdpost.vl[["tissue"]], val = leuk,
                       filtv = gfilt, m = mdpost)
    mdpost[,mdpost.vl[["tissue"]]] <- txvar}
  if(verbose){message("Getting tissue annotations...")}
  which.var <- unlist(mdpre.vl[names(mdpre.vl) %in% tissue.search.vars])
  txl <- list();txl[["blood"]] <- md_post_tissue_blood()
  txl[["other"]] <- md_post_tissue()
  for(sublist in txl){
    for(tx in names(sublist)){
      pstr <- get_pstr(v = sublist[[tx]])
      gfilt <- get_filt(v = pstr, m = mdpre, varl = which.var)
      txvar <- appendvar(var = mdpost.vl[["tissue"]], val = tx,
                         filtv = gfilt, m = mdpost)
      mdpost[,mdpost.vl[["tissue"]]] <- txvar}}
  if(verbose){message("Getting storage info...")}
  which.var <- unlist(mdpre.vl[names(mdpre.vl) %in% storage.info.vars])
  lstorage <- md_post_storage()
  for(sn in names(lstorage)){
    pstr <- get_pstr(v = lstorage[[sn]])
    gfilt <- get_filt(v = pstr, m = mdpre, varl = which.var)
    sinfovar <- appendvar(var = mdpost.vl[["storageinfo"]], val = sn,
                          filtv = gfilt, m = mdpost)
    mdpost[,mdpost.vl[["storageinfo"]]] <- sinfovar}
  if(verbose){message("Getting age info...")}
  mdpost<-suppressMessages(md_post_handle_age(mdpre=mdpre,mdpost=mdpost,
                                              mdpre.vl=mdpre.vl,
                                              mdpost.vl = mdpost.vl,
                                              verbose = verbose))
  if(verbose){message("Getting sex info...")}
  mdpre.sex.cname <- mdpre.vl[["sex"]];mdpost.sex.cname <- mdpost.vl[["sex"]]
  femv <- get_pstr(v = c("female", "f", "FEMALE"))
  malev <- get_pstr(v = c("male", "MALE", "m"))
  mdpost[,mdpost.sex.cname] <- ifelse(grepl(femv, mdpre[,mdpre.sex.cname]),"F",
                     ifelse(grepl(malev, mdpre[,mdpre.sex.cname]),"M", "NA"))
  if(verbose){message("Saving mdpost to ", mdpost.fpath)}
  save(mdpost, file = mdpost.fpath);return(NULL)
```

```
}
```

A.5 Example conda shell script

An example `conda` [183] shell script, `conda.sh`. This script creates virtual environments to run one of five RI detection tools for short-read RNA-seq data (Section 4). New environments are created using `conda create ...`. Dependencies with specific versions are downloaded using `conda install ...`. Note that many download calls also specify a repository such as `bioconda`. Certain calls also specify a group of commonly downloaded dependencies such as `r-base`.

```
#!/usr/bin/bash

# Author: Sean Maden
#
# Programmatic setup of conda environments for retained intron detection
# tools. Setup is expedited by installing shared dependencies once and
# cloning the consensus environment for specific tools. At any time, use
# `conda info --envs` to show all available environments.
#
# Run setup programmatically, either by running this script, or by
# running `conda env create -f <env_yml_path>`, replacing <env_yml_path>
# with the path to one of the env.*.yml files at ./inst/yml/.
#
# On Windows, you may need to use `source activate <env_name>` and
# `source deactivate <env_name>` rather than `conda activate <env_name>`
# and `conda deactivate <env_name>`.
#

conda update conda
conda install python=3.7.0
conda install python=2.7.18
conda install r=3.5.1
conda install -c bioconda bioconductor-biocinstaller

#-------------------
# manage python envs
#-------------------

# python3 envs
conda create -n py370 python=3.7.0
# clone for preprocessing
conda create --name py370_preprocess --clone py370

# python 2 envs
conda create -n py2718 python=2.7.18
# clone for iread
conda create --name py2718_iread --clone py2718

#--------------
# manage r envs
#--------------

# make r v4### env
conda create -n r403; conda activate r403
conda install -c conda-forge r-base
conda install -c conda-forge/label/gcc7 r-base
conda deactivate
# clone r v4### env for interest and superintronic
conda create --name r403_interest --clone r403
conda create --name r403_superintronic --clone r403
# 2 tools can share the same env
conda create --name r403_interest_superintronic --clone r403

# make r v3### env
```

```
conda create -n r351 r=3.5.1; conda attach r351
conda install -c conda-forge/label/gcc7 r-devtools=2.0.1
conda install -c conda-forge/label/gcc7 r-data.table=1.14.0
# clone r v3### env for sirfinder, kma
conda create --name r351_sirfinder --clone r351
conda create --name r351_kma --clone r351

#---------------------
# preprocess env setup
#---------------------
conda activate py370_preprocess
# dependencies
conda install -c bioconda samtools=1.3.1
conda install -c bioconda bowtie2=2.3.4.3
conda install -c bioconda star=2.7.6a

conda env export > env_preprocess.yml

conda deactivate

#-------------------
# interest env setup
#-------------------
conda activate r403_interest
# install dependencies
conda install -c bioconda r-seqinr=4.2_5 # for IntEREst
conda install -c bioconda bioconductor-edgeR=3.32.1 # for IntEREst
conda install -c bioconda bioconductor-DEXSeq=1.36.0 # for IntEREst
# install interest
conda install -c bioconda bioconductor-interest

conda env export > env_interest.yml # export yml file

conda deactivate

#------------------------
# superintronic env setup
#------------------------
conda activate r403_superintronic
# dependencies
conda install -c bioconda bioconductor-plyranges=1.10.0
conda install -c bioconda bioconductor-genomicfeatures=1.42.2
conda install -c conda-forge r-devtools=2.4.0
conda install -c conda-forge r-patchwork=1.1.1
conda install -c conda-forge r-ggplot2=3.3.3

conda install -c anaconda gxx_linux-64

# install superintronic
R
devtools::install_github("sa-lee/superintronic", build_vignette = FALSE)

conda env export > env_superintronic.yml # export yml file

conda deactivate

#----------------
# iread env setup
#----------------
conda activate py2718_iread
conda install perl=5.26.2
git clone https://github.com/genemine/iread/
# iread dependencies
conda install -c bioconda samtools=1.2
conda install -c bioconda bedops=2.4.20
conda install -c conda-forge argparse=1.4.0
conda install -c bioconda perl-parallel-forkmanager=2.02

conda env export > env_iread.yml # export yml file

conda deactivate

#--------------------
# sirfinder env setup
#--------------------
conda activate r351_sirfinder
# dependencies
# conda install -c conda-forge/label/gcc7 r-devtools # to install from github
conda install -c conda-forge/label/gcc7 r-rfast=1.9.5
conda install -c conda-forge/label/gcc7 r-rlang=0.4.10

R
devtools::install_github("lbroseus/SIRFindeR", build_vignette = TRUE)

conda env export > env_sirfinder.yml # export yml file

conda deactivate

#--------------
# kma env setup
#--------------
conda activate r351_kma
# dependencies
conda install -c conda-forge/label/gcc7 r-reshape2=1.4.3
conda install -c conda-forge/label/gcc7 r-dplyr=0.7.8

# get fastq-dump to obtain sample fastqs
conda install -c bioconda/label/cf201901 sra-tools

# preprocessing dependencies
python -m pip install pyfasta
python -m pip install pysam
python -m pip install biopython==1.76
conda install -c bioconda bowtie2=2.3.4.3

# quantification dependencies
conda install -c bioconda express

# get kma
R
devtools::install_github("https://github.com/adamtongji/kma")

conda env export > env_kma.yml # export yml file

conda deactivate
```

A.6 Example `conda` YML script

An example YML script, `env_kma.yml`, is shown. This was automatically generated for the virtual environment specified in the above script `conda.sh` (Section A.5). It exactly reproduces a virtual environment supporting the KMA RI-detection tool for short-read RNA-seq data, which was used to test the reliability of called introns (Section 4).

```
name: r351_kma
channels:
  - conda-forge/label/gcc7
  - conda-forge
  - bioconda
  - defaults
```

```
dependencies:
  - _r-mutex=1.0.1=anacondar_1
  - bwidget=1.9.14=h694c41f_0
  - bzip2=1.0.8=h0d85af4_4
  - ca-certificates=2018.10.15=ha4d7672_0
  - cairo=1.16.0=hec6a9b0_1003
  - cctools=949.0.1=h22b1bf0_0
  - cctools_osx-64=949.0.1=h5ba7a2e_0
  - certifi=2020.12.5=py39h6e9494a_1
  - clang=9.0.1=default_ha9b4ba2_1
  - clang-9=9.0.1=default_heda16ac_1
  - clang_osx-64=9.0.1=h05bbb7f_0
  - clangxx=9.0.1=default_he082bbe_1
  - clangxx_osx-64=9.0.1=h05bbb7f_2
  - compiler-rt=9.0.1=h6a512c6_3
  - compiler-rt_osx-64=9.0.1=h99342c6_3
  - curl=7.68.0=h8754def_0
  - fontconfig=2.13.1=h10f422b_1005
  - freetype=2.10.4=h4cff582_1
  - fribidi=1.0.10=hbcb3906_0
  - gettext=0.19.8.1=h7937167_1005
  - gfortran_impl_osx-64=7.5.0=hae4d780_7
  - gfortran_osx-64=7.5.0=h044cb63_8
  - glib=2.68.1=he49afe7_0
  - glib-tools=2.68.1=he49afe7_0
  - gmp=6.2.1=h2e338ed_0
  - graphite2=1.3.13=h2e338ed_1001
  - gsl=2.5=ha2d443c_1
  - harfbuzz=2.4.0=hd8d2a14_3
  - icu=64.2=h6de7cb9_1
  - isl=0.22.1=hb1e8313_2
  - jpeg=9d=hbcb3906_0
  - krb5=1.16.4=h1752a42_0
  - ld64=530=0
  - ld64_osx-64=530=h3c32e8a_0
  - libblas=3.9.0=8_openblas
  - libcblas=3.9.0=8_openblas
  - libclang-cpp9=9.0.1=default_heda16ac_1
  - libcurl=7.68.0=h709d2b2_0
  - libcxx=11.1.0=habf9029_0
  - libedit=3.1.20191231=h0678c8f_2
  - libffi=3.3=h046ec9c_2
  - libgfortran=4.0.0=7_5_0_h1a10cd1_22
  - libgfortran4=7.5.0=h1a10cd1_22
  - libglib=2.68.1=hd556434_0
  - libiconv=1.16=haf1e3a3_0
  - libllvm9=9.0.1=h223d4b2_3
  - libopenblas=0.3.12=openmp_h63d9170_1
  - libpng=1.6.37=h7cec526_2
  - libssh2=1.9.0=h52ee1ee_6
  - libtiff=4.2.0=h7c11950_1
  - libwebp-base=1.2.0=h0d85af4_2
  - libxml2=2.9.10=h53d96d6_0
  - llvm-openmp=11.1.0=hda6cdc1_1
  - lz4-c=1.9.3=h046ec9c_0
  - make=4.3=h22f3db7_1
  - mpc=1.1.0=ha57cd0f_1009
  - mpfr=4.0.2=h72d8aaf_1
  - ncurses=6.2=h2e338ed_4
  - openssl=1.1.1k=h0d85af4_0
  - pango=1.42.4=haa940fe_4
  - pcre=8.44=hb1e8313_0
  - pip=21.1.1=pyhd8ed1ab_0
  - pixman=0.38.0=h01d97ff_1003
  - python=3.9.2=h2502468_0_cpython
  - python_abi=3.9=1_cp39
  - r=3.5.1=r35_1003
  - r-askpass=1.1=r35h17f1fa6_1
  - r-assertthat=0.2.0=r351h6115d3f_1001
  - r-backports=1.1.3=r351h46e59ec_1000
  - r-base=3.5.1=hc03ab29_1012
  - r-bh=1.66.0_1=r351_2001
  - r-bindr=0.1.1=r351h6115d3f_1001
  - r-bindrcpp=0.2.2=r351h466af19_1001
  - r-boot=1.3_25=r35h6115d3f_0
  - r-callr=3.1.1=r351h6115d3f_1000
  - r-class=7.3_17=r35h17f1fa6_0
  - r-cli=1.0.1=r351h6115d3f_1000
  - r-clipr=0.4.1=r351h6115d3f_1001
  - r-clisymbols=1.2.0=r351h6115d3f_1001
  - r-cluster=2.1.0=r35h384270c_2
  - r-codetools=0.2_16=r35h6115d3f_1001
  - r-crayon=1.3.4=r351h6115d3f_1001
  - r-curl=3.2=r351h46e59ec_1002
  - r-desc=1.2.0=r351h6115d3f_1001
  - r-devtools=2.0.1=r351h6115d3f_1000
  - r-digest=0.6.18=r351h46e59ec_1000
  - r-dplyr=0.7.8=r351h466af19_1000
  - r-fansi=0.4.0=r351h46e59ec_1000
  - r-foreign=0.8_76=r35h17f1fa6_0
  - r-fs=1.2.6=r351h466af19_1000
  - r-gh=1.0.1=r351h6115d3f_1001
  - r-git2r=0.26.1=r35h8d49edc_1
  - r-glue=1.3.0=r351h1de35cc_1002
  - r-httr=1.4.0=r351h6115d3f_1000
  - r-ini=0.3.1=r351h6115d3f_1001
  - r-jsonlite=1.6=r351h46e59ec_1000
  - r-kernsmooth=2.23_17=r35h3830744_0
  - r-lattice=0.20_41=r35h17f1fa6_1
  - r-magrittr=1.5=r351h6115d3f_1001
  - r-mass=7.3_51.6=r35h17f1fa6_1
  - r-matrix=1.2_18=r35h17f1fa6_1
  - r-memoise=1.1.0=r351h6115d3f_1001
  - r-mgcv=1.8_31=r35h17f1fa6_0
  - r-mime=0.6=r351h46e59ec_1000
  - r-nlme=3.1_147=r35h384270c_0
  - r-nnet=7.3_14=r35h17f1fa6_0
  - r-openssl=1.4.1=r35h9d0ceee_0
  - r-pillar=1.3.1=r351h6115d3f_1000
  - r-pkgbuild=1.0.3=r35h6115d3f_1
  - r-pkgconfig=2.0.2=r351h6115d3f_1001
  - r-pkgload=1.0.2=r351h466af19_1000
  - r-plogr=0.2.0=r351h6115d3f_1001
  - r-plyr=1.8.4=r351h466af19_1002
  - r-prettyunits=1.0.2=r351h6115d3f_1001
  - r-processx=3.2.1=r351h46e59ec_1000
  - r-ps=1.3.0=r351h46e59ec_1000
  - r-purrr=0.2.5=r351h46e59ec_1002
  - r-r6=2.3.0=r351h6115d3f_1000
  - r-rcmdcheck=1.3.2=r351h6115d3f_1000
  - r-rcpp=1.0.0=r351h466af19_1000
  - r-recommended=3.5.1=r35_1003
  - r-remotes=2.0.2=r351h6115d3f_1000
  - r-reshape2=1.4.3=r351h466af19_1002
  - r-rlang=0.3.0.1=r351h1de35cc_1000
  - r-rpart=4.1_15=r35h17f1fa6_1
  - r-rprojroot=1.3_2=r351h6115d3f_1001
  - r-rstudioapi=0.8=r351h6115d3f_1001
  - r-sessioninfo=1.1.1=r351h6115d3f_1000
  - r-spatial=7.3_12=r35h17f1fa6_0
  - r-stringi=1.2.4=r351h466af19_1001
  - r-stringr=1.3.1=r351h6115d3f_1001
  - r-survival=3.1_12=r35h17f1fa6_0
  - r-sys=2.1=r351h46e59ec_1000
  - r-tibble=2.0.0=r351h46e59ec_1000
  - r-tidyselect=0.2.5=r351h466af19_1000
  - r-usethis=1.4.0=r351h6115d3f_1000
  - r-utf8=1.1.4=r351h46e59ec_1000
  - r-whisker=0.3_2=r351h6115d3f_1001
  - r-withr=2.1.2=r351h6115d3f_1000
  - r-xopen=1.0.0=r351h6115d3f_1000
  - readline=8.1=h05e3726_0
  - setuptools=49.6.0=py39h6e9494a_3
  - sqlite=3.35.5=h44b9ce1_0
  - tapi=1000.10.8=h879752b_4
  - tk=8.6.10=h0419947_1
  - tktable=2.10=h49f0cf7_3
  - tzdata=2021a=he74cb21_0
  - wheel=0.36.2=pyhd3deb0d_0
  - xz=5.2.5=haf1e3a3_1
  - zlib=1.2.11=h7795811_1010
  - zstd=1.4.9=h582d3a0_0
prefix: /Users/maden/miniconda3/envs/r351_kma
```

A.7 Example `Snakefile` script used by `recountmethylation_instance`

This `Snakefile` was used to define rules for `recountmethylation_instance` [61]. These rules reference functions defined in two respective dependencies, `recountmethylation_server` [193] and `recountmethylation.pipeline` [194] resources. Running this script automatically produces status logs that track progress and store console messages, warnings, and errors.

```
#!/usr/bin/env snakemake

# Author: Sean Maden
#
# Description:
# This script describes the workflow processes to create and manage an
# instance of recountmethylation with logging. This includes management of the
# target platform accession ID, the ‘recount-methylation-files‘ directory tree,
# acquision/download of files from GEO, reporting on the status of various file
# types, compilation of DNAm data into databases, and mapping sample metadata.
#
# Setup:
# There are 2 principal ways to start and manage a recountmethylation instance:
#
# * Containerization : Use the docker image and docker compose to rapidly set up
#   an instance, with automatic download and install of major dependencies
#
# * Local use : Manage a local instance without containerization. This entails
#   that you follow the setup instructions and manage dependency access for your
#   particular system.
#
# Quick start:
# You may optionally set the platform accession ID as follows. Otherwise, the
# instance defaults to targeting all available samples for the HM450K platform:
#
# > snakemake --cores 1 set_acc
#
# From a nix console, you may start the instance by running ‘server.py‘ using:
#
# > snakemake --cores 1 run_server
#
```

```
# You may now monitor the progress of the server from a new terminal. To aid
# with instance monitoring, run the script `serverdash.py` to initialize a
# dashboard:
#
# > snakemake --cores 1 server_dash
#
# This will update automatically with current information about files in the
# instance.
#

#------------------------------
# Set dependencies, paths, etc.
#------------------------------
import os, sys, random, string
# Set scripts path
server_repo_name = "recountmethylation_server"
server_repo_path = os.path.join(server_repo_name)
if os.path.isdir(server_repo_path):
    print("Found server repo path.")
    srcpath = os.path.join(server_repo_path, "src")
    if os.path.isdir(srcpath):
        print("Found server src path.")
    else:
        print("Error, couldn't find server src dir at path "+srcpath)
else:
    print("Error, couldn't find server repo at path "+server_repo_path)
# add server src to path
sys.path.insert(0, srcpath); from utilities import gettime_ntp
# pipeline repo
rmp_path = os.path.join("recountmethylation.pipeline", "inst", "snakemake")
# research synth resources repo
# rsynth_path = os.path.join("recount.synth", "inst", "scripts", "snakemake")
# logs info
logsfn = "snakemakelogs"; logspath = os.path.join(logsfn)

#---------------
# Manage logging
#---------------
if not os.path.isdir(logsfn):
    os.mkdir(logsfn)
if len(os.listdir(logspath)) > 0:
    print("Found "+str(len(os.listdir(logspath)))+" files in logs dir "+
        logspath)
    accept = str(input("(Y/N) Do you want to clear existing logs from the "+
        "logs dir?\n"))
    if accept in ["y", "Y", "yes", "Yes", "YES"]:
        print("Removing old log files...")
        for file in os.listdir(logspath):
            os.remove(os.path.join(logspath, file))
    elif accept in ["n", "N", "no", "No", "NO"]:
        print("Skipping logs dir cleanup...")
    else:
        print("Error, invalid input. Skipping logs dir cleanup...")

#---------------
# Get timestamps
#---------------
ts = gettime_ntp(); print("New timestamp for run: "+ts)

#-------------------
# Workflow processes
#-------------------
# Server processes
# NOTE: Rules to handle file acquisition and formatting from GEO.

# Set the target platform accession for the instance
rule set_acc:
    input: os.path.join(srcpath, "set_acc.py")
    log: os.path.join(logspath, "set_acc_"+ts+".log")
    shell: "python3 {input} > {log}"

# Query the GEO DataSets API for samples, studies for the indicated platform
rule new_eqd:
    input: os.path.join(srcpath, "edirect_query.py")
    log: os.path.join(logspath, "eqd_"+ts+".log")
    shell: "python3 {input} > {log}"

# Exclude samples included in the most recent freeze located at
# ./inst/freeze_gsmv
rule exclude_gsm:
    input: os.path.join(srcpath, "gsm_exclude.py")
    log: os.path.join(logspath, "gsm_exclude_"+ts+".log")
    shell: "python3 {input} > {log}"

# Run the server process to download study SOFT files and sample IDAT files
rule run_server:
    input: os.path.join(srcpath, "server.py")
    log: os.path.join(logspath, "run_server_"+ts+".log")
    shell: "python3 {input} > {log}"

# Run the server dashboard utility. View by opening a browser window at the
# indicated IP address
rule server_dash:
    input: os.path.join(srcpath, "serverdash.py")
    log: os.path.join(logspath, "serverdash_"+ts+".log")
    shell: "python3 {input} > {log}"

# Unzip .gz compressed IDAT files
rule unzip_idats:
    input: os.path.join(srcpath, "process_idats.py")
    log: os.path.join(logspath, "unzip_idats_"+ts+".log")
    shell: "python3 {input} > {log}"

# Make new IDAT hardlinks with identical basenames (e.g. same timestamps)
rule make_idat_hlinks:
    input: os.path.join(srcpath, "rsheet.py")
    log: os.path.join(logspath, "rsheet_"+ts+".log")
    shell: "python3 {input} > {log}"

# Expand study SOFT files, extract sample metadata, and store as
# sample-specific files
rule process_soft:
    input: os.path.join(srcpath, "process_soft.py")
    log: os.path.join(logspath, "process_soft_"+ts+".log")
    shell: "python3 {input} > {log}"

# Convert SOFT-derived sample metadata to JSON format, then further sample JSON
# data to remove any study-specific metadata fields
rule apply_jsonfilt:
    input: os.path.join(srcpath, "jsonfilt.R")
    log: os.path.join(logspath, "apply_jsonfilt_"+ts+".log")
    shell: "Rscript {input} >& {log}"

#rule soft_cleanup:
#    input: os.path.join(srcpath, "process_soft.py")
#    log: os.path.join(logspath, "soft_cleanup_"+ts+".log")
#    shell: "python3 {input} --cleanup True  > {log}"

# Generate a report summarizing the current instance files
rule report:
    input: os.path.join(srcpath, "report.py")
    log: os.path.join(logspath, "report_"+ts+".log")
    shell: "python3 {input} > {log}"

#---------------------------
# DNAm database compilations
#---------------------------
# NOTE: Rules to form the initial compilation files

# Gets input for instance version, timestamp, etc.
rule new_instance_md:
    input: os.path.join(rmp_path, "new_instance_md.R")
    log: os.path.join(logspath, "new_instance_md_"+ts+".log")
    shell: "Rscript {input} >& {log}"

# Run the full pipeline (makes h5 and h5se files in 3 formats)
rule run_dnam_pipeline:
    input: os.path.join(rmp_path, "run_dnam_pipeline.R")
    log: os.path.join(logspath, "run_dnam_pipeline_"+ts+".log")
    shell: "Rscript {input} >& {log}"

# Compile the red/green signal data table files
rule get_rg_compilations:
    input: os.path.join(rmp_path, "get_rg_compilations.R")
    log: os.path.join(logspath, "get_rg_compilations_"+ts+".log")
    shell: "Rscript {input} >& {log}"

# Make the h5 rg database
rule get_h5db_rg:
    input: os.path.join(rmp_path, "get_h5db_rg.R")
    log: os.path.join(logspath, "get_h5db_rg_"+ts+".log")
    shell: "Rscript {input} >& {log}"

# Make the h5se rg database
rule get_h5se_rg:
    input: os.path.join(rmp_path, "get_h5se_rg.R")
    log: os.path.join(logspath, "get_h5se_rg_"+ts+".log")
    shell: "Rscript {input} >& {log}"

# Make the h5 gm database
rule get_h5db_gm:
    input: os.path.join(rmp_path, "get_h5db_gm.R")
    log: os.path.join(logspath, "get_h5db_gm_"+ts+".log")
    shell: "Rscript {input} >& {log}"

# Make the h5 gm database
rule get_h5se_gm:
    input: os.path.join(rmp_path, "get_h5se_gm.R")
    log: os.path.join(logspath, "get_h5se_gm_"+ts+".log")
    shell: "Rscript {input} >& {log}"

# Make the h5 gr database
rule get_h5db_gr:
    input: os.path.join(rmp_path, "get_h5db_gr.R")
    log: os.path.join(logspath, "get_h5db_gr_"+ts+".log")
    shell: "Rscript {input} >& {log}"

# Make the h5se gr database
rule get_h5se_gr:
    input: os.path.join(rmp_path, "get_h5se_gr.R")
    log: os.path.join(logspath, "get_h5se_gr_"+ts+".log")
    shell: "Rscript {input} >& {log}"
```

```

#-------------------------
# Process sample metadata
#-------------------------
# NOTE: Rules to extract and map sample metadata

# Run the MetaSRA-pipeline
rule run_msrap:
    input: os.path.join(srcpath, "run_msrap.R")
    log: os.path.join(logspath, "run_msrap_"+ts+".log")
    shell: "Rscript {input} >& {log}"

# Map and harmonize metadata from filtered JSON files
rule do_mdmap:
    input: os.path.join(rmp_path, "do_mdmap.R")
    log: os.path.join(logspath, "do_mdmap_"+ts+".log")
    shell: "Rscript {input} >& {log}"

# Get DNAm-derived md and qc metrics
rule do_dnam_md:
    input: os.path.join(rmp_path, "do_dnam_md.R")
    log: os.path.join(logspath, "do_dnam_md_"+ts+".log")
    shell: "Rscript {input} >& {log}"

# Get composite md from mdfinal, mdpred, mdqc
rule make_md_final:
    input: os.path.join(rmp_path, "make_md_final.R")
    log: os.path.join(logspath, "make_md_final_"+ts+".log")
    shell: "Rscript {input} >& {log}"

# Append updated md to available compilation files
rule append_md:
    input: os.path.join(rmp_path, "append_md.R")
    log: os.path.join(logspath, "append_md_"+ts+".log")
    shell: "Rscript {input} >& {log}"
```

A.8 Example interoperability script

The following code from the script `dnam_search_index.R` shows how a `conda` virtual environment is set up and a Python script, `dnam_search_index.py`, is sourced and used from an R environment with the `reticulate` and `basilisk` R packages.

```
#!/usr/bin/env R

# Author: Sean Maden
#
# Functions to manage search index construction from DNAm array data, including
# dimension reduction with feature hashing and support for k nearest neighbors
# search lookup. The HNSW implementation in the hnswlib Python library is used
# for search index construction.
#
# Due to the varying availability of the Python package depedencies across
# operating systems, to ensure package build success these functions aren't
# exported and must be called with "recountmethylation:::". For background and
# instructions about workign with search indices for DNAm arrays, see the package
# vignette "Nearest neighbors analysis for DNAm arrays."
#

#' setup_sienv
#'
#' Set up a new virtual environment for search index construction using
#' the basilisk package.
#'
#' @param env.name Name of the new virtual environment (required,
```

```r
#' "dnam_si-hnswlib")
#' @param pkgv Vector of the dependencies and their versions for the new
#' virtual environment (required, format: "packagename==versionnum").
#' @returns New basilisk environment object.
setup_sienv <- function(env.name = "dnam_si_hnswlib",
                        pkgv = c("python==3.7.1", "hnswlib==0.5.1", "pandas==1.2.2
   ",
                          "numpy==1.20.1", "mmh3==3.0.0", "h5py==3.2.1")){
  message("Defining the virtual env dependencies...")
  my_env <- basilisk::BasiliskEnvironment(envname = env.name,
                                          pkgname = "recountmethylation",
                                          packages = pkgv)
  message("Running virtual environment setup...")
  proc <- basilisk::basiliskStart(my_env) # define env process
  on.exit(basilisk::basiliskStop(proc)) # set exit process
  return(proc)
}

#' get_fh
#'
#' Get the hashed features for a data table. Uses reticulate package to
#' call the Python script to do feature hashing on a table of data. It is
#' assumed the input table has sample data in rows, with probe data in
#' columns. The input data table should have row names but not column names.
#'
#' @param csv_savepath Name/path of hashed features table to write (required,
#' string, writes new csv where rows = samples, cols = hashed features).
#' @param csv_openpath Name/path of table to hash (required, string, assumes
#' a csv where rows = samples, cols = probes).
#' @param ndim Number of hashed features (integer, 1000).
#' @param lstart Line index to start on (0-based for Python, required, int, 0).
#' @returns Path to new hashed featuers table.
#' @examples
#' # get example bval csv
#' # of_fpath <- system.file("extdata", "fhtest",
#' #        package = "recountmethylation")
#' # of_fpath <- file.path(of_fpath, "tbval_test.csv")
#' # write new hashed features results
#' # get_fh(csv_savepath = "bval_fn.csv", csv_openpath = of_fpath, ndim = 100)
get_fh <- function(csv_savepath, csv_openpath, ndim = 1000, lstart = 1){
  message("Starting basilisk process...")
  proc <- recountmethylation:::setup_sienv()
  pyscript.path <- system.file("python", package = "recountmethylation")
  pyscript.path <- file.path(pyscript.path, "dnam_search_index.py")
  message("Doing feature hashing...")
  ncol <- length(unlist(strsplit(readLines(csv_openpath, 1), ",")))
  message("Target features will reduce dimensions from ", ncol,
          " to ", ndim, "...")
  basilisk::basiliskRun(proc, function(pyscript.path, csv_savepath,
                                       csv_openpath, ndim, lstart){
    message("Sourcing Python functions...")
    reticulate::source_python(pyscript.path)
    make_fhmatrix_autolabel(wf_name = csv_savepath, of_name = csv_openpath,
                            ndim = as.integer(ndim), lstart = lstart)
  }, pyscript.path = pyscript.path, csv_savepath = csv_savepath,
  csv_openpath = csv_openpath, ndim = ndim, lstart = lstart)
  if(file.exists(csv_savepath)){return(csv_savepath)} else{return(FALSE)}
  return(NULL)
}

#' make_si
#'
#' Make search index from table of hashed features. Additional details about
#' the hnswlib search index parameters (e..g `space_val`, `efc_val`, `m_val`,
#' and `ef_val`) can be found in the Python package docstrings and ReadMe.
#'
#' @param fh_csv_fpath Name/path of csv (e.g. a table of hashed features)
#' containing data for the index (required, string, "bvaltest.csv", where
#' rows = samples, cols = features).
#' @param si_fname Name of new search index file to save (required, string,
#' "new_search_index.pickle")
#' @param si_dict_fname Name of new index dictionary, with string labels, to
#' save (required, string, "new_index_dict.pickle").
#' @param threads Number of threads for processing new index (required, int, 4).
#' @param space_val Space value for new search index (required, valid string,
#' l2').
#' @param efc_val EFC value for the index (required, int, 2000).
#' @param m_val M value for the index (required, int, 1000).
#' @param ef_val EF value for the index (required, int, 2000).
#' @returns Boolean, TRUE if new search index and dictionary created, FALSE if
#' creating the new search index and dictionary files failed, otherwise NULL.
#' @examples
#' # fh_csv_fpath <- system.file("extdata", "fhtest",
#' # package = "recountmethylation")
#' # fh_csv_fpath <- file.path(fh_csv_fpath, "bval_fn.csv")
#' # make_si(fh_csv_fpath)
make_si <- function(fh_csv_fpath, si_fname = "new_search_index.pickle",
                    si_dict_fname = "new_index_dict.pickle", threads = 4,
                    space_val = 'l2', efc_val = 2000, m_val = 1000, ef_val = 2000)
  {
  message("Starting basilisk process...")
  proc <- recountmethylation:::setup_sienv()
  pyscript.path <- system.file("python", package = "recountmethylation")
  pyscript.path <- file.path(pyscript.path, "dnam_search_index.py")
  basilisk::basiliskRun(proc, function(pyscript.path, fh_csv_fpath, si_fname,
                                       si_dict_fname, threads, space_val,
                                       efc_val, m_val, ef_val){
    message("Sourcing Python functions...")
    reticulate::source_python(pyscript.path)
    message("Making search index...")
    make_hnsw_si(fname = fh_csv_fpath, index_name = si_fname,
            dindex_name = si_dict_fname, threads = as.integer(threads),
            efc_val = as.integer(efc_val), m_val = as.integer(m_val),
            ef_val = as.integer(ef_val))
  },
  pyscript.path = pyscript.path, fh_csv_fpath = fh_csv_fpath,
  si_fname = si_fname, si_dict_fname = si_dict_fname, threads = threads,
  space_val = space_val, efc_val = efc_val, m_val = m_val, ef_val = ef_val
  )
  if(file.exists(si_fname) & file.exists(si_dict_fname)){
    message("Made new search index '",si_fname,
            "' and search index dict '", si_dict_fname,"'")
    return(TRUE)
  } else{
    return(FALSE)
  }
  return(NULL)
}

#' query_si
#'
#' Query an HNSW search index. Does K Nearest Neighbors lookup on a previously
#' saved search index object, returning the K nearest neighbors of the queried
#' sample(s). The `query_si()` function returns verbose output, which can be
#' silenced with suppressMessages()`.
#'
#' @param sample_idv Vector of valid sample IDs, or GSM IDs, which are included
#' in the rownames of the hashed features table at fh_csv_fpath (requried,
#' vector of char strings).
#' @param fh_csv_fpath Path to the hashed features table, which includes rownames
#' corresponding to sample ID strings in the sample_idv vector (required, char).
#' @param si_fname Base filename of the search index object, used to find the
#' search index and index dict files, which are expected to be located at
#' si_fapth (required, char).
#' @param si_fpath Path to the directory containing the search index and index
#' dict files (required, char).
#' @param lkval Vector of K nearest neighbors to return per query (optional,
#' int, c(1,2)).
#' @returns
#' @examples
#' # file paths
#' # fh table
#' # fh_csv_fname <- system.file("extdata", "fhtest",
#' # package = "recountmethylation")
#' # fh_csv_fname <- file.path(fh_csv_fname, "bval_fh10.csv")
#' # si dict
#' # index_dict_fname <- system.file("extdata", "sitest",
#' # package = "recountmethylation")
#' # index_dict_fname <- file.path(index_dict_fname, "new_index_dict.pickle")
#'
#' # set sample ids to query
#' # sample_idv <- c("GSM1038308.1548799666.hlink.GSM1038308_5958154021_R01C01",
#' #                 "GSM1038309.1548799666.hlink.GSM1038309_5958154021_R02C01")
#' # set a list of k nearest neighbors to query
#' # lkval <- c(1,2,3)
#'
#' # get query results as a data frame (with verbose results messaging)
#' # dfk <- query_si(sample_idv = sample_idv, lkval = lkval,
#' #                 fh_csv_fname = "bval_fn.csv",
#' #                 index_dict_fname = "new_index_dict.pickle")
#' # returns:
#' # Starting basilisk process...
#' # Defining the virtual env dependencies...
#' # Running virtual environment setup...
#' # Sourcing Python functions...
#' # Querying the search index...
#' # Getting hashed features data for samples...
#' # Getting index data for sample:
#' # GSM1038308.1548799666.hlink.GSM1038308_5958154021_R01C01'
#' # Getting index data for sample:
#' # GSM1038309.1548799666.hlink.GSM1038309_5958154021_R02C01'
#' # Beginning queries of k neighbors from lk...
#' # ii =  0 , ki =  1
#' # Loading search index...
#' # Querying 2 elements in data with k = 1 nearest neighbors...
#' # Query completed, time: 0.0007359981536865234
#' # Applying labels to query results...
#' # Returning data (sample id, k index, and distance)...
#' # ii =  1 , ki =  2
#' # Loading search index...
#' # Querying 2 elements in data with k = 2 nearest neighbors...
#' # Query completed, time: 0.0006208419799804688
#' # Applying labels to query results...
#' # Returning data (sample id, k index, and distance)...
#' # ii =  2 , ki =  3
#' # Provided k '3' > n si samples, skipping...
#' # Returning query results...
query_si <- function(sample_idv, fh_csv_fpath,
                     si_fname = "new_search_index",
                     si_fpath = ".", lkval = c(1,2)){
  message("Checking index and table locations...")
  if(!file.exists(fh_csv_fpath)){
    stop("Error: didn't find fh table at location:\n'",fh_csv_fpath,"'")}
  si_hnsw_fpath <- file.path(si_fpath, paste0(si_fname, ".pickle"))
  if(!file.exists(si_hnsw_fpath)){
    stop("Error: didn't find search index at location:\n'",v,"'")}
  si_dict_fpath <- file.path(si_fpath, paste0(si_fname, "_dict.pickle"))
  if(!file.exists(si_dict_fpath)){
    stop("Error: didn't find index dict at location:\n'",si_dict_fpath,"'")}
  message("Starting basilisk process...")
  proc <- recountmethylation:::setup_sienv()
  lkval <- lapply(lkval, as.integer) # format query k values list
  pyscript.path <- system.file("python", package = "recountmethylation")
  pyscript.path <- file.path(pyscript.path, "dnam_search_index.py")
  query_result <- basilisk::basiliskRun(proc, function(pyscript.path,
    sample_idv, lkval, si_dict_fpath, fh_csv_fpath){
    message("Sourcing Python functions...")
    reticulate::source_python(pyscript.path)
    message("Querying the search index...")
    dfk <- make_dfk_sampleid(sample_idv = sample_idv, lk = lkval,
      index_dict_fname = si_dict_fpath, fh_csv_fname = fh_csv_fpath)
    return(dfk)}, pyscript.path = pyscript.path, sample_idv = sample_idv,
    lkval = lkval, si_dict_fpath = si_dict_fpath, fh_csv_fpath = fh_csv_fpath)
  return(query_result)
}
```

The follow code shows the Python script `dnam_search_index.py` which is called by the above R script.

```
#!/usr/bin/env python3

# Author: Sean Maden
#
# Manage DNAm array search indexes. Options to make an HNSW-based
# search index from hashed features tables of DNAm array data.
#

import mmh3
import numpy as np
import pandas as pd
import hnswlib, sys, os, re, pickle, random
from time import time
import faulthandler
faulthandler.enable()

def feature_hash(arr, target_dim=10000):
    """ Perform feature hashing on an array of data

    Perform feature hashing on the data in arr, into a vector of target_dim
    total hashed features.

    Arguments:
        * arr: An array of values to be hashed.
        * target_dim: The target number of hashed values.
    Returns:
        * low_d_rep, or an array of hashed values of len == target_dim

    """
    low_d_rep = [0.0 for _ in range(target_dim)]
    for i, el in enumerate(arr):
        hashed = mmh3.hash(str(i))
        if hashed > 0.0:
            low_d_rep[hashed % target_dim] += arr[i]
        else:
            low_d_rep[hashed % target_dim] -= arr[i]
    return low_d_rep

def make_fhmatrix_autolabel(wf_name, of_name, lnotfloat = ['','NA','NaN'],
    ndim = 10000, lstart = 0):
    """

    Get the hashed features table from an input data table. This function
    automatically sets row labels as the first column of data in the input data
    table at ‘of_name.‘

    Arguments:
        * wf_name: Name/path of hashed features table to write (required,
            string, rows = samples, cols = hashed features).
        * of_name: Name/path of table to hash (required, tring, rows =  samples,
            cols = probes).
        * lnotfloat: List of expected missing value symbols (required, ['','NA',
            'NaN']). These are replaced by the row-wise meidans of non-missing
            values.
        * ndim: Number of hashed features (integer, 1000).
        * lstart: Line to start reading (required, int, 0).
    Returns:
        * None, saves new hashed features table to ‘wf_name‘.

    """
    with open(of_name, "r") as fr:
        with open(wf_name, "w") as fw:
            for li, line in enumerate(fr):
                line_format = line.replace('\n', '').split(',')
                newrow = line_format[0]; lli = line_format[1::]
                if li >= lstart:
                    # replace NAs with median values
                    lli_median = np.median([float(ii) for ii in lli
                        if not ii in lnotfloat])
                    lli_format = [float(ii) if not ii in lnotfloat
                                    else lli_median for ii in lli]
                    lli_fh = feature_hash(lli_format, target_dim = ndim)
                    newrow = newrow + ',' + ','.join([str(ii) for ii in lli_fh])
                    newrow = newrow + '\n'
                    print('Found new sample: '+newrow[0:100])
                    fw.write(newrow)
                print("Finished with line number "+str(li))
    return None

def make_fhmatrix_specifylabels(labels_list, wf_name, of_name,
    lnotfloat = ['','NA','NaN'], ndim = 10000):
    """ Make the hashed feaures matrix a Beta-values table

    Saves a .csv table of ndim total hashed features (columns) by elements,
    where the number of elements is equal to the row count in the of_name table.
    Missing/NA values are replaced by the row median. New rows are processively
    written to the file wf_name.

    Arguments:
        * labels_list : Column names of the output table
        * wf_name : Name of the output table
        * of_name : Name of table to be hashed. Rows will be hashed. First line
                    is colnames and thus skipped.
    Returns:
        * None, produces a new hashed features .csv table of dim nrow_of_name by
        ndim+1 (columns), where first column has feature labels


    """
    with open(of_name, "r") as fr:
        with open(wf_name, "w") as fw:
            for li, line in enumerate(fr):
                if li > 0:
                    lli = line.replace('\n', '').split(',')[1::]
                    newrow = labels_list[li] # append label to new row
                    # replace NAs with median values
                    lli_median = np.median([float(ii) for ii in lli
                        if not ii in lnotfloat])
                    lli_format = [float(ii) if not ii in lnotfloat
                                    else lli_median for ii in lli]
                    lli_fh = feature_hash(lli_format, target_dim = ndim)
                    newrow = newrow + ',' + ','.join([str(ii) for ii in lli_fh])
                    newrow = newrow + '\n'
                    print('Writing new row: '+newrow[0:100])
                    fw.write(newrow)
                    print("Finished with line number "+str(li))
    return None

def make_hnsw_si(fname, index_name, dindex_name, space_val = 'l2', threads = 10,
    efc_val = 2000, m_val = 1000, ef_val = 2000):
    """ Makes a new search index from a csv table

    Arguments:
        * fname: Name of CSV file containing index data.
        * index_name: Name of new search index file to save.
        * dindex_name: Name of new index dict, with str labels, to save.
        * space_val: Space value for the index.
        * threads: Number of threads for processing the index.
        * efc_val: EFC value for the index.
        * m_val: M value for the index.
        * ef_val: EF value for the index.
    Returns
        * None, saves the new index and index dict

    """
    print("Loading the dataset...")
    df=pd.read_csv(fname, sep=',',header=None)
    df_str_labels = df.iloc[0::,0]
    df = df.iloc[0::,1::] # subset to exclude string labels
    num_elements = df.shape[0]
    dim = df.shape[1]
    print("Making the new index...")
    p_iter = hnswlib.Index(space = space_val, dim = dim)
    p_iter.set_num_threads(threads)
    df_labels = np.arange(num_elements)
    p_iter.init_index(max_elements = num_elements,
        ef_construction = efc_val, M = m_val)
    p_iter.set_ef(ef_val)
    print("Adding new data to the search index...")
    p_iter.add_items(df, df_labels)
    print("Saving the search index...")
    p_iter.save_index(index_name)
    print("Saving the index dict with str labels, as pickle object...")
    dindex = {'index' : p_iter, 'strlabels' : df_str_labels}
    file_open = open(dindex_name, "wb")
    pickle.dump(obj = dindex, file = file_open)
    return None

def query_si(query_data, si_fname, si_labels = [], kval = 2):
    """ Perform a lookup on an hnswlib-saved search index

    Arguments:
        * query_data : Vector of feature hashed data, of dims R x C, where C is
   the
                same as in siobject/search index object.
        * si_fname : Name/path to the queried search index.
        * si_labels : Sample ID labels corresponding to index positions in
```

```
            search index at `si_fname`.
        * kval : K number of nearest neighbors to return from siobject lookup.

    Returns:
        * Vector of elements, where elements are indices if si_labels = False,
                or else element labels if si_labels is provided.

    """
    time_start = time()
    print("Loading search index...")
    sidict = pickle.load(open(si_fname, "rb"))
    siobject = sidict["index"]
    print("Querying "+str(query_data.shape[0])+" elements in data with k = "+
        str(kval)+" nearest neighbors...")
    kval_labels, kval_distances = siobject.knn_query(query_data, k = kval)
    time_str = str(time()-time_start);print("Query completed, time: "+time_str)
    if len(si_labels) > 0:
        print("Applying labels to query results..."); kval_str_labels = []
        for ll in kval_labels:
            kval_str_labels.append([si_labels[ii] for ii in ll])
        print("Returning data (sample id, k index, and distance)...")
        return kval_str_labels, kval_labels, kval_distances
    else:
        print("Returning data (k index and distance)...")
        return kval_labels, kval_distances
    return NULL

def make_dfk_sampleid(sample_idv, lk = [1,2], fh_csv_fname = "bval_100_fh10.csv",
    index_dict_fname = "new_index_dict.pickle"):
    """ make_dfk_sampleid

    Get the sample labels for the k nearest neighbors on a series of queries.
    Use a vector gsmv to identify samples for the query, from metadata.

    Arguments:
        * sample_idv : Vector or list of sample ID strings, corresponding to
            sample ID labels in the rownames/first column of the hashed features
            table at `fh_csv_fname`, which can correspond to sample names in the
            queried search index (requried, list/vector of strings).
        * lk : List of k nearest neighbors to query (required, list of int
            values, 1000).
        * index_dict_fname: Name/path of the search index file (required,
            string).
        * fh_csv_fname: Name/path of the hashed features csv to read (required,
            string, )
    Returns:
        * dfk_final

    """
    print("Getting hashed features data for samples...")
    fhdict = {}
    with open(fh_csv_fname, "r") as of:
        for line in of:
            lline = line.split(",")
            sample_id = lline[0].replace('"', '').split(".")[0]
            if sample_id in sample_idv:
                print("Getting index data for sample: '{0}'".format(sample_id))
                fhdict[sample_id] = [float(fhi.replace("\n", ""))
                    for fhi in lline[1::]
                ]
    dfi = pd.DataFrame.from_dict(fhdict).T
    si_dict = pickle.load(open(index_dict_fname, "rb"))
    dim_si_keylabv = len([ii for ii in si_dict['strlabels']])
    dfk = pd.DataFrame()
    dfk["sample_id"] = fhdict.keys()
    for ii, ki in enumerate(lk):
        print("Querying ",ki," neighbors from lk...")
        if ki <= dim_si_keylabv:
            kval_str_labels, kval_labels, kval_distances = query_si(
                query_data = dfi, si_fname = index_dict_fname,
                si_labels = si_dict["strlabels"], kval = ki
            )
            kli_format = [";".join(ii) for ii in kval_str_labels]
            dfki = pd.DataFrame(kli_format)
            dfk['k=' + str(ki)] = [ii for ii in dfki[0]]
        else:
            print("Provided k '{0}' > n si samples, skipping...".format(ki))
    print("Returning query results...")
    return dfk

def main():
    """
    """
```

A.9 Example parallelization script

The following code depicts parallelization of study ID bias correction simulations. These are described in Sections 3.2.4, and 3.3.2, and depicted in the workflow diagram C.1. In brief, random sets of studies and probes are selected from a data compilation. For each simulation rep, ANOVAs are performed on each of 3 models using uncorrected Beta-values, study ID bias-corrected Beta-values from 2-4 studies, and study ID bias-corrected Beta-values from 5 studies, respectively. Models are defined in a context-sensitive manner, so that the platform variable is only included when more than one platform is represented among selected studies, etc. Note the aov function is used to expedite ANOVAs, and that each simulation rep is broken into a parallel session.

```
#!/usr/bin/env R

# Author: Sean Maden
#
# Simulate the result of GSE bias corrections on variances, inc.
# a general GSE bias correction across available studies (adj1) and
# a precise GSE bias correction on a subset of studies evaluated (adj2).
# For comparison, the variances from unadjusted DNAm is also calculated.
#
# Main script steps:
# 1. Load the grset (autosomal probes noob-norm only, not GSE-adjusted).
# 3. Filter samples for a given blood subgroup.
# 4. Run the simulations, evaluations across iterations of probes and
#     studies. Adjustments are performed in the same probes across
#     randomized study sets.
#
#

library(HDF5Array); library(minfi)
library(methyPre); library(limma); library(sva)
library(data.table)

#--------------------
# load data -- server
#--------------------
save.dpath <- file.path("home", "metamaden", "bioinfo_appnote", "manuscript_final_
    results")
# load noob-norm data (pre gse-adj)
gr.fname <- "gr-noob_h5se_hm450k-epic-merge_0-0-3"
gr <- loadHDF5SummarizedExperiment(file.path(save.dpath, gr.fname))

#--------------------------------
# helper functions, parallel method
#--------------------------------
```

```r
# get anova data from model string
get_aovdat <- function(aov.str, labstr = "", pheno_subset,
                       orderv = c("gse", "glint.epi.pc1", "glint.epi.pc2", "predage",
                                  "predcell.CD8T", "predcell.CD4T", "predcell.NK",
                                  "predcell.Bcell", "predcell.Mono", "predcell.Gran",
                                  "Residuals", "predsex", "platform")){
  xaov <- eval(parse(text = aov.str)); xdat <- xaov[[1]]
  namev <- gsub(" ", "", rownames(xdat[1])); xdat.iter <- xdat[,c(2,4,5)]
  typev <- c("sumsq", "fval", "pval")
  vdat <- unlist(lapply(1:3, function(ii){
    datii <- xdat.iter[,ii]; names(datii) <- namev
    for(vname in c("predsex", "platform")){
      if(!vname %in% namev){
        datii <- c(datii, "NA"); names(datii)[length(datii)] <- vname}}
    datii <- datii[order(match(names(datii), orderv))]
    names(datii) <- paste0(names(datii), "_", typev[ii])
    return(datii)}))
  names(vdat) <- paste0(names(vdat), "_", labstr)
  return(vdat)
}

# parallel function for gse reps
par_gserep <- function(gse_rep, num.studies.subset, gsev, pheno, cgidv, table.fpath){
  # get study subset
  message("Working on rep ", gse_rep, " with ", num.studies.subset," studies...")
  gsev_filt <- sample(gsev, num.studies.subset)
  message("Using studies ", paste0(gsev_filt, collapse = ", "), "...")
  pheno_subset <- pheno[pheno$gse %in% gsev_filt,]
  pheno_subset$gse <- droplevels(pheno_subset$gse)
  # get the adj2 data
  which.bunadj <- which(grepl("_unadj", colnames(pheno_subset)))
  bunadj_subset <- t(as.matrix(pheno_subset[,which.bunadj]))
  rownames(bunadj_subset) <- paste0(gsub("_unadj", "",
                                         rownames(bunadj_subset)), "_adj2")
  bval_adj2 <- removeBatchEffect(bunadj_subset, batch = pheno_subset$gse)
  pheno_subset <- cbind(pheno_subset, t(bval_adj2))
  # get the vector of model strings
  cnv_numeric <- c("glint.epi.pc1", "glint.epi.pc2", "predage",
                   colnames(pheno_subset)[grepl("predcell", colnames(pheno_subset))])
  lm.str <- paste0("~ gse + ", paste0(cnv_numeric, collapse = " + "))
  num.lvl.predsex <- length(unique(pheno_subset$predsex))
  num.lvl.platform <- length(unique(pheno_subset$platform))
  if(num.lvl.predsex > 1){lm.str <- paste0(lm.str, " + predsex")}
  if(num.lvl.platform > 1){lm.str <- paste0(lm.str, " + platform")}
  # get results for 3 adj levels as a matrix
  cnv_all <- colnames(pheno_subset)
  mrep <- do.call(cbind, lapply(c("unadj", "adj", "adj2"), function(labstri){
    message("Working on models of type '", labstri, "'...")
    cg_cnv <- cnv_all[grepl(paste0(".*_", labstri, "$"), cnv_all)]
    lm.str.vect <- paste0(cg_cnv, " ", lm.str, ", data = pheno_subset")
    aov.str.vect <- paste0("summary(aov(", lm.str.vect, "))")
    mrep <- do.call(rbind, lapply(aov.str.vect, get_aovdat,
                                  pheno_subset = pheno_subset,
                                  labstr = labstri))
    mrep <- as.data.frame(mrep, stringsAsFactors = F)
    return(mrep)}))
  mrep <- as.data.frame(mrep, stringsAsFactors = F)
  mrep$gse_rep <- gse_rep
  mrep$ngse <- num.studies.subset
  mrep$gsev <- paste0(unique(pheno_subset$gse), collapse = ";")
  mrep$cgid <- cgidv
  # write new results rows
  message("Writing new results rows to file ", table.fpath, "...")
  data.table::fwrite(mrep, file = table.fpath, sep = ",", row.names = F,
                      col.names = F, append = T)
  return(mrep)
}

# get all results for a series of study reps
get_mgserep <- function(ngse_rep, num.studies.subset, cgidv, pheno, table.fpath){
  gsev <- unique(pheno$gse) # get studies
  # format pheno colnames
  cnv_numeric <- c("predage", "glint.epi.pc1", "glint.epi.pc2",
                   colnames(pheno)[grepl("predcell", colnames(pheno))])
  cnv_factor <- c("predsex", "platform", "gse")
  for(c in cnv_numeric){pheno[,c] <- as.numeric(pheno[,c])}
  for(c in cnv_factor){pheno[,c] <- as.factor(pheno[,c])}
  # get results matrix, process reps in parallel
  message("Processing studies using ", ngse_rep, " cores...")
  mgserep <- do.call(rbind, mclapply(seq(ngse_rep), par_gserep,
                                     num.studies.subset = num.studies.subset,
                                     gsev = gsev, pheno = pheno, cgidv = cgidv,
                                     mc.cores = ngse_rep, table.fpath = table.
    fpath))
  return(mgserep)
}

# get the results of all reps of random study selections
get_mgse_all <- function(grf, ngsev, table.fpath, num.gse = 5,
                         ngse_rep = 10, num.probes = 500){
  message("Getting ",num.probes," random probes and ",
          num.gse, " random studies...")
  cgidv <- sample(rownames(grf), num.probes)
  gsev <- sample(unique(grf$gse), num.gse)
  grff <- grf[cgidv, colnames(grf[,grf$gse %in% gsev])]
  bval_unadj <- getBeta(grff)
  mval_unadj <- logit2(bval_unadj)
  message("Getting adjustment 1 data...")
  mval_adj <- removeBatchEffect(mval_unadj, batch = grff$gse)
  bval_adj <- ilogit2(mval_adj)
  message("Appending DNAm to pheno...")
  pheno <- colData(grff)
  rownames(bval_unadj) <- paste0(rownames(bval_unadj), "_unadj")
  rownames(bval_adj) <- paste0(rownames(bval_adj), "_adj")
  pheno <- cbind(pheno,
                 cbind(t(as.matrix(bval_unadj)),
                       t(as.matrix(bval_adj))))
  message("Beginning iterations of random study selection and ANVOAs...")
  mgse.all <- do.call(rbind, lapply(ngsev, function(num.studies.subset){
    get_mgserep(ngse_rep = ngse_rep, num.studies.subset = num.studies.subset,
                cgidv = cgidv, pheno = pheno, table.fpath = table.fpath)}))
  message("Finished all ANOVA iterations; returning...")
  return(mgse.all)
}

# parallel wrapper for ‘get_mgse_all()‘
par_mgse_all <- function(cgrep, grf, ngsev, table.fpath, num.gse = 5,
                         ngse_rep = 10, num.probes = 500){
  t1 <- Sys.time(); message("Beginning cgrep ", cgrep, "...")
  mgse_all <- get_mgse_all(grf = grf, ngsev = ngsev, num.gse = num.gse,
                           ngse_rep = ngse_rep, num.probes = num.probes,
                           table.fpath = table.fpath)
  mgse_all <- as.data.frame(mgse_all, stringsAsFactors = F)
  message("Finished cgrep ", cgrep, ", time: ", Sys.time() - t1)
```

```r
  mgse_all$cgrep <- cgrep; return(mgse_all)
}

# script to process a subgroup
do_subgroup_tests <- function(grf, num.cgrep = 5, ngsev = c(2, 3, 4), num.gse = 6,
     sepsym = ",",
                              ngse.rep = 5, num.probes = 100, num.batch = 30, seed
     = 1,
                              rda.fname = "mgse-adj-test_blood-group-all_2
    platforms.rda",
                              table.fname = "mgse-adj-test_blood-group-all_2
    platforms.csv",
                              save.dpath = file.path("home", "metamaden", "bioinfo
    _appnote",
                                                     "manuscript_final_results")){
  set.seed(seed);t1 <- Sys.time()
  # write new table, then write new lines as processes finish
  cnv <- c("gse", "glint.epi.pc1", "glint.epi.pc2", "predage", "predcell.CD8T",
           "predcell.CD4T", "predcell.NK", "predcell.Bcell", "predcell.Mono",
           "predcell.Gran", "Residuals", "predsex", "platform")
  cnv <- c(paste0(cnv, "_sumsq"), paste0(cnv, "_fval"), paste0(cnv, "_pval"))
  cnv <- c(paste0(cnv, "_unadj"), paste0(cnv, "_adj"), paste0(cnv, "_adj2"))
  cnv <- c(cnv, "gse_rep", "ngse", "gsev", "cgid")
  table.fpath <- file.path(save.dpath, table.fname)
  mcnv <- matrix(nrow = 0, ncol = length(cnv));colnames(mcnv) <- cnv
  data.table::fwrite(mcnv, file = table.fpath, sep = sepsym, append = F,
                     col.names = T, row.names = F)
  # do iterations, writing new lines as they complete
  for(batch in seq(num.batch)){
    message("Beginning batch ",batch, ", time: ", Sys.time() - t1)
    mclapply(seq(num.cgrep), par_mgse_all, grf = grf, ngsev = ngsev,
             num.gse = num.gse, ngse_rep = ngse.rep, num.probes = num.probes,
             mc.cores = num.cgrep, table.fpath = table.fpath)
  }
  if(file.exists(table.fpath)){return(TRUE)
  } else{
    message("Couldn't find new table file at ", table.fpath)
    return(FALSE)
  }
  return(NULL)
}

#---------------------------------------
# process subgroups -- server, test runs
#---------------------------------------
# process all -- takes about 20hr to complete on remote server
new.fname <- "mcg-gsebias-final_blood-4stypes-2platforms"
rda.fname <- paste0(new.fname, ".rda"); table.fname <- paste0(new.fname, ".csv")
mcg.test <- do_subgroup_tests(grf = gr, num.cgrep = 3, ngsev = c(2,3,4), num.gse =
     5,
                             ngse.rep = 3, num.probes = 500, num.batch = 20, seed
    = 1,
                             table.fname = table.fname, rda.fname = rda.fname)

# append colnames
csv.fname <- "mcg-gsebias-final_blood-4stypes-2platforms.csv"
csv.fpath <- file.path("home", "metamaden", "bioinfo_appnote",
                       "manuscript_final_results")
csv.fpath <- file.path(csv.fpath, csv.fname)
csv <- fread(csv.fpath, sep = ",", header = F, data.table = F)
# get colnames
cnv <- c("gse", "glint.epi.pc1", "glint.epi.pc2", "predage", "predcell.CD8T",
         "predcell.CD4T", "predcell.NK", "predcell.Bcell", "predcell.Mono",
         "predcell.Gran", "Residuals", "predsex", "platform")
```

```r
cnv <- c(paste0(cnv, "_sumsq"), paste0(cnv, "_fval"), paste0(cnv, "_pval"))
cnv <- c(paste0(cnv, "_unadj"), paste0(cnv, "_adj"), paste0(cnv, "_adj2"))
cnv <- c(cnv, "gse_rep", "ngse", "gsev", "cgid")
colnames(csv) <- cnv
# save new table
csv.fname <- "mcg-gsebias-final_blood-4stypes-2platforms.csv"
csv.fpath <- file.path(csv.fname)
fwrite(csv, file = csv.fpath, sep = ",", col.names = T, row.names = F)

#--------------------------------
# get variance fract by variables
#--------------------------------
save.dpath <- file.path("home", "metamaden", "bioinfo_appnote",
                        "manuscript_final_results")
# append colnames
csv.fname <- "mcg-gsebias-final_blood-4stypes-2platforms.csv"
csv.fpath <- file.path(save.dpath, csv.fname)
csv <- fread(csv.fpath, sep = ",", header = T, data.table = F)

# get sum of sq vars by model type
typev <- c("unadj", "adj1", "adj2")
varv <- c("gse", "glint.epi.pc1", "glint.epi.pc2", "predage", "predcell.CD8T",
          "predcell.CD4T", "predcell.NK", "predcell.Bcell", "predcell.Mono",
          "predcell.Gran", "Residuals", "predsex", "platform")
cnames.mfsq <- paste0(varv, "_", rep(typev, each = length(varv))) # groups by
   probe, group
lindex <- list("unadj" = 1:13, "adj" = 40:52, "adj2" = 79:91) # model type var
   indices
# get the list of fract sumsq by group
mcg <- csv
mfsq <- t(apply(mcg, 1, function(dati){
  do.call(cbind, lapply(lindex, function(indexv){
    datii <- dati[indexv]
    tot.rsq <- sum(as.numeric(datii), na.rm = t)
    fract.indexv.rsq <- as.numeric(datii)/tot.rsq
    names(fract.indexv.rsq) <- names(datii)
    return(fract.indexv.rsq)
  }))
}))
colnames(mfsq) <- cnames.mfsq
mfsq <- as.data.frame(mfsq, stringsAsFactors = F)
mfsq$ngse <- mcg$ngse
mfsq$cgid <- mcg$cgid
mfsq$gsev <- mcg$gsev

# save mfsq matrix
msq.fname <- "msq-gse-bias_all-blood-2-platforms.rda"
msq.fpath <- file.path(save.dpath, msq.fname)
save(mfsq, file = msq.fpath)

#--------------------------------
# get variance diffs by variables
#--------------------------------
# get 3x FEV differences
cname.diff <- paste0(rep(varv, each = 3), "_",
                     rep(c("diff1", "diff2", "diff3"),
                         times = length(varv)))
mdiff.all <- do.call(cbind, lapply(varv, function(vari){
  fii <- mfsq[,grepl(vari, colnames(mfsq))];namev <- names(fii)
  diff1 <- as.numeric(fii[,1]) - as.numeric(fii[,2])
  diff2 <- as.numeric(fii[,1]) - as.numeric(fii[,3])
  diff3 <- as.numeric(fii[,2]) - as.numeric(fii[,3])
  mdiff <- cbind(diff1, cbind(diff2, diff3))
  return(mdiff)}))
```

```r
mdiff.all <- as.data.frame(mdiff.all, stringsAsFactors = F)
colnames(mdiff.all) <- cname.diff
mdiff.all$ngse <- mfsq$ngse

# save mdiff.all matrix
mdiff.fname <- "mdiff-gse-bias_all-blood-2-platforms.rda"
mdiff.fpath <- file.path(save.dpath, mdiff.fname)
save(mdiff.all, file = mdiff.fpath)
message("done")
```

A.10 Feature hashing equations

We can describe feature hashing in detail by paraphrasing the descriptions from [174]. Suppose we wish to learn some predictor w_s, which derives from the set of array probes x for sample s such that $s \in \mathcal{S}$. We calculate the vector of hashed weights w_h, consisting of weight vectors $w_0 ... w_{\mathcal{S}}$, using the corresponding vector of signed hash functions $\phi_0 ... \phi_{\mathcal{S}}$ to derive $w_h \in \mathbb{R}^m$. We define the semi-positive kernel matrix k as:

$$k(x_i, x_j) := \langle \phi(x_i), \phi(x + j) \rangle \,. \tag{18}$$

We next use the concept of projecting input vectors x with unmodified dimensional space $\mathbb{R}^d$ into the low-dimensional space $\mathbb{R}^m$ with some hash function ϕ [175].

$$\phi : \mathcal{X} \to \mathbb{R}^m \,. \tag{19}$$

$$\phi_i^{(h,\xi)}(x) = \sum_{j:h(j)=i} \xi(i) x_i \,. \tag{20}$$

$$(x, x^{'})_\phi := \langle \phi^{(h,\xi)}(x), \phi^{h,\xi}(x^{'}) \rangle \,. \tag{21}$$

This can be represented as:

$$w_h = \phi_0(w_0) + \sum \phi_s(w_s) \,. \tag{22}$$

We can now write the output for x and w_h with errors from two sources: (1) interference between samples, ϵ_i; and (2) distortions within each sample, ϵ_d.

$$\langle \phi_0(x) + \phi_u(x), w_h \rangle = \langle x, w_0 + w_u \rangle + \epsilon_d + \epsilon_i \,. \tag{23}$$

$$\epsilon_i = \sum_{v \in \mathcal{S}, v \neq 0} \langle \phi_s(x), \phi_v(w_v) \rangle + \sum_{v \in S}, v \neq s \langle \phi_s(x), \phi_v(w_v) \rangle \,. \tag{24}$$

$$\epsilon_d = \sum_{v \in \{u,0\}} |\langle \phi_v(x), \phi_v(w_v) \rangle - \langle x, w_v \rangle| \,. \tag{25}$$

ϵ_i results from collisions between hashed features for a sample s, x_s, and hashed features for remaining samples v, x_v. By contrast, ϵ_d results for hashed features from the same sample s, or $\langle x, w_v \rangle$, which result from internal self-collisions among features in s. It can be shown empirically and in theory that both ϵ_d and ϵ_i are small with high probability [174].

B Supplementary Material for “Human methylome variation across Infinium 450K data on the Gene Expression Omnibus”

B.1 Supplementary Figures

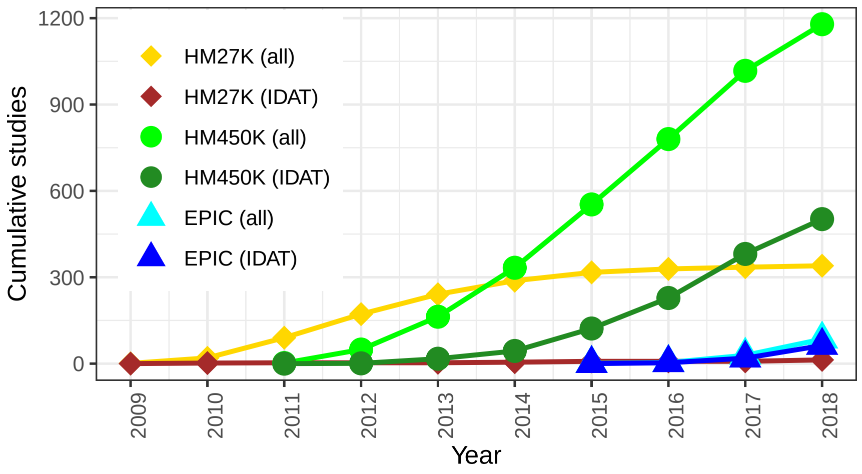


Figure B.1: Cumulative yearly DNAm array studies in GEO for three platforms: HM27K (diamonds, yellow is all studies, brown is studies with IDATs), HM450K (circles, light green is all studies, dark green is studies with IDATs), and EPIC (triangles, light blue is all studies, and dark blue is studies with IDATs).

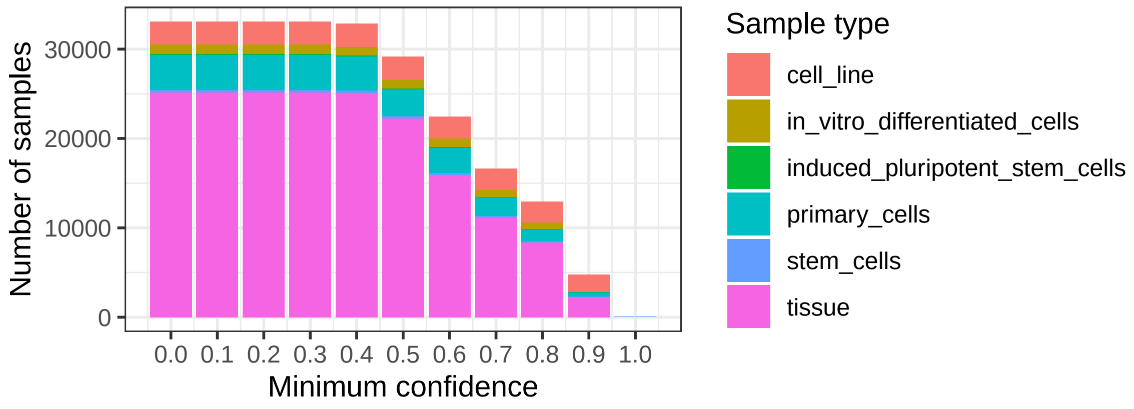


Figure B.2: Most likely sample types (colors) predicted from MetaSRA-pipeline, by maximum confidence cutoffs (x-axis, barplots).

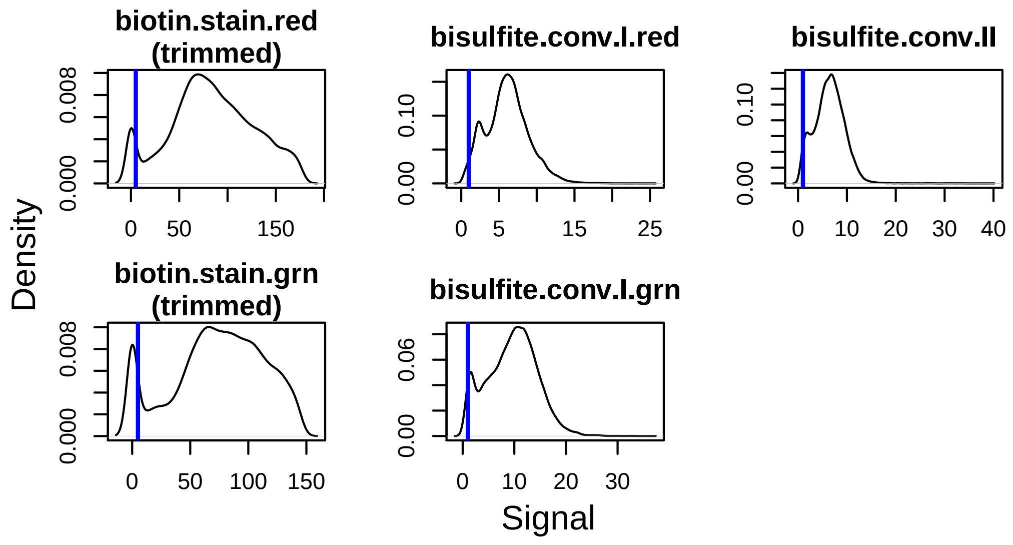


Figure B.3: Signal density plots across five Illumina BeadArray controls, showing trimmed sample frequencies (x-axis is signal, y-axis is density) and published minimum signal thresholds (vertical blue lines).

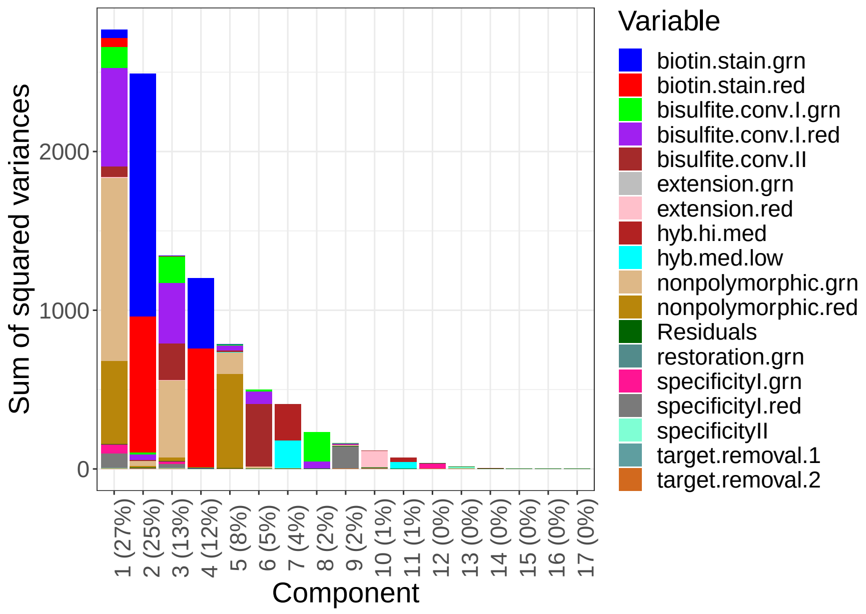


Figure B.4: Screeplot from PCA of BeadArray control pass status (i.e., pass = 1, fail = 0) across samples, where the y-axis denotes the sum of the squared variance, the x-axis denotes principal component. Percentages in labels are corresponding component contributions to total variance. Colors capture magnitudes of component variances explained by each model variable from analyses of variance.

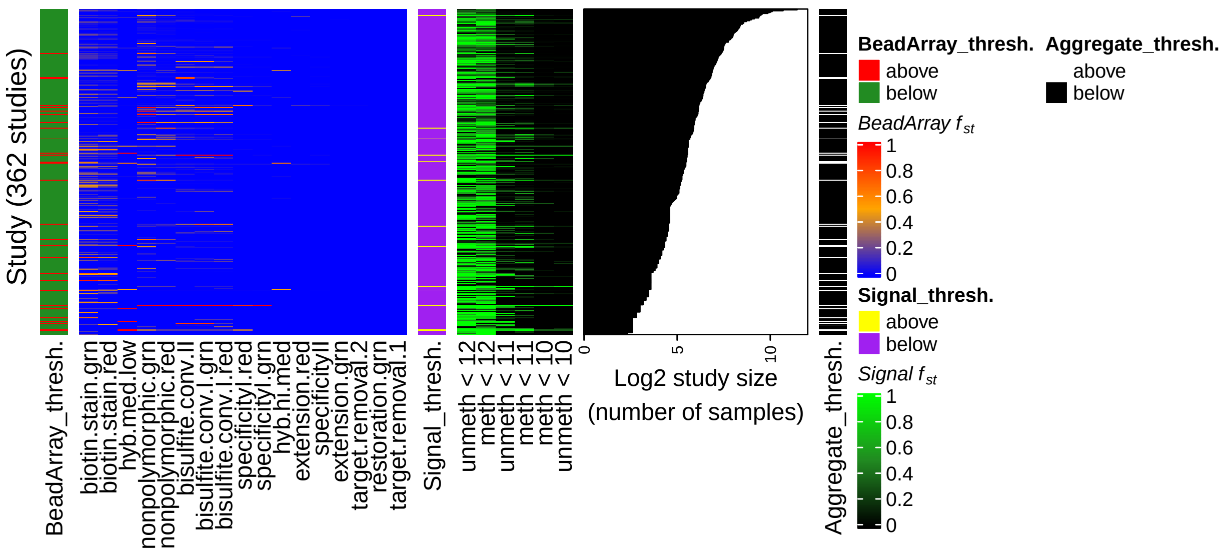


Figure B.5: Heatmaps depicting fraction (f_{st} in legends) of samples in a study failing quality assessments across 362 studies. *BeadArray* f_{st} values are shown on the left, where blue is low, orange is intermediate, and red is high. *Signal* f_{st} values for three methylated (M, “meth") and unmethylated (U, “unmeth") signal levels (10, 11, and 12) are shown in the middle, where black is low, dark green intermediate, and light green is high. The $\log_2$ study sizes are shown in barplots on the right. Annotations include (far left annotation) whether a study showed >0.6 frequency for the *BeadArray* f_{st} (“BeadArray_thresh.", red >0.6, green is <0.6), (middle annotation) whether a study showed >0.6 frequency for *Signal* f_{st} (“Signal_thresh.", minimum signal level 11, yellow >0.6, purple <0.6), and (far right) whether a study was above >0.6 frequency for either f_{st} metric (“Aggregate_thresh.", white is either metric >0.6, black is neither metric >0.6).

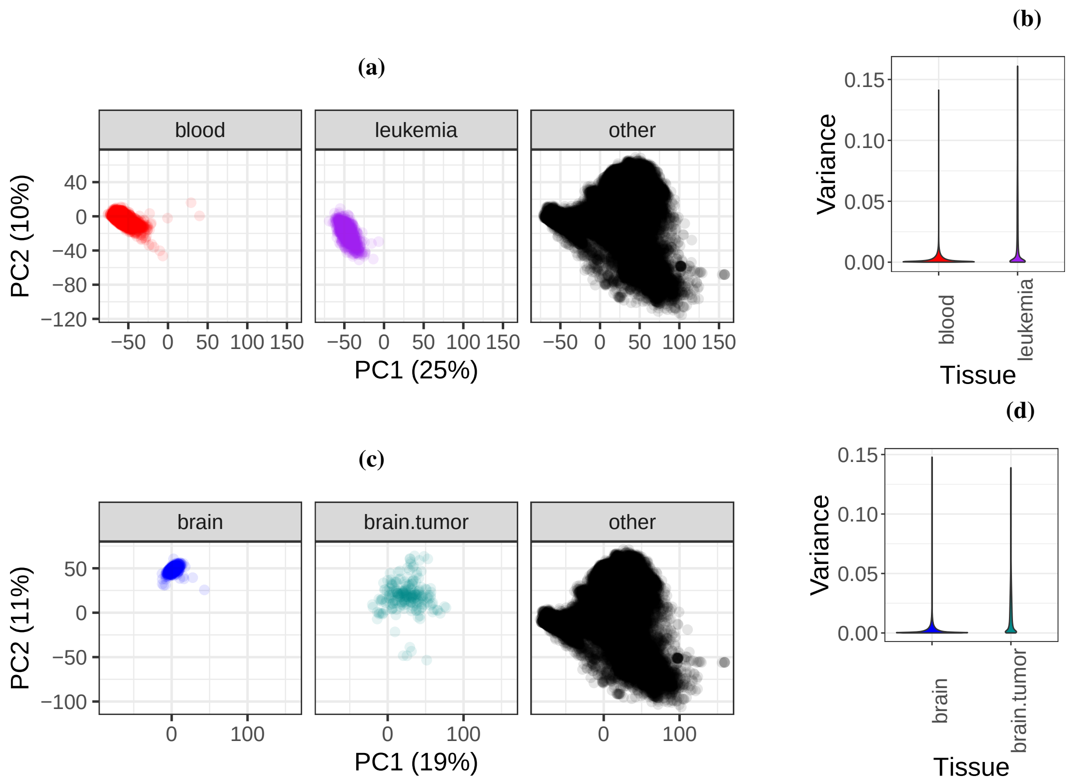
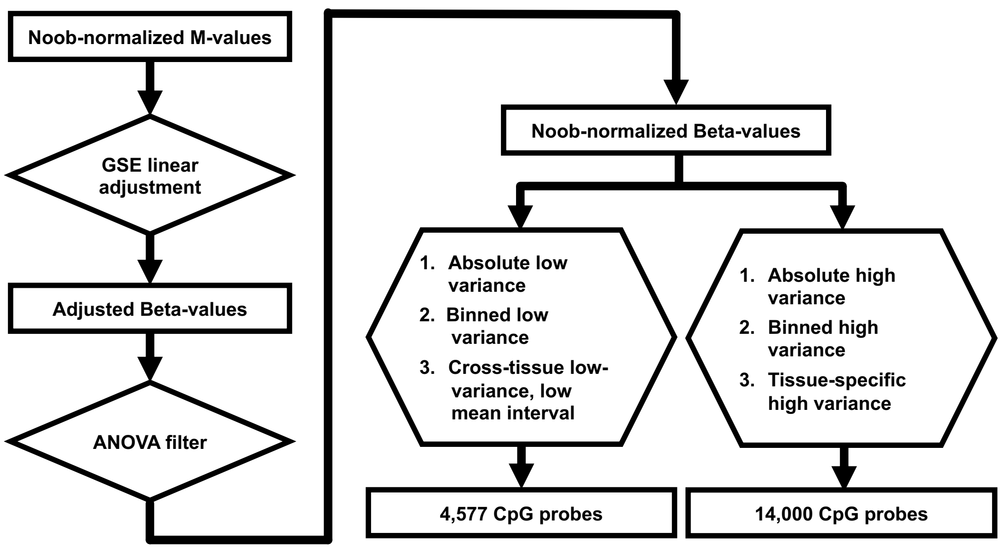
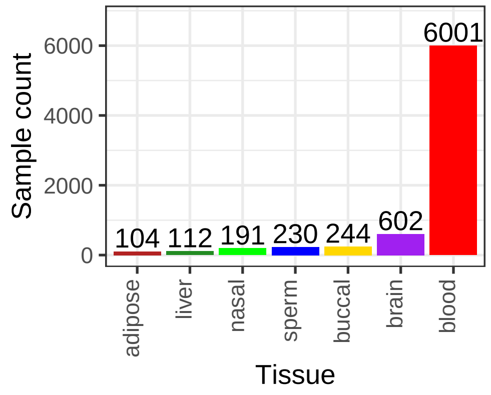


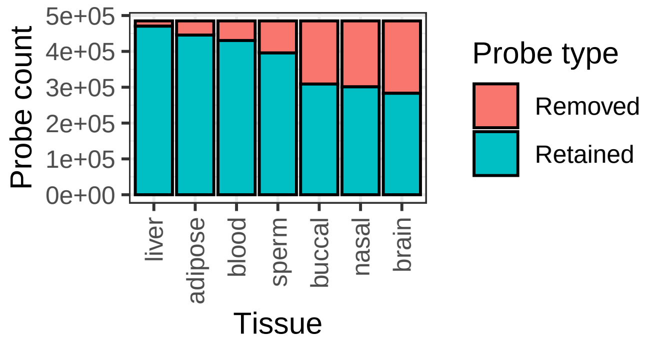
Figure B.6: PCA facet plots and probe variance distributions of autosomal DNAm (noob-normalized Beta-values). (a) Facet plots of non-cancer blood (left, red), leukemias (middle, purple), and remaining samples (right, black) from Fig. B.6a. (b) Violin plots of DNAm variances in blood and leukemia samples. (c) Facet plots of non-cancer brain (left, blue), brain tumors (middle, dark cyan), and remaining samples (right, black) from Fig. B.6c. (d) Violin plots of DNAm variances in brain and brain tumor samples.



(a)



(b)



(c)

Figure B.7: Analysis of autosomal DNAm variation across 7 non-cancer tissues (adipose, blood, brain, buccal, nasal, liver, and sperm). (a) Workflow to normalize, preprocess, and analyze DNAm variation within 7 tissues (adipose, buccal, brain, liver, sperm, nasal, and blood), including references to relevant figures. (b) Numbers of available samples by tissue type, with number labels showing barplot amounts. (c) Number of probes removed and retained from ANOVA filters within tissues (2.2.10).

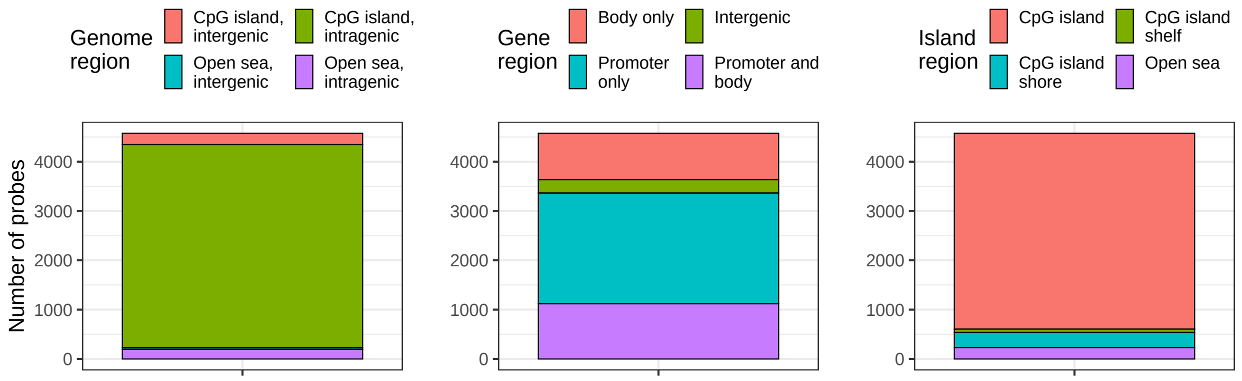


Figure B.8: Genome mappings among 4,577 CpG probes with low variance across 7 tissues (adipose, brain, buccal, nasal, blood, liver, and sperm, 2.2.10, Fig. B.7). Color fills depict (left) CpG island and gene region overlaps, (middle) gene region overlaps, and (right) CpG island region overlaps.

B.2 Supplementary Tables

gsm	gsm_title	gseid	disease	tissue
GSM1873723	SOG048 3999876086_R02C02	GSE72874;GSE72872	cancer	distal;esophagus;blood
GSM3584264	NA	GSE125895	NA	NA
GSM1858832	Breast tumor from TOP Cohort patient P_41	GSE72308;GSE72254	cancer	tumor;breast
GSM2293090	CellLine_COLO201	GSE86078	cancer	colorectal;intestine;colon;rectum
GSM2724331	9904796033_R03C02	GSE102119;GSE102120	cancer	ovary;endocrine_system
GSM2565812	393N_normal	GSE97466	normal	NA
GSM2792160	EPN-PFA SAMPLE 486	GSE104210	cancer	brain
GSM1670362	CellLine_RCC-FG2	GSE68379	cancer	kidney
GSM2941227	MB, G4, sample 503 [validation set]	GSE109379;GSE109381	cancer	brain
GSM1873911	PAH046 9611518135_R01C02	GSE72874;GSE72872	normal;healthy;control	distal;esophagus;blood
GSM2814369	whole blood taken from sample 7342	GSE105018	normal	blood;whole_blood
GSM1858758	Breast tumor from Cohort 2 patient P_86	GSE72251;GSE72308	cancer	tumor;breast
GSM1236074	6969568028_R03C02	GSE51057;GSE51032	normal;healthy;control	blood
GSM1425640	SAMPLE_61_CD4	GSE59065	NA	NA
GSM1633025	Sample146_Tumor	GSE66836	cancer	tumor
GSM2122901	E0007_FASD	GSE80261	NA	oral;buccal;throat;epithelial
GSM1501445	SAMPLE_44_Adult	GSE61278;GSE61279	NA	liver
GSM2756932	genomic DNA from gastric normal 33	GSE103186	normal;healthy;control	stomach
GSM2405127	CHORDM, sample 2271 [reference set]	GSE90496;GSE109381	cancer	NA
GSM3053777	Genomic DNA from prostate_4_normal	GSE112047	normal;healthy;control	prostate
GSM1616987	maternal_whole_blood_rep2	GSE66210	NA	blood;whole_blood
GSM1235863	5809079064_R02C01	GSE51032	cancer;breast_cancer	breast;blood
GSM2817867	IPSPDL1.6L_DN	GSE105093	NA	neuron;stem_cell
GSM2333940	X40 genomic DNA from whole blood	GSE87571	normal	blood;whole_blood
GSM2403835	MNG, sample 982 [reference set]	GSE90496;GSE109381	cancer	brain;oral
GSM2818043	NA	GSE105109	NA	NA
GSM1922528	Buffy_Placebo_Baseline_Subject_18	GSE74548	NA	NA
GSM1586910	Ind4_F-iPSC [Methylation]	GSE65079;GSE65078	NA	stem_cell
GSM1614000	iPSC PiZZ T0 3	GSE66077;GSE66078	NA	stem_cell
GSM1848002	gDNA_CD4_Control_rep11	GSE71955;GSE71957	healthy;control	blood;white_blood_cell;t_cell
GSM2403828	EPN, MPE, sample 975 [reference set]	GSE90496;GSE109381	cancer	NA
GSM2402879	DMG, K27, sample 26 [reference set]	GSE90496;GSE109381	cancer	blood;neuron
GSM2815379	whole blood taken from sample 2911	GSE105018	normal	blood;whole_blood

Table 6.1: Postprocessed metadata for 35,360 samples (rows). Columns 1 and 2 are the GSM record ID and record title. Column 3 is the collapsed GSE record ID (see Methods and Supplemental Information). Columns 4 and 5 are variables with learned labels for disease/study group and tissue characteristics. Column 6 is the most likely predicted sample type. Columns 7 and 8 are the array ID and basename for the sample IDATs. Column 9 is the mined age values with available units. Column 10 is the predicted age. Columns 11 and 12 are the mined and predicted sex labels. Columns 13-18 are predicted blood cell fractions. Column 19 is the annotated storage procedure. Preview of supplemental table; full table available online.

Table 6.2: Quality and summary metrics for 35,360 samples (rows, see Methods and Supplemental Information). Column 1 is the GSM ID. Column 2 is the GSE ID. Columns 3-19 are the BeadArray quality metrics. Columns 20-21 are the log2 medians of methylated and unmethylated signals. Column 22 shows the GSM IDs likely sharing genetic identity from the same GSE record. Column 23 is the quantity of samples from column 22. Preview of supplemental table; full table available online.

gsm	gseid	ba.restoration.grn	ba.biotin.stain.red	ba.biotin.stain.grn	ba.specificityI.red
GSM1873723	GSE72874;GSE72872	0.02	731.5	496.6	13.51
GSM2459564	GSE93646	0.09	119.13	74.56	4.48
GSM3584264	GSE125895	0.09	183.02	85.48	7.36
GSM2363655	GSE89278	0.06	46.01	0	6.66
GSM1858832	GSE72308;GSE72254	0.11	64.99	48.72	5.49
GSM2293090	GSE86078	0.06	74.42	39.39	5.11
GSM2465258	GSE93933	0.08	54.49	83.44	6.6
GSM2724331	GSE102119;GSE102120	0.11	108.42	91.47	6.69
GSM2565812	GSE97466	0.13	25.9	14.27	3.98
GSM2792160	GSE104210	5.22	8593	19.16	2.35
GSM1670362	GSE68379	0.1	60.38	51.32	3.69
GSM2941227	GSE109379;GSE109381	6.29	111.03	9.18	5
GSM1873911	GSE72874;GSE72872	0.03	199.94	257.12	8.64
GSM2814369	GSE105018	0.07	150.3	289.59	7.06
GSM1858758	GSE72251;GSE72308	0.12	40.28	69	6.67
GSM1236074	GSE51057;GSE51032	0.08	70.98	31.24	4.41
GSM1425640	GSE59065	0.08	53.96	62.01	4.8
GSM3025701	GSE111223	0.05	10887	131.21	4.52
GSM1633025	GSE66836	0.05	462.82	217.64	5.42
GSM2122901	GSE80261	0.11	87.81	64.3	4.07
GSM2363592	GSE89278	0.04	119.33	256.44	6.5
GSM1501445	GSE61278;GSE61279	0.09	61.85	55.61	3.76
GSM2756932	GSE103186	0.04	43.43	0	3.63
GSM2905416	GSE108576	2.16	26.65	12.53	9.16
GSM2405127	GSE90496;GSE109381	7.31	457.29	157.65	5.92
GSM3053777	GSE112047	0.04	6.44	0	3.71
GSM1616987	GSE66210	0.11	121.74	63.04	8.04
GSM1235863	GSE51032	0.03	41.49	20	4.45
GSM2817867	GSE105093	0.04	75.79	90.74	5.78
GSM2333940	GSE87571	0.07	131.34	156.33	5.77
GSM2403835	GSE90496;GSE109381	0.11	155.77	79.4	7.53
GSM2818043	GSE105109	0.03	0	290.88	6.26
GSM1203553	GSE49032;GSE49031	0.06	116.17	101.06	5.02

Table 6.3: Information about the 17 BeadArray quality metrics (rows, Methods, Supplemental Information). Column 1 is the metric name. Column 2 is the metric description. Column 3 is the published minimum threshold, and column 4 is the formula calculation from control probe signals. Preview of supplemental table; full table available online.

name	description	threshold	formula
Restoration Green	Extent of DNA restoration success for FFPE samples	0	green signal/background
Biotin Staining Red	Efficiency of staining step, red color channel	5	dnp high/dnp background
Biotin Staining Green	Efficiency of staining step, green color channel	5	biotin high/biotin background
Specificity I Red	Non-specific primer extension at non-polymorphic T site, Infinium I probes, red channel	1	min(PM)/max(MM)
Specificity I Green	Non-specific primer extension at non-polymorphic T site, Infinium I probes, green channel	1	min(PM)/max(MM)
Specificity II	Non-specific primer extension at non-polymorphic T site, Infinium II probes	1	min(S1.R, S2.R, S3.R)/max(S1.G, S2.G, S3.G)
Extension Red	Sample-independent extension efficiency at hairpin probe, A and T nucleotides, red channel	5	min(C or G)/max(A or T)
Extension Green	Sample-independent extension efficiency at hairpin probe, C and G nucleotides, green channel,	5	min(C or G)/max(A or T)
Hybridization, High-Medium	Overall assay performance using high- versus medium-concentration synthetic targets	1	high concentration/medium concentration
Hybridization, Medium-Low	Overall assay performance using medium- versus low-concentration synthetic targets	1	medium concentration/low concentration
Target Removal 1	Efficiency of stripping step after extension reaction for Infinium I probes, from green channel	1	background/control intensity

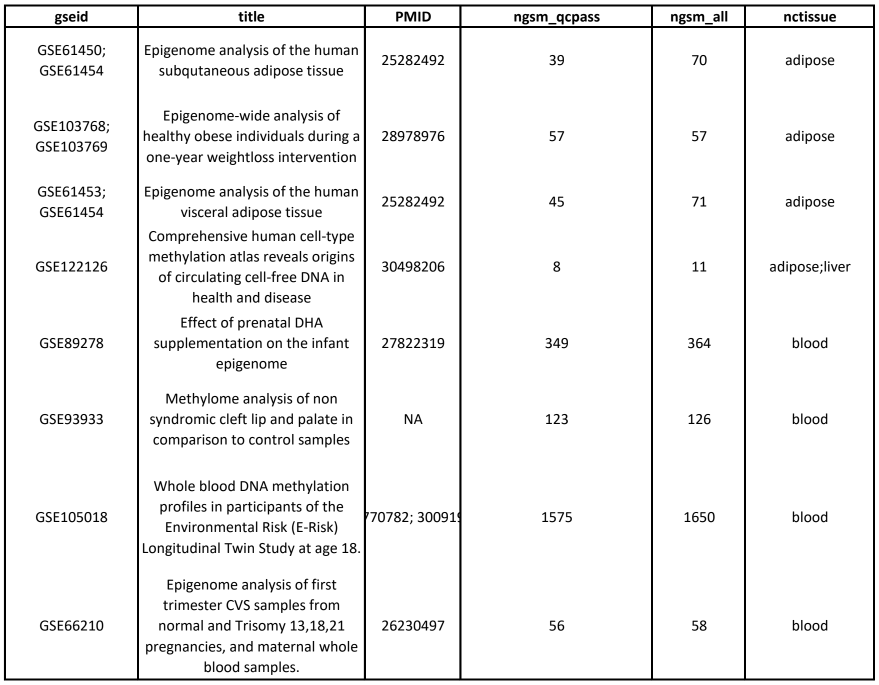
Table 6.4: Characteristics of studies (rows) for storage condition analyses. Column 2 is the number of samples with available storage information. Column 3 is the quantity of FFPE samples. Column 4 is the quantity of fresh frozen samples. Columns 5 and 6 are the unions of unique tissue and disease labels, respectively. Preview of supplemental table; full table available online.

gseid	num.gsm	num.ffpe	num.frozen	tx.terms	dx.terms
GSE109379;GSE109381	1087	1087	0	brain;blood;endocrine_system;oral;glia; cerebellum;NA;neuron;epithelial;ectoderm; nervous_system	cancer
GSE105018	1650	0	1650	blood;whole_blood	normal
GSE108576	87	87	0	metastasis;brain;respiratory_system;lung; breast;tumor	cancer;skin_cancer; lung_cancer; breast_cancer
GSE90496;GSE109381	2778	1859	919	NA;brain;oral;blood;neuron;cerebellum;endocrine_system;nervous_system;white_blood_cell; t_cell;epithelial;eye;rod;lymphatic_system; nasal;cancer;ectoderm;glia;chest;neck	cancer;normal;healthy; control
GSE112047	47	47	0	prostate;tumor	normal;healthy; control;cancer
GSE87571	725	0	725	blood;whole_blood	normal
GSE105109	381	0	381	NA	NA
GSE104293	132	87	45	NA	NA
GSE74193	671	0	671	brain	NA;control
GSE66351	189	0	189	brain;neuron;oral;glia	NA
GSE114753	155	0	155	sperm	control;NA
GSE71678	337	0	337	placenta	normal
GSE60185	279	0	279	breast	cancer;breast_cancer; normal
GSE65163;GSE65205	71	0	71	nasal;epithelial	NA
GSE104210	63	39	24	brain;blood;neuron	NA
GSE103659	42	42	0	tumor;brain;oral;cerebellum	cancer
GSE102970	47	0	47	sperm	NA
GSE73549	92	92	0	prostate;metastasis; lymphatic_system;tumor	normal;cancer;NA
GSE103768;GSE103769	57	0	57	adipose	NA
GSE107352;GSE107353	51	51	0	tumor;colorectal;intestine; colon;rectum;mucosa	cancer;case;normal; healthy;control
GSE122126	11	0	11	neuron;adipose;liver;endocrine_system; pancreas;blood;white_blood_cell;t_cell	normal;NA

Table 6.5: Sample sub-threshold frequencies (fst) for 362 studies. Column 1 is the GSE accession number for the study record. Column 2 is the number of GSM records. Columns 3-19 are frequencies of samples below published thresholds for 17 BeadArray metrics. Columns 20-25 are frequencies of samples below log2 median methylated and unmethylated signal thresholds of 10, 11, and 12. Column 26 shows if fst is >5% for a threshold of 11 in both methylated and unmethylated signals. Column 27 shows if fst is >5% for samples failing at least 1 BeadArray metric. Preview of supplemental table; full table available online.

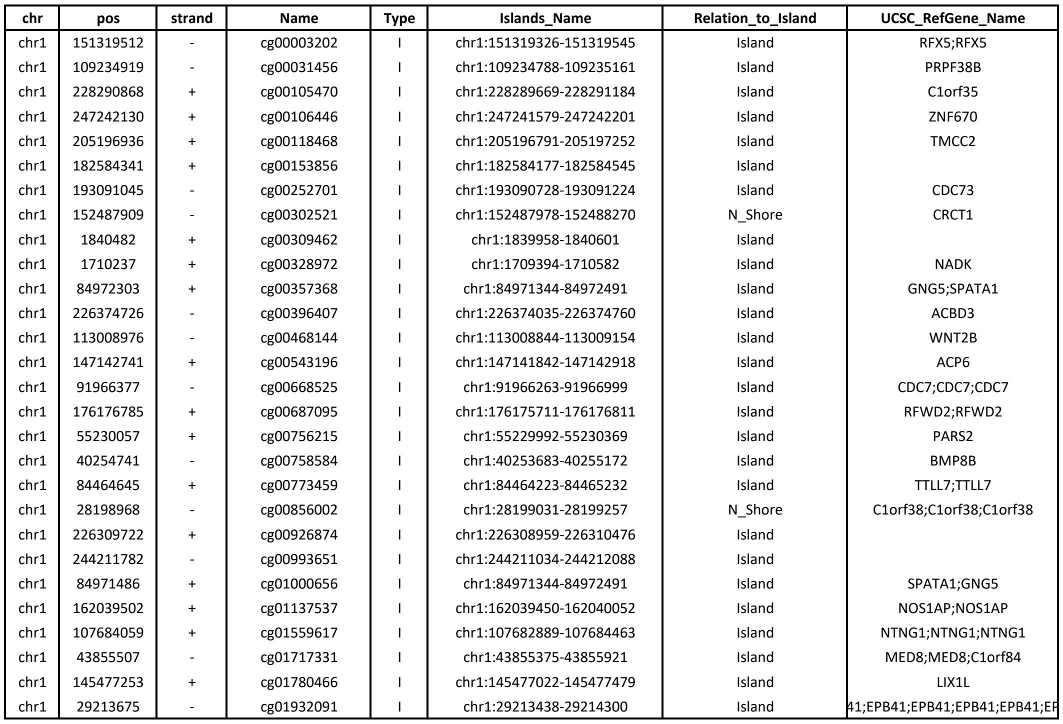
gseid	ngsm	restoration.grn	biotin.stain.red	biotin.stain.grn	specificityI.red	specificityI.grn
GSE72874;GSE72872	248	0	0.03	0.03	0	0
GSE93646	423	0	0	0.01	0	0
GSE125895	267	0	0	0.01	0	0
GSE89278	364	0	0.02	0.04	0	0
GSE72308;GSE72254	58	0	0	0	0	0
GSE86078	146	0	0.11	0.14	0	0
GSE93933	126	0	0	0	0	0
GSE102119;GSE102120	146	0	0.05	0.14	0.02	0.02
GSE97466	140	0	0	0	0	0
GSE104210	666	0	0.09	0.11	0	0
GSE68379	1018	0	0	0	0	0
GSE109379;GSE109381	1097	0	0.09	0.16	0	0
GSE105018	1650	0	0.1	0.17	0	0
GSE72251;GSE72308	115	0	0	0	0	0
GSE51057;GSE51032	326	0	0	0	0	0
GSE59065	293	0	0	0	0	0
GSE111223	256	0	0.14	0.21	0	0
GSE66836	181	0	0	0.01	0.02	0
GSE80261	214	0	0	0	0	0
GSE61278;GSE61279	110	0	0	0	0	0
GSE103186	187	0	0.1	0.14	0	0
GSE108576	87	0	0.15	0.15	0	0
GSE90496;GSE109381	2778	0	0.03	0.05	0	0
GSE112047	47	0	0.15	0.11	0	0
GSE66210	58	0	0	0	0	0
GSE51032	512	0	0	0	0	0
GSE105093	11	0	0	0	0	0
GSE87571	725	0	0	0	0	0

Table 6.6: Study (rows) information for DNAm variability analyses in 7 tissues (Methods, Supplemental Information). Column 1 is the GSE record ID. Column 2 is the record title. Column 3 is the available PubMed ID. Column 4 is whether the study includes samples from cancer patients. Column 5 is the percent of samples removed during quality control and pre-filters. Column 6 is the number of samples passing quality checks and pre-filters. Column 7 is the total number of sample records. Column 8 is the tissue type(s) among retained samples. Column 9 is the number of samples used in the DNAm variability analyses. Column 10 is the annotated sample labels and information. Column 11 is condition or disease info. Column 12 is the available study population information. Preview of supplemental table; full table available online.



gseid	title	PMID	ngsm_qcpass	ngsm_all	nctissue
GSE61450; GSE61454	Epigenome analysis of the human subqutaneous adipose tissue	25282492	39	70	adipose
GSE103768; GSE103769	Epigenome-wide analysis of healthy obese individuals during a one-year weightloss intervention	28978976	57	57	adipose
GSE61453; GSE61454	Epigenome analysis of the human visceral adipose tissue	25282492	45	71	adipose
GSE122126	Comprehensive human cell-type methylation atlas reveals origins of circulating cell-free DNA in health and disease	30498206	8	11	adipose;liver
GSE89278	Effect of prenatal DHA supplementation on the infant epigenome	27822319	349	364	blood
GSE93933	Methylome analysis of non syndromic cleft lip and palate in comparison to control samples	NA	123	126	blood
GSE105018	Whole blood DNA methylation profiles in participants of the Environmental Risk (E-Risk) Longitudinal Twin Study at age 18.	770782; 30091	1575	1650	blood
GSE66210	Epigenome analysis of first trimester CVS samples from normal and Trisomy 13,18,21 pregnancies, and maternal whole blood samples.	26230497	56	58	blood

Table 6.7: Characteristics of DNAm array CpG probes (rows) with recurrent low Beta-value variances and mean intervals across 7 non-cancer tissues. Column 1 is the probe ID. Column 2 is the chromosome. Column 3 is the genome coordinate. Column 4 is the DNA strand. Column 5 is the assay type. Column 6 is the CpG island name. Column 7 is the CpG island region type. Column 8 is the gene ID(s). Column 9 is the gene accession(s). Column 10 is the gene region group(s). Columns 11-24 are the Beta-value variances and means across samples for each tissue (tissues: adipose, blood, brain, buccal, liver, nasal, and sperm). Column 25 is the maximum absolute mean interval across all pairwise comparisons of tissue Beta-value means. Preview of supplemental table; full table available online.



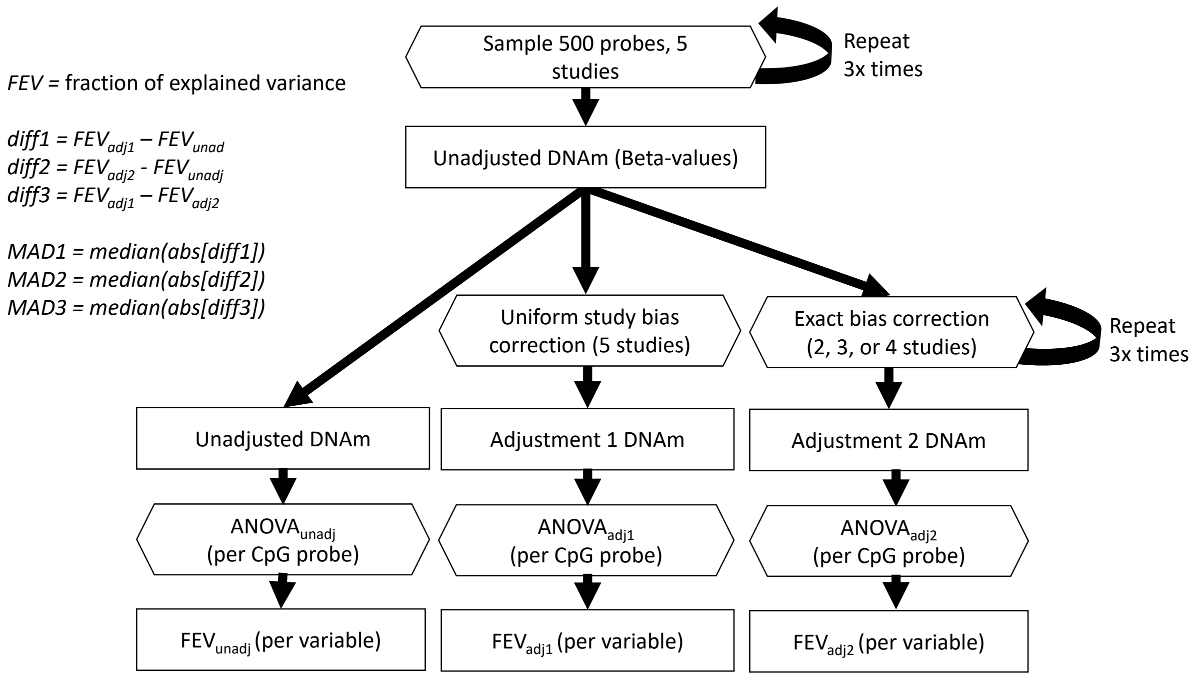
chr	pos	strand	Name	Type	Islands_Name	Relation_to_Island	UCSC_RefGene_Name
chr1	151319512	-	cg00003202	I	chr1:151319326-151319545	Island	RFX5;RFX5
chr1	109234919	-	cg00031456	I	chr1:109234788-109235161	Island	PRPF38B
chr1	228290868	+	cg00105470	I	chr1:228289669-228291184	Island	C1orf35
chr1	247242130	+	cg00106446	I	chr1:247241579-247242201	Island	ZNF670
chr1	205196936	+	cg00118468	I	chr1:205196791-205197252	Island	TMCC2
chr1	182584341	+	cg00153856	I	chr1:182584177-182584545	Island	
chr1	193091045	-	cg00252701	I	chr1:193090728-193091224	Island	CDC73
chr1	152487909	-	cg00302521	I	chr1:152487978-152488270	N_Shore	CRCT1
chr1	1840482	+	cg00309462	I	chr1:1839958-1840601	Island	
chr1	1710237	+	cg00328972	I	chr1:1709394-1710582	Island	NADK
chr1	84972303	+	cg00357368	I	chr1:84971344-84972491	Island	GNG5;SPATA1
chr1	226374726	-	cg00396407	I	chr1:226374035-226374760	Island	ACBD3
chr1	113008976	-	cg00468144	I	chr1:113008844-113009154	Island	WNT2B
chr1	147142741	+	cg00543196	I	chr1:147141842-147142918	Island	ACP6
chr1	91966377	-	cg00668525	I	chr1:91966263-91966999	Island	CDC7;CDC7;CDC7
chr1	176176785	+	cg00687095	I	chr1:176175711-176176811	Island	RFWD2;RFWD2
chr1	55230057	+	cg00756215	I	chr1:55229992-55230369	Island	PARS2
chr1	40254741	-	cg00758584	I	chr1:40253683-40255172	Island	BMP8B
chr1	84464645	+	cg00773459	I	chr1:84464223-84465232	Island	TTLL7;TTLL7
chr1	28198968	-	cg00856002	I	chr1:28199031-28199257	N_Shore	C1orf38;C1orf38;C1orf38
chr1	226309722	+	cg00926874	I	chr1:226308959-226310476	Island	
chr1	244211782	-	cg00993651	I	chr1:244211034-244212088	Island	
chr1	84971486	+	cg01000656	I	chr1:84971344-84972491	Island	SPATA1;GNG5
chr1	162039502	+	cg01137537	I	chr1:162039450-162040052	Island	NOS1AP;NOS1AP
chr1	107684059	+	cg01559617	I	chr1:107682889-107684463	Island	NTNG1;NTNG1;NTNG1
chr1	43855507	-	cg01717331	I	chr1:43855375-43855921	Island	MED8;MED8;C1orf84
chr1	145477253	+	cg01780466	I	chr1:145477022-145477479	Island	LIX1L
chr1	29213675	-	cg01932091	I	chr1:29213438-29214300	Island	41;EPB41;EPB41;EPB41;EPB41;EP

Table 6.8: Characteristics of 14,000 DNAm array CpG probes (rows) with recurrent high and tissue-specific variances (2,000 probes per tissue, tissues: adipose, blood, brain, buccal, liver, nasal, and sperm). Column 1 is the tissue group. Column 2 is the probe ID. Columns 3-6 are the Beta-value median, mean, variance, and standard deviations across samples for the tissue. Column 7 is the chromosome. Column 8 is genome coordinate. Column 9 is the DNA strand. Column 10 is the assay type. Column 11 is the CG-island name, where applicable. Column 12 is the CG-island region type. Column 13 is the gene ID(s). Column 14 is the gene accession(s). Column 15 is the gene region group(s). Preview of supplemental table; full table available online.

tissue	cgid	median	mean	var	sd	chr	pos	strand	Type	Islands_Name
adipose	cg00149537	0.753	0.756	0.008	0.087	chr1	9409810	-	I	
adipose	cg01413848	0.862	0.851	0.007	0.081	chr1	3081226	-	I	chr1:3080934-3081292
adipose	cg01772149	0.602	0.619	0.017	0.129	chr1	3038216	-	I	chr1:3038067-3038343
adipose	cg10096873	0.123	0.152	0.007	0.084	chr1	92546470	+	I	chr1:92545819-92546480
adipose	cg10622019	0.732	0.729	0.009	0.092	chr1	4221485	-	I	chr1:4221387-4221672
adipose	cg10703826	0.256	0.265	0.008	0.089	chr1	119532116	+	I	chr1:119531991-119532196
adipose	cg14038482	0.772	0.775	0.008	0.089	chr1	8384484	+	I	chr1:8384387-8384719
adipose	cg15930240	0.575	0.571	0.011	0.105	chr1	153749015	-	I	chr1:153747655-153748698
adipose	cg18273417	0.316	0.307	0.007	0.081	chr1	153518418	-	I	
adipose	cg18533397	0.157	0.179	0.007	0.086	chr1	110186005	-	I	chr1:110185961-110186164
adipose	cg19025786	0.737	0.738	0.012	0.108	chr1	2206662	-	I	
adipose	cg19924619	0.202	0.211	0.006	0.08	chr1	165323692	-	I	chr1:165323486-165323811
adipose	cg20163085	0.728	0.732	0.011	0.103	chr1	10510396	-	I	
adipose	cg23677911	0.729	0.722	0.009	0.093	chr1	230256394	-	I	
adipose	cg23848152	0.766	0.756	0.009	0.094	chr1	145211389	+	I	chr1:145208944-145210075
adipose	cg24434800	0.408	0.402	0.008	0.087	chr1	119542295	-	I	chr1:119543056-119543454
adipose	cg25340966	0.252	0.264	0.01	0.102	chr1	119532195	-	I	chr1:119531991-119532196
adipose	cg25362585	0.625	0.637	0.018	0.135	chr1	3320431	+	I	chr1:3321269-3322310
adipose	cg25407979	0.358	0.365	0.013	0.115	chr1	204256846	+	I	
adipose	cg00035316	0.24	0.254	0.008	0.087	chr2	176993017	+	I	chr2:176992950-176993186
adipose	cg02565132	0.83	0.82	0.007	0.085	chr2	109559473	-	I	chr2:109558965-109559186
adipose	cg04316624	0.47	0.465	0.008	0.092	chr2	177036809	+	I	chr2:177036254-177037213
adipose	cg04688351	0.128	0.146	0.007	0.083	chr2	223154140	-	I	chr2:223155726-223156154
adipose	cg05669418	0.346	0.358	0.011	0.104	chr2	177022962	-	I	chr2:177024501-177025692
adipose	cg05864326	0.586	0.573	0.018	0.136	chr2	177030150	-	I	chr2:177029413-177029941
adipose	cg07873325	0.759	0.765	0.007	0.085	chr2	88355771	+	I	chr2:88354840-88355574
adipose	cg08079908	0.228	0.244	0.007	0.082	chr2	176997277	+	I	chr2:176993479-176995557
adipose	cg08781325	0.067	0.096	0.008	0.088	chr2	26521777	-	I	

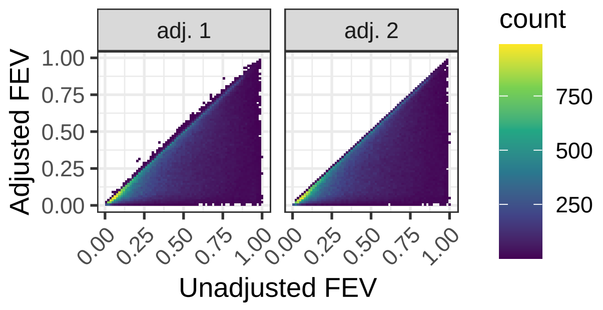
C Supplementary Material for "`recountmethylation` enables flexible analysis of public blood DNA methylation array data."

C.1 Supplementary Figures

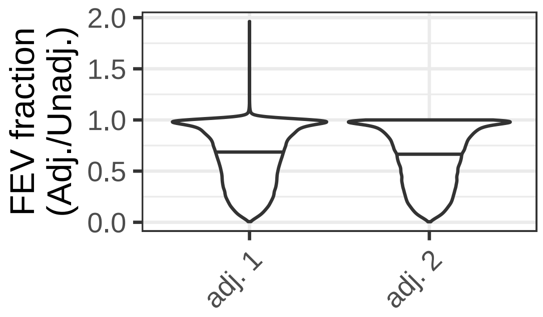


FEV = fraction of explained variance
diff1 = FEV_adj1 – FEV_unad
diff2 = FEV_adj2 - FEV_unadj
diff3 = FEV_adj1 – FEV_adj2
MAD1 = median(abs[diff1])
MAD2 = median(abs[diff2])
MAD3 = median(abs[diff3])
Sample 500 probes, 5 studies
Repeat 3x times
Unadjusted DNAm (Beta-values)
Uniform study bias correction (5 studies)
Exact bias correction (2, 3, or 4 studies)
Repeat 3x times
Unadjusted DNAm
Adjustment 1 DNAm
Adjustment 2 DNAm
ANOVA_unadj (per CpG probe)
ANOVA_adj1 (per CpG probe)
ANOVA_adj2 (per CpG probe)
FEV_unadj (per variable)
FEV_adj1 (per variable)
FEV_adj2 (per variable)

Figure C.1: Workflow diagram to simulate the impact of GSE bias corrections on explained variances. This diagram shows a single simulation rep, including repeated probe and study selections where indicated. From top to bottom, the workflow shows random selection of 500 CpG probes, random selection of 5 studies, and calculation of 3 DNAm datasets per CpG probe: (1) unadjusted DNAm; (2) DNAm after local adjustment on subsets between 1-4 study IDs among 5 selected (a.k.a. adjustment 1); (3) DNAm after uniform adjustment on all 5 selected study IDs (a.k.a. adjustment 2). Finally, ANOVAs are conducted across the 3 DNAm models, and fractions of explained variances are determined from sum of squared variances (3.2.4). Terminology for workflow terms is shown at top left.



(a)



(b)

Figure C.2: Fraction explained variance (FEV) between unadjusted and adjusted DNAm across GSE bias simulations. (a) Density plots of unadjusted FEV on x axes and adjusted FEV on y-axes. Color fill shows density of simulation outcome counts (dark blue = low, green = moderate, yellow = high). (b) Violin plots of FEV fractions, or adjusted FEV over unadjusted FEV, by adjustment type on the x-axis.

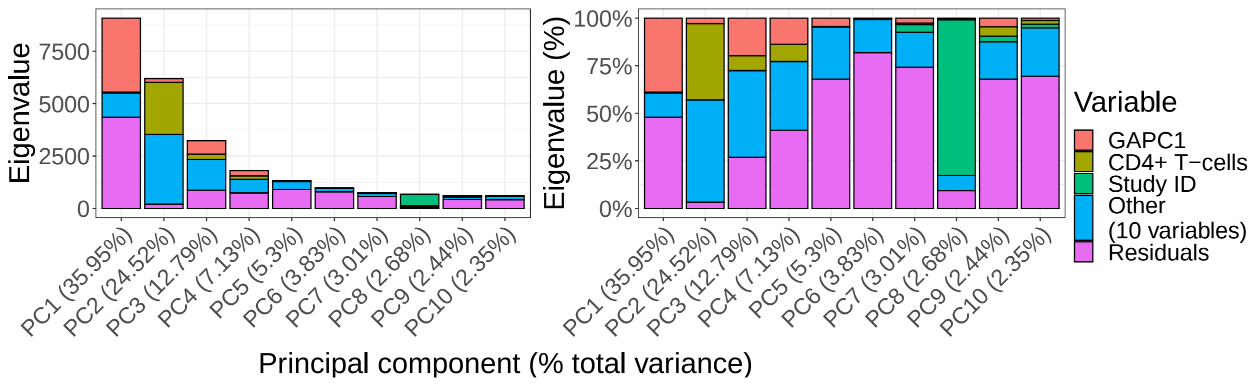


Figure C.3: Autosomal DNAm PCA results across normal blood samples. Eigenvalues explained by select variables. Stacked barplots show the eigenvalue magnitudes (left) and percentages (right) for the top ten components (x-axis). Fill colors indicate magnitudes of component sum of squared variances explained by select variables (red = genetic ancestry PC1, yellow = predicted CD4+ T-cells, green = Study ID, blue = other variable, purple = residuals). The term “other” stores the 10 remaining model variables tested. X-axis labels show the percent of total variances explained by each component in parentheses.

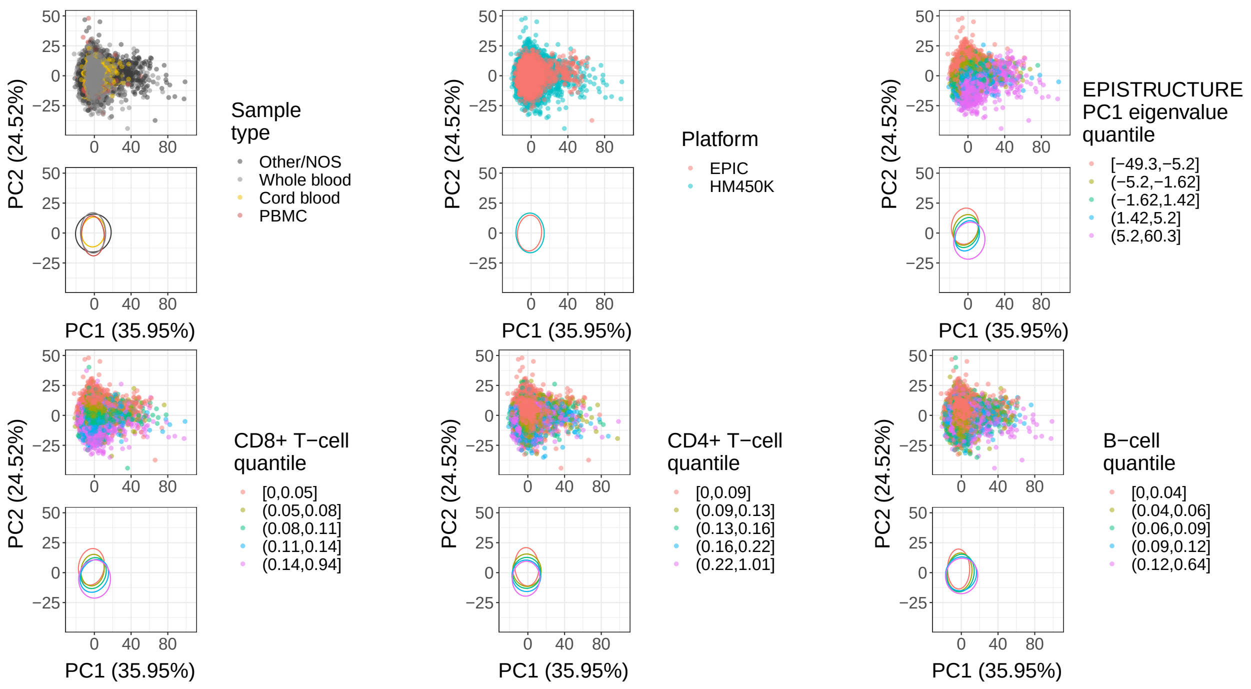
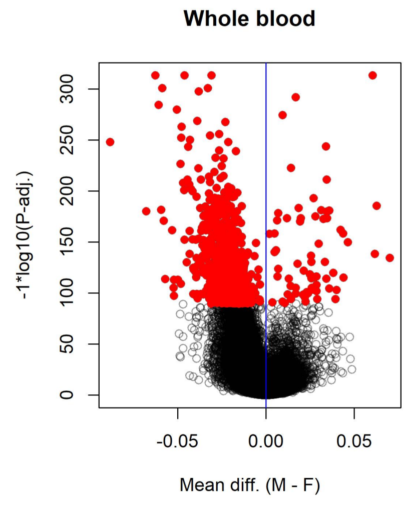
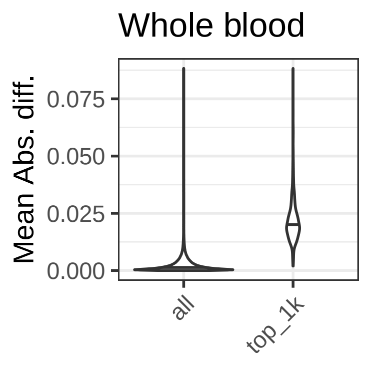


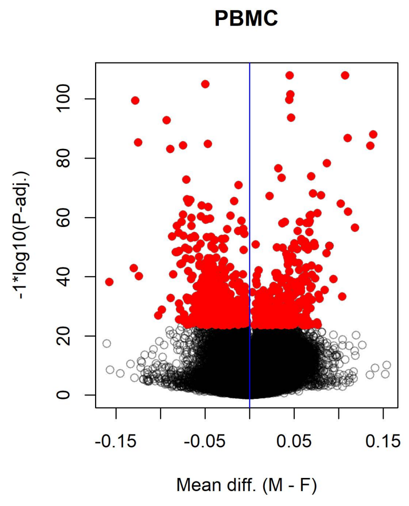
Figure C.4: Top two components from PCAs of autosomal DNAm (noob-normalized Beta-values) colored according to different variables. Each panel includes a scatter plot (top) and 95% confidence interval ellipses (bottom) where the x and y axes correspond to, respectively, the first and second components. Colors specify sample type (top left, black = other/not otherwise specified, gray = whole blood, yellow = cord blood, red = PBMC), platform (top middle, red = EPIC/HM850K, blue = HM450K), the first component of genetic ancestry (top right, [32]), and predicted fractions [31] for CD8+ T-cells (bottom left), CD4+ T-cells (bottom middle), and B-cells (bottom right). Color labels for the latter four continuous variables correspond to sample quintile bins (e.g. 5 quantile ranges: pink = 0-20, yellow = 20-40, green = 40-60, blue = 60-80, purple = 80-100).



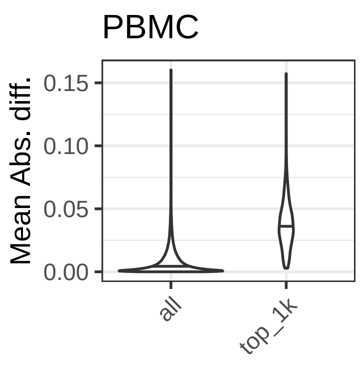
(a)



(b)

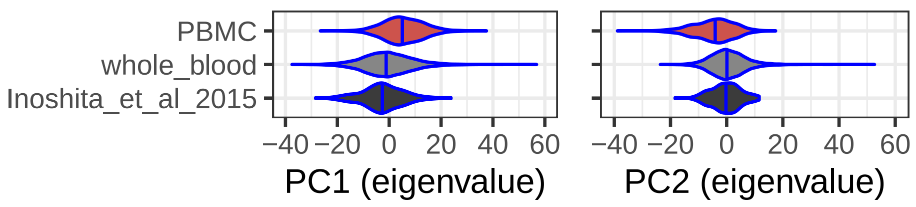


(c)

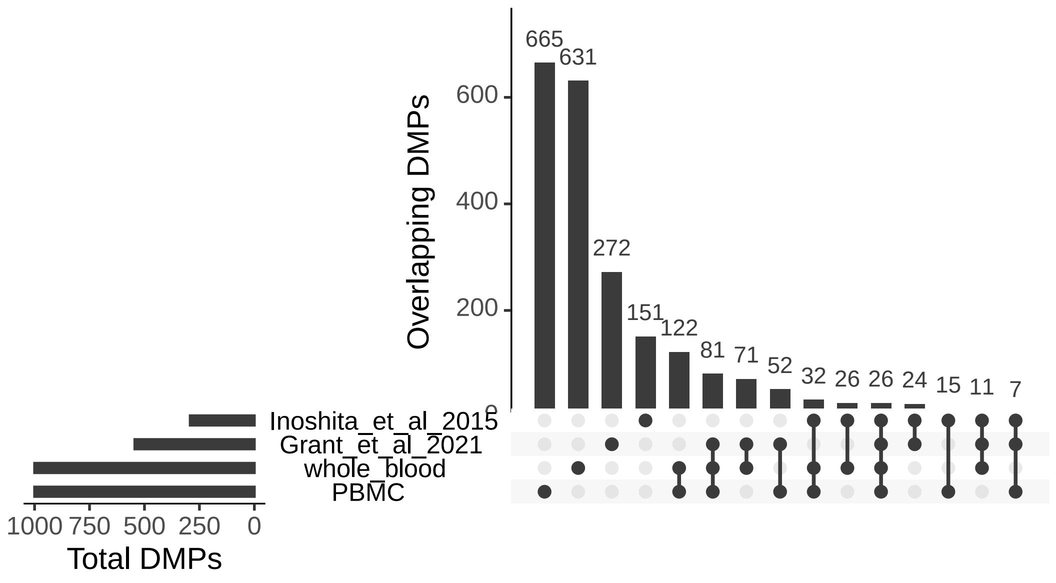


(d)

Figure C.5: Summaries of differential DNAm by sex in whole blood and peripheral blood mononuclear cells (PBMC). (a) Volcano plot for whole blood showing difference in mean Beta-values (males minus females, x-axis) versus probe significance (-1*log10[P-adj.], Benjamini Hotchberg adjustment [167], y-axis). Red dots represent the top 1,000 DMPs, black circles represent non-DMP probes. (b) violin plots for whole blood showing distributions of absolute difference in mean Beta-values (males minus females, y-axis) for all tested CpG probes (left) and only the top 1,000 most significant differentially methylated probes (DMPs, right). Horizontal black lines indicate the distribution medians. (c) Volcano plot for PBMC showing difference in mean Beta-values (males minus females, x-axis) versus probe significance (-1*log10[P-adj.], y-axis). Red dots represent the top 1,000 DMPs, black circles represent non-DMP probes. (d) violin plots for PBMC showing distributions of absolute difference in mean Beta-values (males minus females, y-axis) for all tested CpG probes (left) and only the top 1,000 most significant differentially methylated probes (DMPs, right). Horizontal black lines indicate the distribution medians.



(a)



(b)

Figure C.6: Genetic ancestry differences and intersection of sex DMPs across whole blood and PBMC datasets. (a) Violin plots of the top two genetic ancestry components for the three available datasets (black = "Inoshita et al 2015" [138], red = PBMC compilation, gray = whole blood compilation). (b) Upset plot showing DMP overlaps (lower left and top right barplot magnitudes) among the 4 DMP sets whole blood, PBMC, "Inoshita_et_al_2015" [138], and "Grant_et_al_2021" [236] (lower y-axis labels).

C.2 Supplementary Tables

Table C.1: Blood sample type (rows) availability and demographic variables (columns).

	All platforms		HM450K		EPIC			Age (years old)								
Type	**GSE**	**GSM**	**GSE**	**GSM**	**GSE**	**GSM**	**Fract. M**	**Median**	**SD**	**1$<$**	**1-10**	**10-20**	**20-40**	**40-60**	**60-80**	**$>$80**
all	62	12,242	51	9,083	12	3,159	0.44	30	25	2,155	859	1,640	2,923	2,452	2,053	160
whole blood	31	6,866	25	5,613	6	1,253	0.43	36	21	21	289	1,352	2,252	1,386	1,442	124
cord blood	9	1,475	7	722	2	753	0.52	0	0	1,475	0	0	0	0	0	0
PBMC	9	627	7	382	2	245	0.37	18	19	0	234	87	166	102	35	3

Table C.2: Mined and predicted metadata for uniformly processed blood samples with either raw HM450K or raw EPIC data available. Preview of supplemental table; full table available online.

gsm	gse	gsm_title	platform	blood.subgroup	predsex	predage
GSM1051533	GSE42861	Normal genomic DNA from sample 9	hm450k	other/NOS	M	65.884
GSM1051534	GSE42861	Normal genomic DNA from sample 10	hm450k	other/NOS	M	47.527
GSM1051535	GSE42861	Normal genomic DNA from sample 11	hm450k	other/NOS	M	35.988
GSM1051536	GSE42861	Normal genomic DNA from sample 12	hm450k	other/NOS	M	55.954
GSM1051537	GSE42861	Normal genomic DNA from sample 13	hm450k	other/NOS	M	49.468
GSM1051538	GSE42861	Normal genomic DNA from sample 14	hm450k	other/NOS	M	50.916
GSM1051539	GSE42861	Normal genomic DNA from sample 15	hm450k	other/NOS	M	68.512
GSM1051540	GSE42861	Normal genomic DNA from sample 16	hm450k	other/NOS	M	67.35
GSM1051541	GSE42861	Normal genomic DNA from sample 17	hm450k	other/NOS	M	64.241
GSM1051542	GSE42861	Normal genomic DNA from sample 18	hm450k	other/NOS	M	61.116
GSM1051543	GSE42861	Normal genomic DNA from sample 19	hm450k	other/NOS	M	55.714
GSM1051544	GSE42861	Normal genomic DNA from sample 20	hm450k	other/NOS	M	58.769
GSM1051545	GSE42861	Normal genomic DNA from sample 21	hm450k	other/NOS	M	49.344
GSM1051549	GSE42861	Normal genomic DNA from sample 25	hm450k	other/NOS	F	48.083
GSM1051550	GSE42861	Normal genomic DNA from sample 26	hm450k	other/NOS	F	62.987
GSM1051551	GSE42861	Normal genomic DNA from sample 27	hm450k	other/NOS	F	67.651
GSM1051552	GSE42861	Normal genomic DNA from sample 28	hm450k	other/NOS	F	61.094
GSM1051553	GSE42861	Normal genomic DNA from sample 29	hm450k	other/NOS	F	66.149
GSM1051555	GSE42861	Normal genomic DNA from sample 31	hm450k	other/NOS	F	64.681
GSM1051556	GSE42861	Normal genomic DNA from sample 32	hm450k	other/NOS	F	59.229
GSM1051557	GSE42861	Normal genomic DNA from sample 33	hm450k	other/NOS	F	57.179
GSM1051558	GSE42861	Normal genomic DNA from sample 34	hm450k	other/NOS	F	29.898
GSM1051559	GSE42861	Normal genomic DNA from sample 35	hm450k	other/NOS	F	63.269
GSM1051560	GSE42861	Normal genomic DNA from sample 36	hm450k	other/NOS	F	52.35
GSM1051561	GSE42861	Normal genomic DNA from sample 37	hm450k	other/NOS	F	71.544
GSM1051562	GSE42861	Normal genomic DNA from sample 38	hm450k	other/NOS	F	57.531
GSM1051563	GSE42861	Normal genomic DNA from sample 39	hm450k	other/NOS	F	30.482
GSM1051564	GSE42861	Normal genomic DNA from sample 40	hm450k	other/NOS	F	56.771
GSM1051565	GSE42861	Normal genomic DNA from sample 41	hm450k	other/NOS	F	51.167
GSM1051566	GSE42861	Normal genomic DNA from sample 42	hm450k	other/NOS	F	51.715
GSM1051567	GSE42861	Normal genomic DNA from sample 43	hm450k	other/NOS	F	40.76
GSM1051568	GSE42861	Normal genomic DNA from sample 44	hm450k	other/NOS	F	46.329
GSM1051569	GSE42861	Normal genomic DNA from sample 45	hm450k	other/NOS	F	32.529
GSM1051570	GSE42861	Normal genomic DNA from sample 46	hm450k	other/NOS	F	56.628
GSM1051571	GSE42861	Normal genomic DNA from sample 47	hm450k	other/NOS	F	31.04
GSM1051572	GSE42861	Normal genomic DNA from sample 48	hm450k	other/NOS	F	68.163
GSM1051573	GSE42861	Normal genomic DNA from sample 49	hm450k	other/NOS	F	49.452
GSM1051574	GSE42861	Normal genomic DNA from sample 50	hm450k	other/NOS	F	53.458
GSM1051575	GSE42861	Normal genomic DNA from sample 51	hm450k	other/NOS	F	57.22

Table C.3: Fraction explained variance (FEV) medians by model type (rows) and variables (columns), across study ID bias correction simulations (see 3.2.4).

	Technical		Biological						Demographic			
Model	**Study ID**	**Platform**	**CD8+ T-cells**	**CD4+ T-cells**	**B-cells**	**Gran.**	**Mono.**	**Natural Killer**	**Sex**	**Age**	**G.A. PC1**	**G.A. PC2**
Unadj.	0.465	0.037	0.014	0.010	0.013	0.020	0.005	0.009	0.003	0.012	0.077	0.054
Adj. 1	0.002	0.086	0.036	0.026	0.035	0.050	0.013	0.024	0.008	0.030	0.212	0.144
Adj. 2	0.000	0.089	0.037	0.026	0.035	0.051	0.013	0.024	0.008	0.031	0.214	0.144

Table C.4: Test results and annotations for sex DMPs from [138] and independent compilations of PBMC and whole blood samples. Preview of supplemental table; full table available online.

cgid	is.inoshita.2015.dmp	is.pbmc.dmp	is.whole.blood.dmp	ttest.pvalue.pbmc
cg04946709	TRUE	TRUE	TRUE	2.69E-114
cg26355737	TRUE	TRUE	TRUE	6.13E-114
cg17232883	TRUE	TRUE	TRUE	8.31E-111
cg06759085	FALSE	TRUE	TRUE	3.11E-107
cg08906898	TRUE	TRUE	TRUE	2.90E-105
cg04737881	FALSE	TRUE	TRUE	5.53E-105
cg13180105	TRUE	TRUE	FALSE	3.44E-99
cg17238319	TRUE	TRUE	TRUE	3.37E-98
cg11388673	FALSE	TRUE	TRUE	2.39E-93
cg03894796	FALSE	TRUE	TRUE	4.80E-92
cg11643285	FALSE	TRUE	TRUE	1.38E-90
cg13323902	TRUE	TRUE	TRUE	5.06E-90
cg04858776	TRUE	TRUE	TRUE	1.50E-89
cg12691488	TRUE	TRUE	TRUE	2.36E-89
cg20299935	TRUE	TRUE	TRUE	3.23E-88
cg21148594	TRUE	TRUE	TRUE	2.11E-83
cg16945633	FALSE	TRUE	FALSE	1.22E-81
cg10507304	FALSE	TRUE	TRUE	6.59E-79
cg21216562	FALSE	TRUE	FALSE	2.00E-78
cg15254881	FALSE	TRUE	TRUE	8.74E-78
ch.6.149019574R	FALSE	TRUE	FALSE	6.49E-76
cg03618918	TRUE	TRUE	TRUE	4.90E-73
cg05132077	FALSE	TRUE	TRUE	2.31E-72
cg17843887	FALSE	TRUE	FALSE	3.98E-72
cg26052357	FALSE	TRUE	TRUE	4.81E-71
cg13346869	FALSE	TRUE	FALSE	9.06E-71
cg08034535	TRUE	TRUE	TRUE	2.28E-70
cg09066361	TRUE	TRUE	TRUE	7.58E-70
cg02530860	FALSE	TRUE	TRUE	1.80E-69
cg20031364	FALSE	TRUE	TRUE	6.66E-69
cg07852945	TRUE	TRUE	TRUE	2.30E-68
cg14825413	FALSE	TRUE	TRUE	8.40E-67
cg25910261	TRUE	TRUE	TRUE	2.73E-66
cg10563109	TRUE	TRUE	TRUE	1.01E-65
cg18795569	FALSE	TRUE	TRUE	1.18E-65
cg18575221	FALSE	TRUE	FALSE	2.19E-65
cg22970003	TRUE	TRUE	TRUE	2.24E-65
cg27645294	TRUE	TRUE	TRUE	4.39E-65
cg11574745	TRUE	TRUE	TRUE	1.59E-64

D Supplementary Material for "Retained introns in long RNA-seq reads are not reliably detected in sample-matched short reads"

D.1 Supplementary Figures

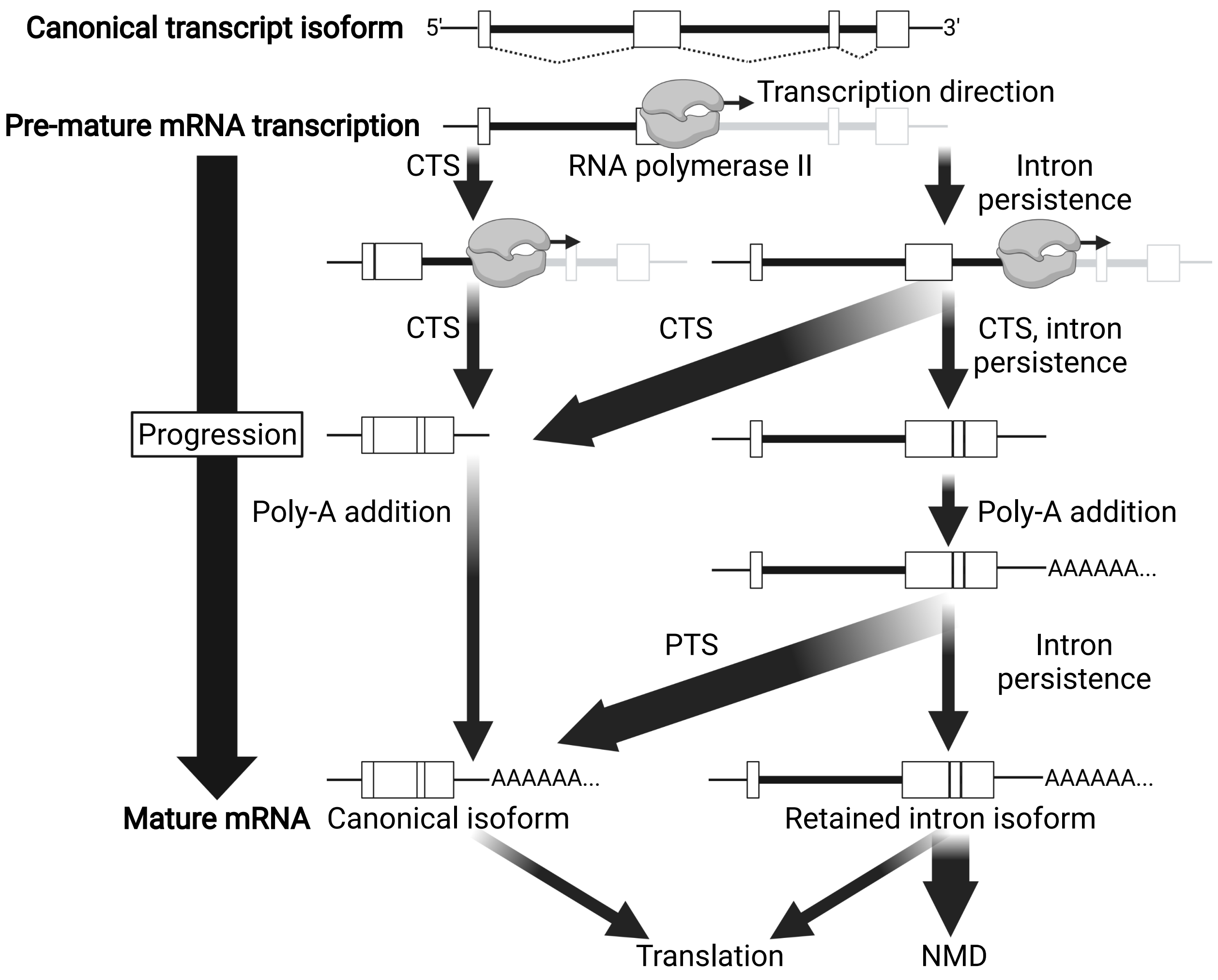


Figure D.1: Progression of transcript diagram, created with BioRender.com. Diagram depicts successive steps in transcript processing which progress from top to bottom. At the top is shown the presumed canonical transcript isoform with its expected splice pattern, followed by pre-mRNA processing steps, which branch between transcription by RNA polymerase II, Co-Transcriptional Splicing (CTS), intron persistence, and poly(A) addition. At the bottom are the possible mature mRNA endpoints, including results from Post-Transcriptional Splicing (PTS) and processing, which include translation and Nonsense-Mediated Decay (NMD). Arrows are labeled with the events they represent, where arrow width sizes indicate their expected event frequencies.

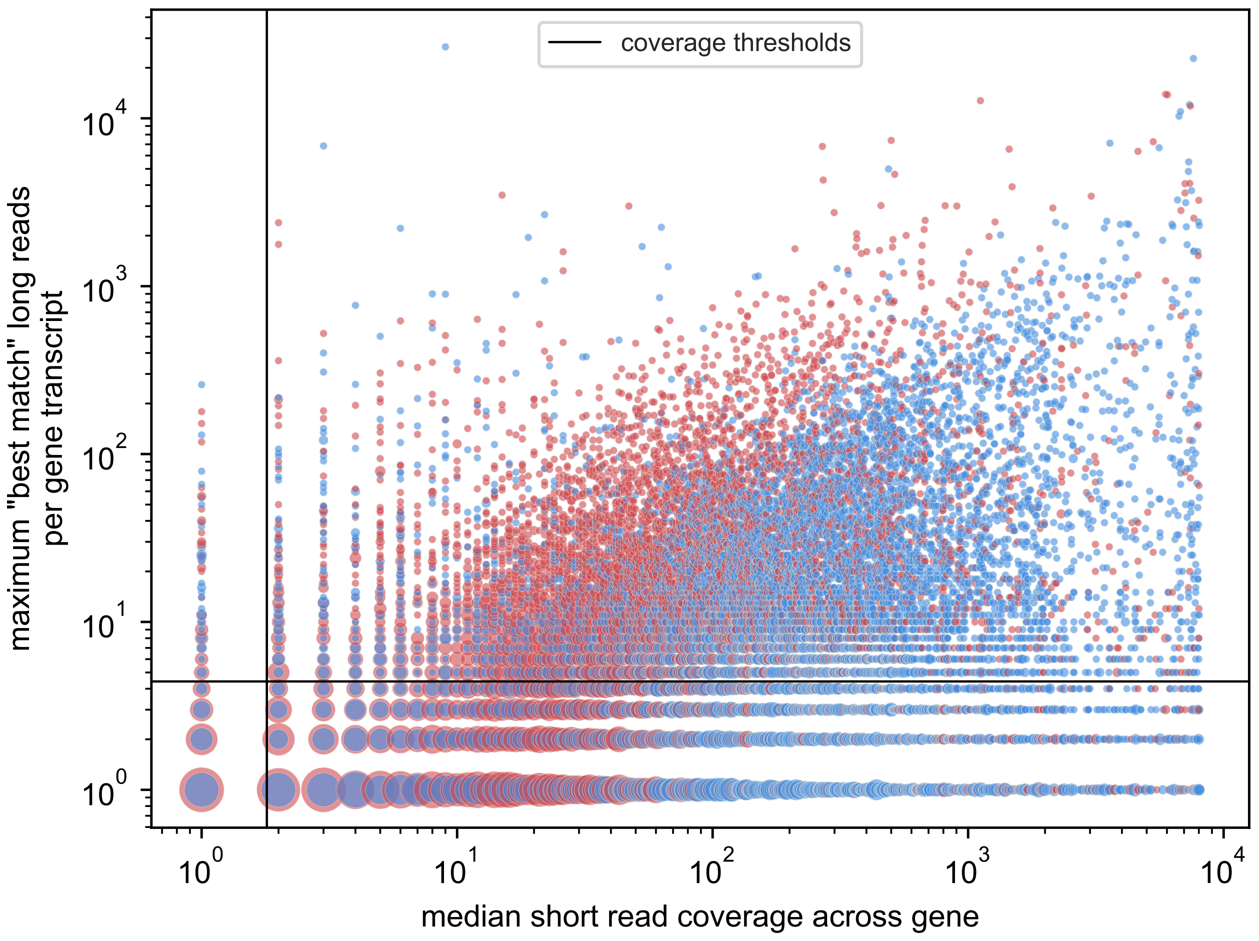


Figure D.2: Short-read and long-read coverage of genes by sample. The maximum number of long reads assigned to one transcript of each gene (y-axis) vs. the median short-read coverage per base across the entire gene (x-axis) for HX1 (red) and iPSC (blue) samples, in log scale. The vertical line represents the minimum median short-read coverage (2) and the horizontal line represents the minimum total long-read coverage per transcript (5) required for a gene to be included in our analysis; genes considered are in the upper right region of the plot.

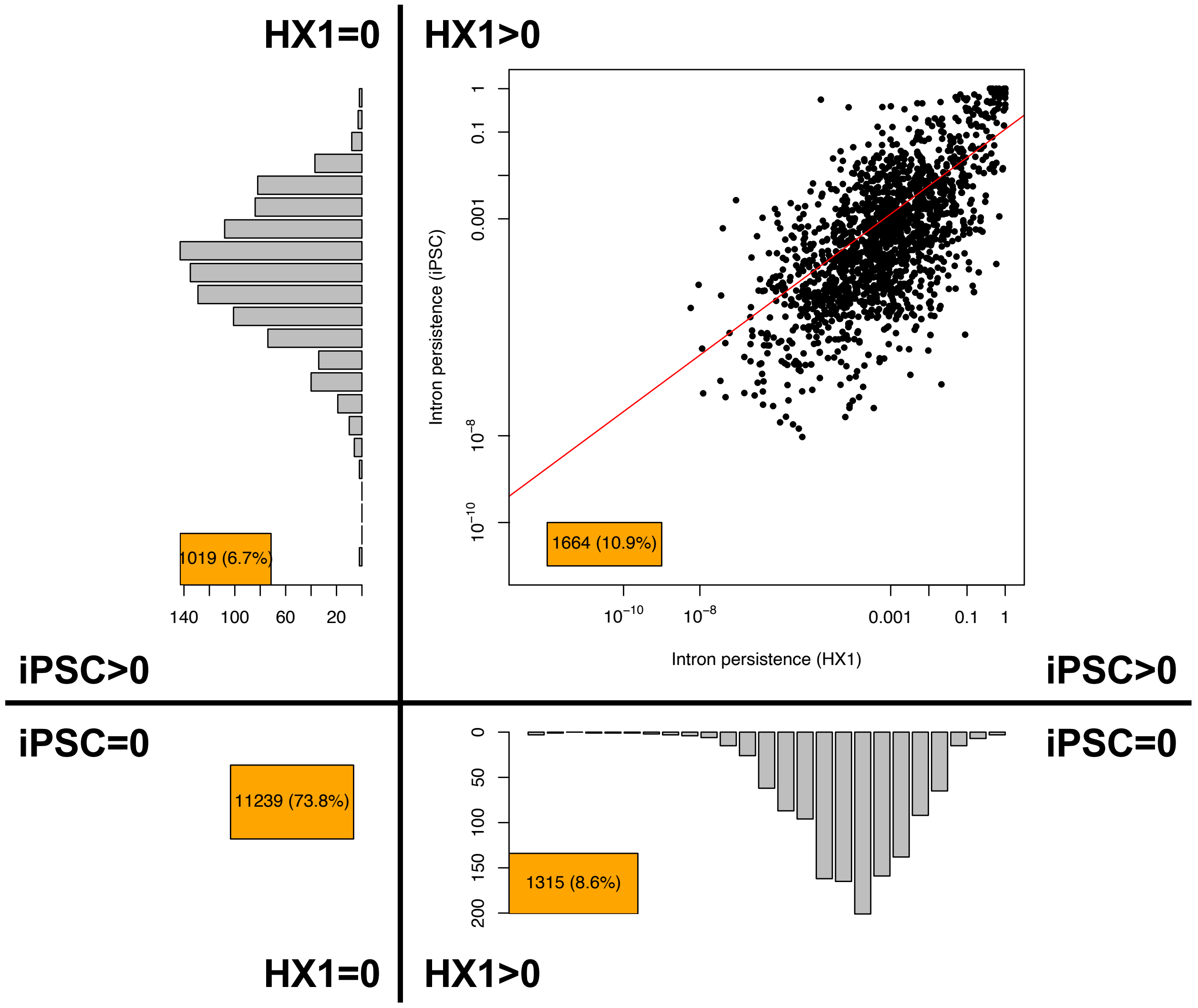


Figure D.3: Distribution of intron persistence values for introns in HX1 and iPSC samples. For introns included in both sample studies, bottom left quadrant represents introns with no persistence across both samples (73.8%), upper left represents introns with persistence in iPSC but not HX1 (6.7%), bottom right represents introns with persistence in HX1 but not iPSC (8.6%), and upper right is a scatterplot of persistences in iPSC (y-axis) vs. HX1 (x-axis) for introns with persistence in both (10.9%).

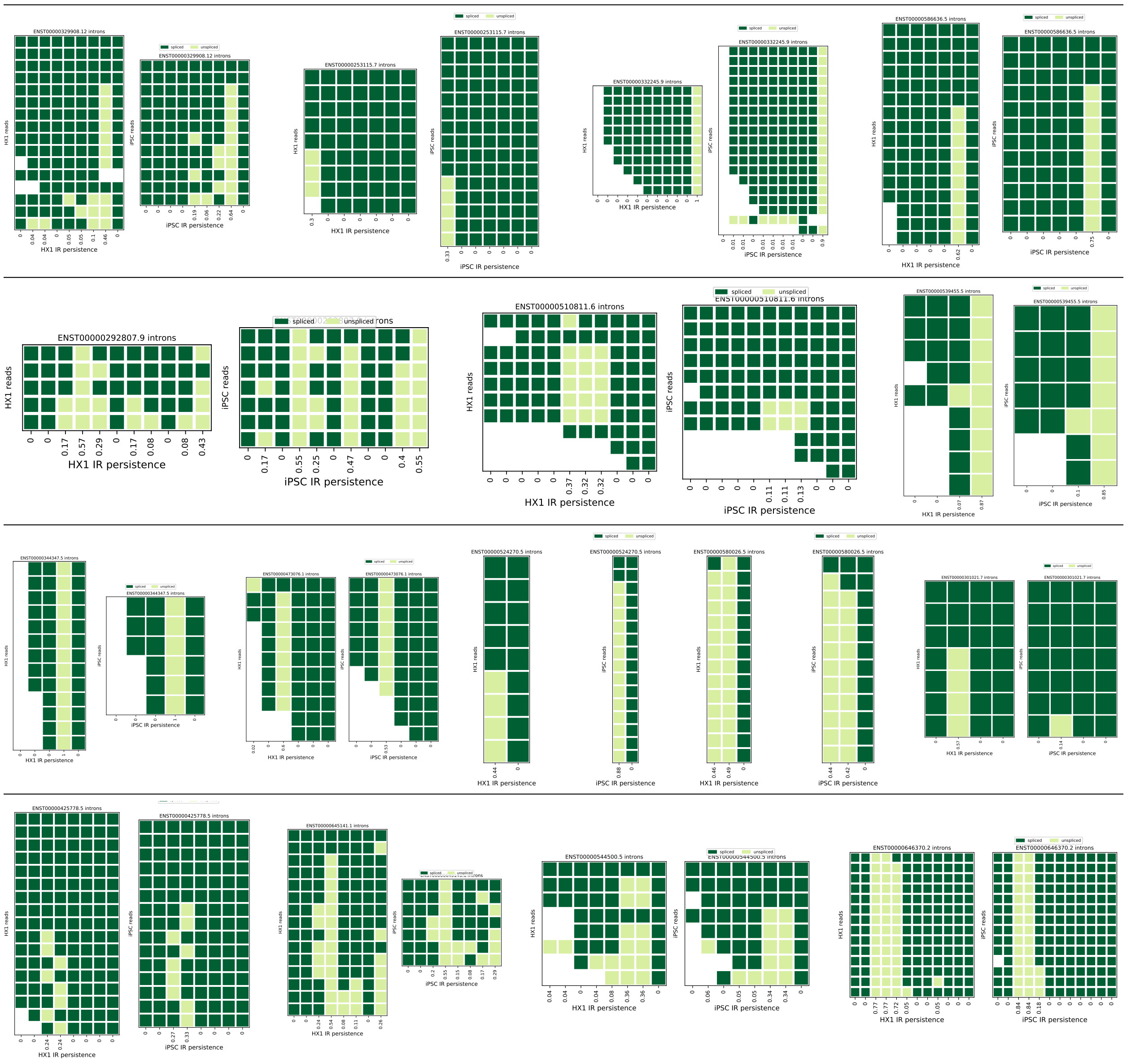
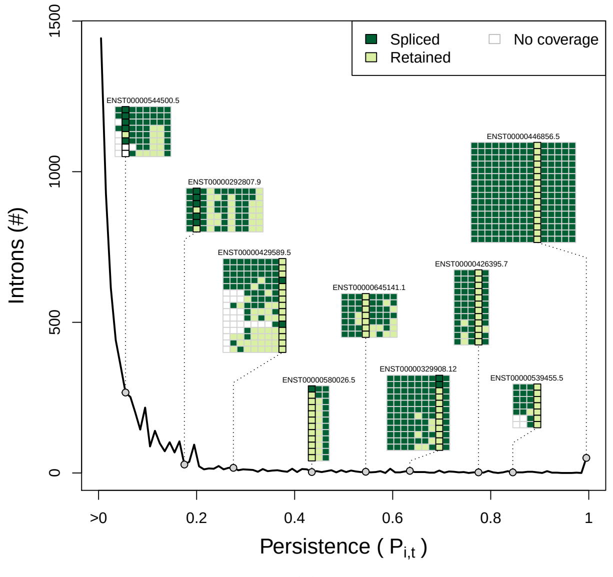
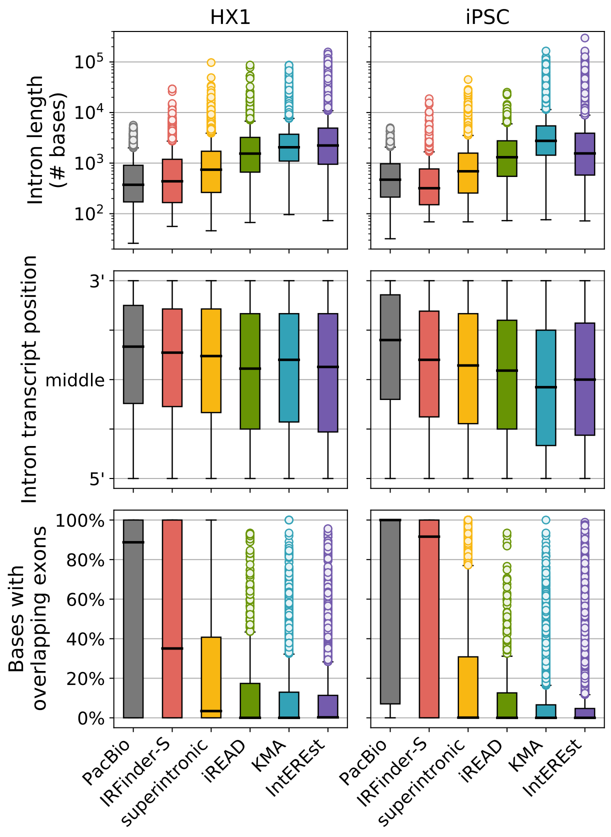


Figure D.4: Splicing similarity between samples. Heatmaps of splicing patterns for a selection of matched transcripts between HX1 and iPSC. Each subplot shows data for one transcript in HX1 (left) and iPSC (right) with rows representing transcript-matched long reads and columns representing transcript introns in 5’ → 3’ order. Transcripts were selected from a subset with 5–20 matched long reads in each sample and 5–20 introns. Dark green indicates a spliced out intron in a given read, light green indicates a retained intron, and white indicates no coverage of the intron in the read.

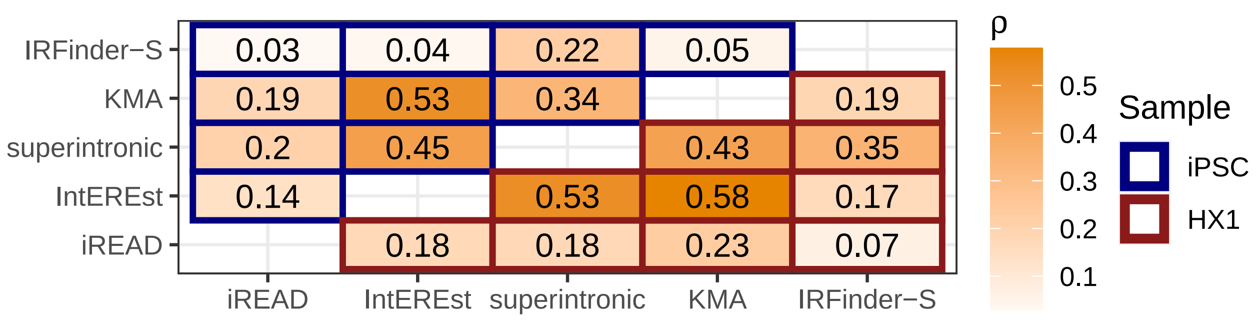


(a)

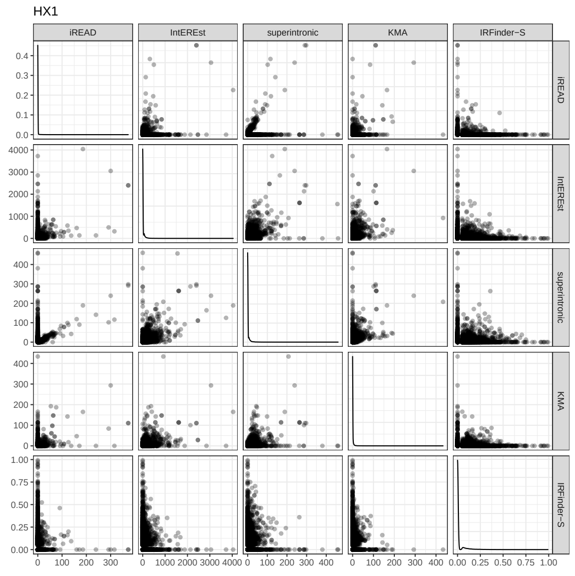


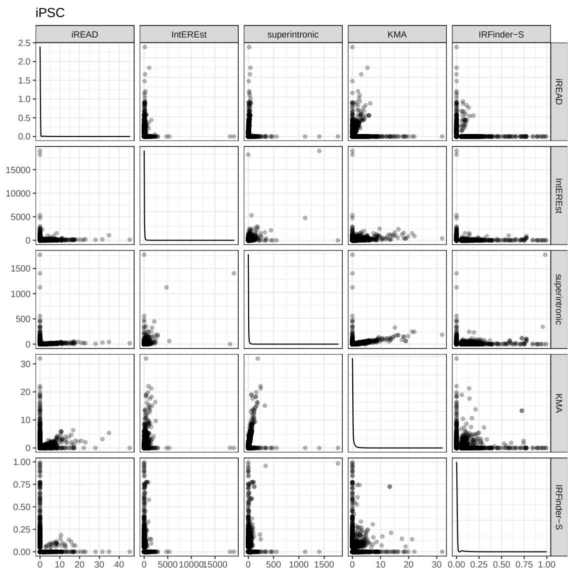
(b)

Figure D.5: (a) Distribution of persistence $P_{i,t}$ and representative transcript examples for iPSC. The number of introns (y-axis) having a given persistence value (x-axis) is shown as a dark black line; note that a large number of introns with $P_i = 0$ are omitted from this analysis. Along the line, gray circles indicate the P_i value corresponding to each of nine introns from representative transcript examples (each transcript is labeled by Ensembl ID, e.g., ENST00000446856.5). Read-level data is shown for each transcript as a colored matrix, where each row is a single long read assigned to the transcript and each column represents a given intron, and color indicates whether an intron is retained (light green), spliced out (dark green), or lacking sequence coverage (white) in a given read. (b) Distributions of properties of persistent and called RIs. Each panel contains a series of boxplots depicting the distribution of intron length (top, log-scale), relative position in transcript (middle), and % of intron bases with overlapping annotated exons (bottom) for HX1 (left) and iPSC (right). The distribution of each of these features is shown for long-read persistent introns ("PacBio", gray) and RIs called by each of the five short read tools: IRFinder-S (red), superintronic (yellow), iREAD (green), KMA (blue), IntEREst (purple).



(a)





(b)

Figure D.6: (a) Pairwise correlations among the intron expression values output by five short-read tools. Each element in this heatmap depicts the correlation in intron expression values (Spearman's test) between the indicated pair of short-read tools, as labeled along the x and y axes. Cell text indicates Spearman ρ coefficient, with corresponding color value obtained by the color gradient scale shown (from white to orange). Cell outline color indicates the sample for which inter-tool correlation was assessed (iPSC [top left] and HX1 [bottom right] are outlined in blue and red, respectively). (b) Intron expression scatter plots between all short-read RI-detection tool pairs (lower and upper triangles of plot grid) and density plots for each of the five individual tools (diagonal plot grid) for HX1. (c) Intron expression scatter plots between all short-read RI-detection tool pairs (lower and upper triangles of plot grid) and density plots for each of the five individual tools (diagonal plot grid) for iPSC.

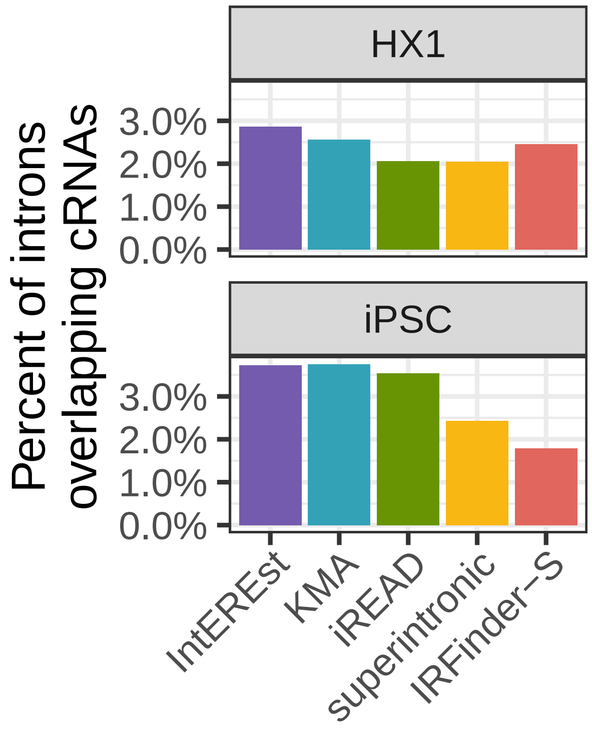


Figure D.7: cRNA overlap among called RIs. Barplots quantify the percent of introns overlapping cRNAs (y-axes) across RIs called from five short-read tools (x-axes, bar color fills, red = IRFinder-S, yellow = superintronic, green = iREAD, purple = IntEREst, blue = KMA).

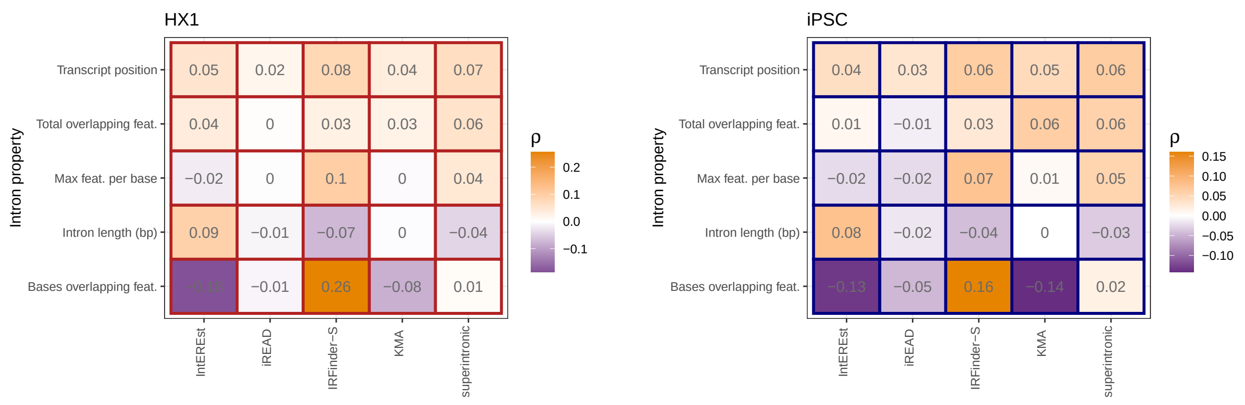


Figure D.8: Correlation of intron expression and continuous properties. Heatmap color fills and text show the Spearman ρ (purple = negative, white = near zero, orange = positive) between intron expression at five short-read tools (x-axes/columns) and five continuous intron properties (y-axes/rows), for samples HX1 (left heatmap) and iPSC (right heatmap).

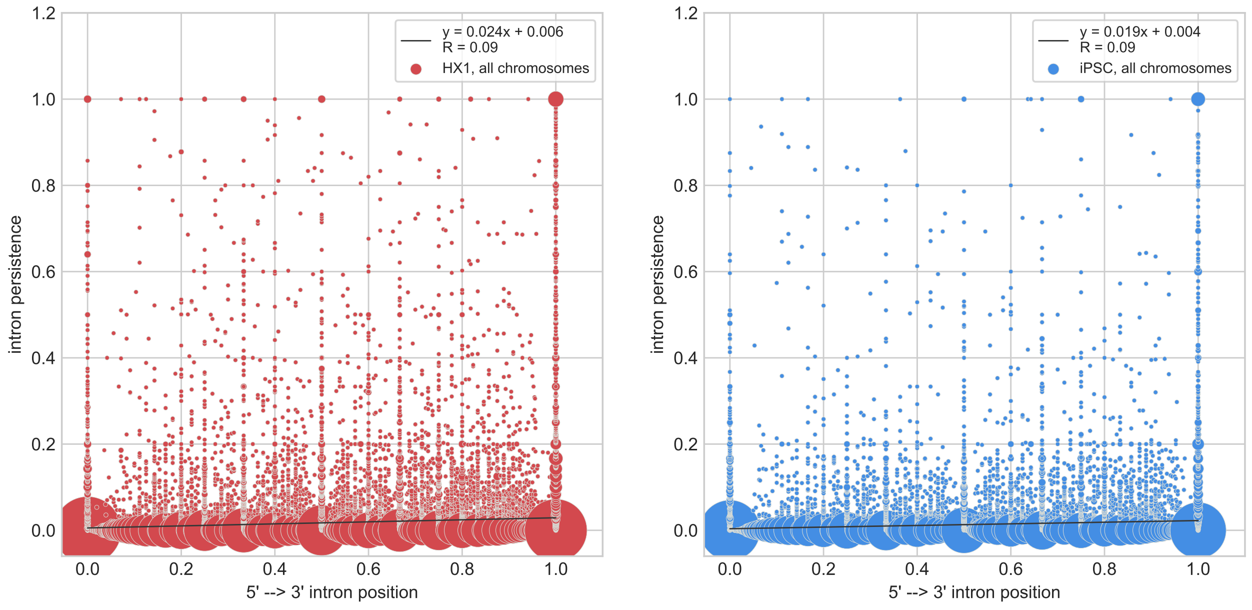


Figure D.9: Association of persistence with transcript position. Scatterplots of intron persistence vs. position within a transcript for HX1 (left, red), iPSC (right, blue). Each point represents one or more introns, with point size representing the number of points at each coordinate. Intron position is an intron-count normalized fraction where 0 represents the transcript's 5' end and 1 represents the 3' end. Plotted lines show the linear fit with equations show in the inset legends.

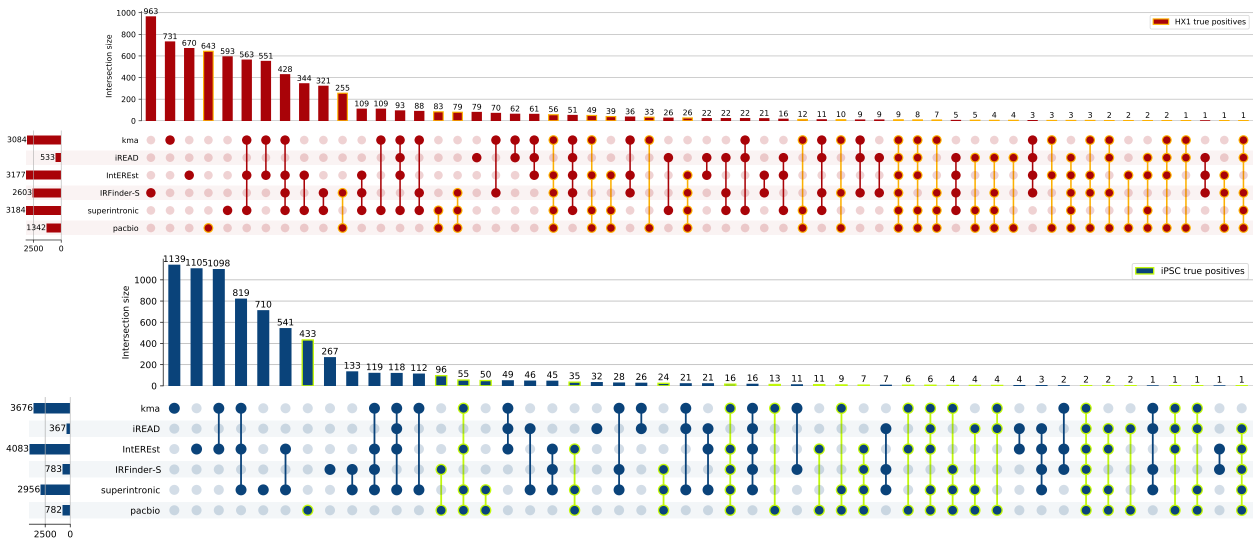


Figure D.10: Set overlaps of persistent introns and called RIs. Upset plots showing overlaps of sets of short-read called RIs and long read persistent introns for iPSC (above, blue) and HX1 (below, red). Sets of true positive persistent introns are highlighted in green for iPSC (above) and orange for HX1 (below).

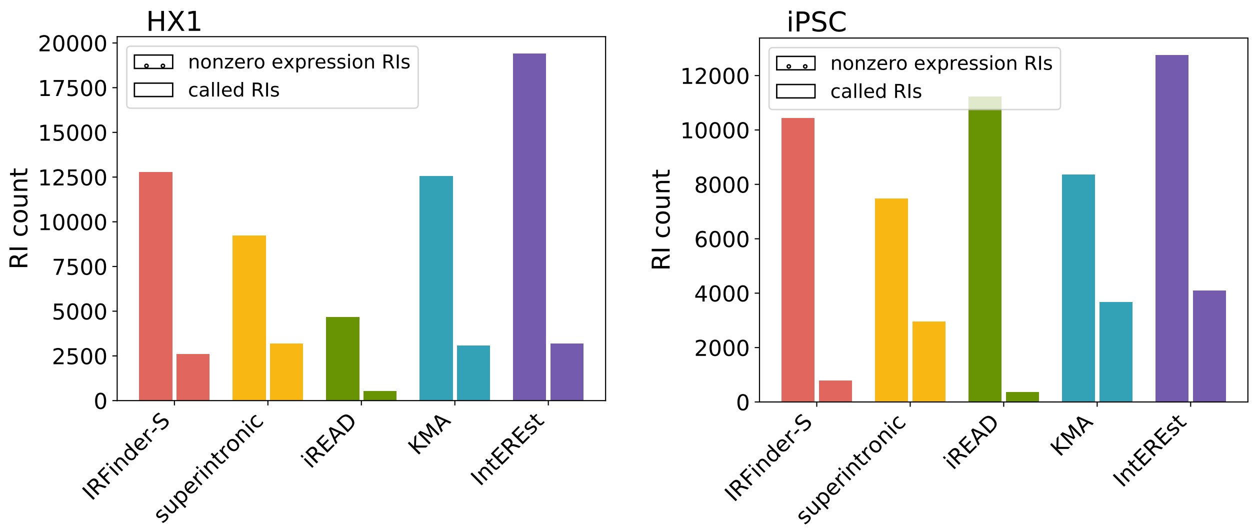


Figure D.11: Potential vs. called RI sets. For HX1 (left) and iPSC (right), counts of all potential (calculated nonzero expression) RIs (dotted hatch, left for each tool) and called (filtered) RIs (no hatch, right for each tool) for each short-read detection tool (red = IRFinder-S, yellow = superintronic, green = iREAD, purple = IntEREst, blue = KMA).

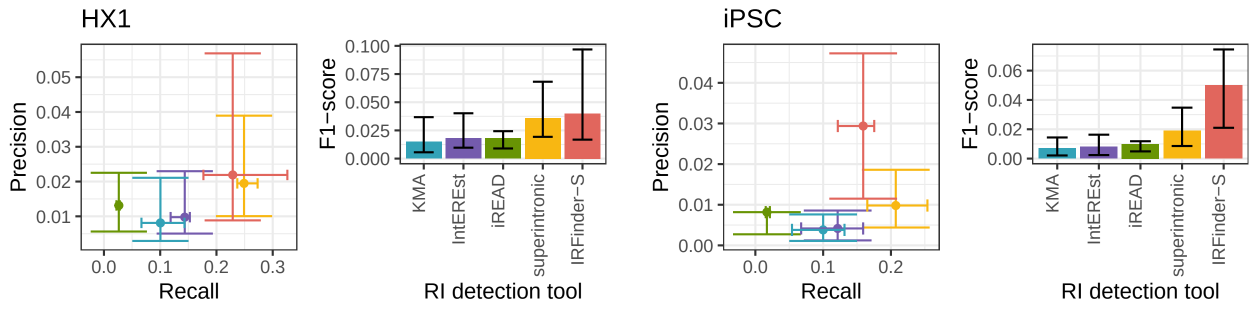


Figure D.12: Performance summaries across persistence cutoffs. Scatter plot y-axes show precision and x-axes show recall, and barplot y-axes show F1-scores, for samples HX1 (left plots) and iPSC (right plots). Colors indicate short-read RI detection tools (red = IRFinder-S, yellow = superintronic, green = iREAD, purple = IntEREst, blue = KMA). Centroids and whiskers indicate the measure medians and interquartile ranges across persistence cutoffs varied from 0.1 to 0.9 at 0.1 intervals (4.2).

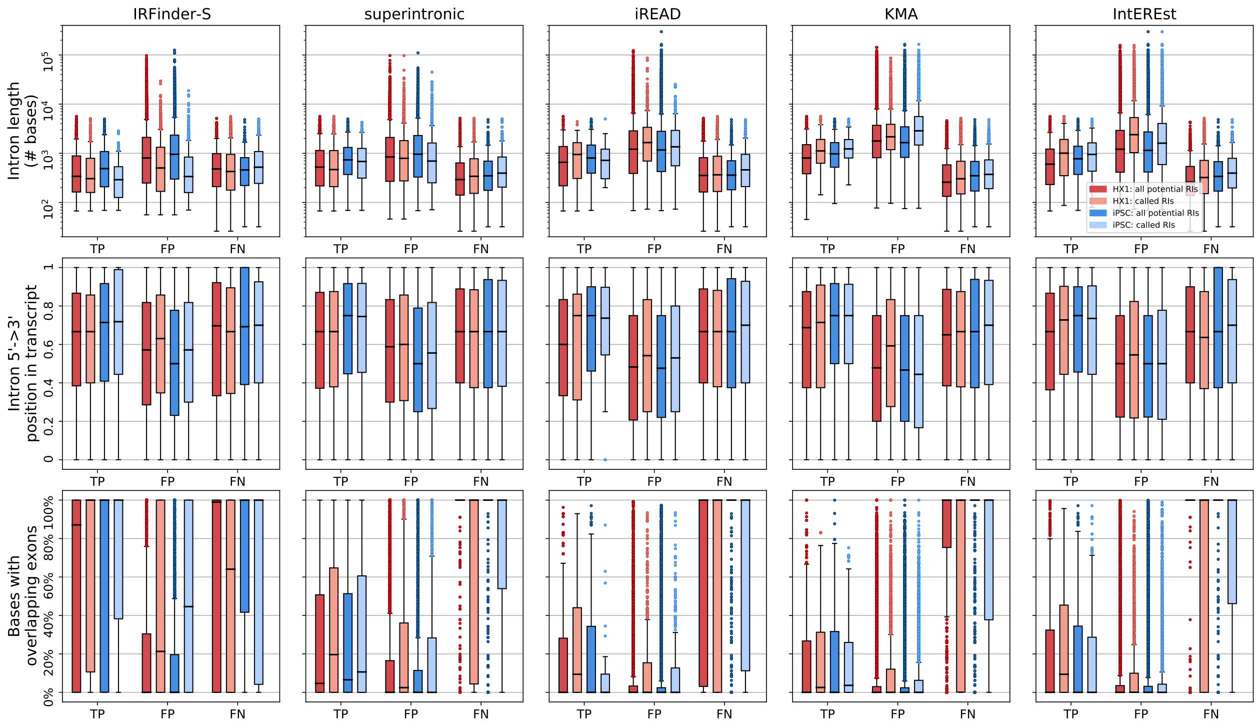


Figure D.13: Three target intron properties, length (top), position along the direction of transcription (0 = 5', 1 = 3') and % of bases with an overlapping annotated exon vs. TP, FP, and FN calls for HX1 and iPSC via potential RIs (darker) and called RIs (lighter), at long read persistence of 0.1 for 5 short read tools.

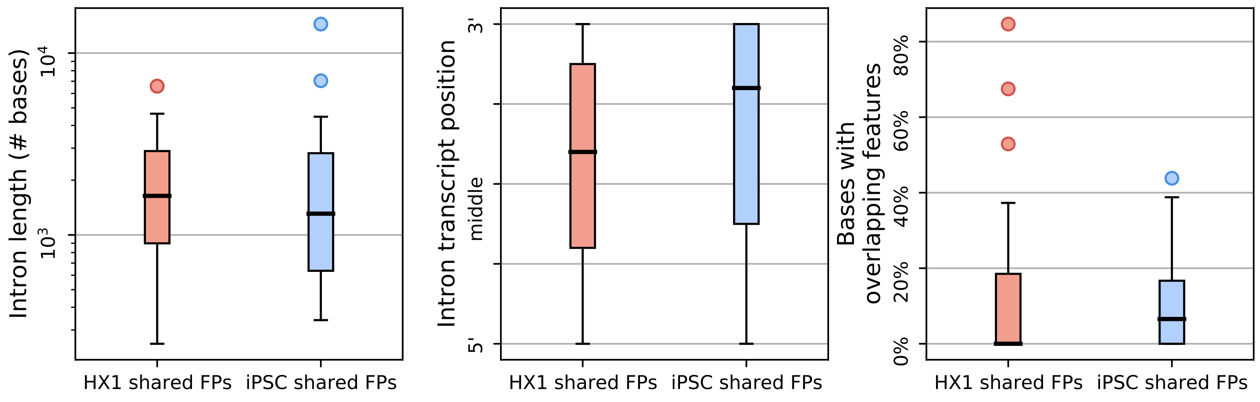


Figure D.14: Distribution of intron properties for shared false positive introns (FPs) called across all five short-read detection tools. Properties are, left to right, intron length in # of bases (log scale), transcript position, and % of bases with overlapping exons, for, in each panel, HX1 (left, red) and iPSC (right, blue).

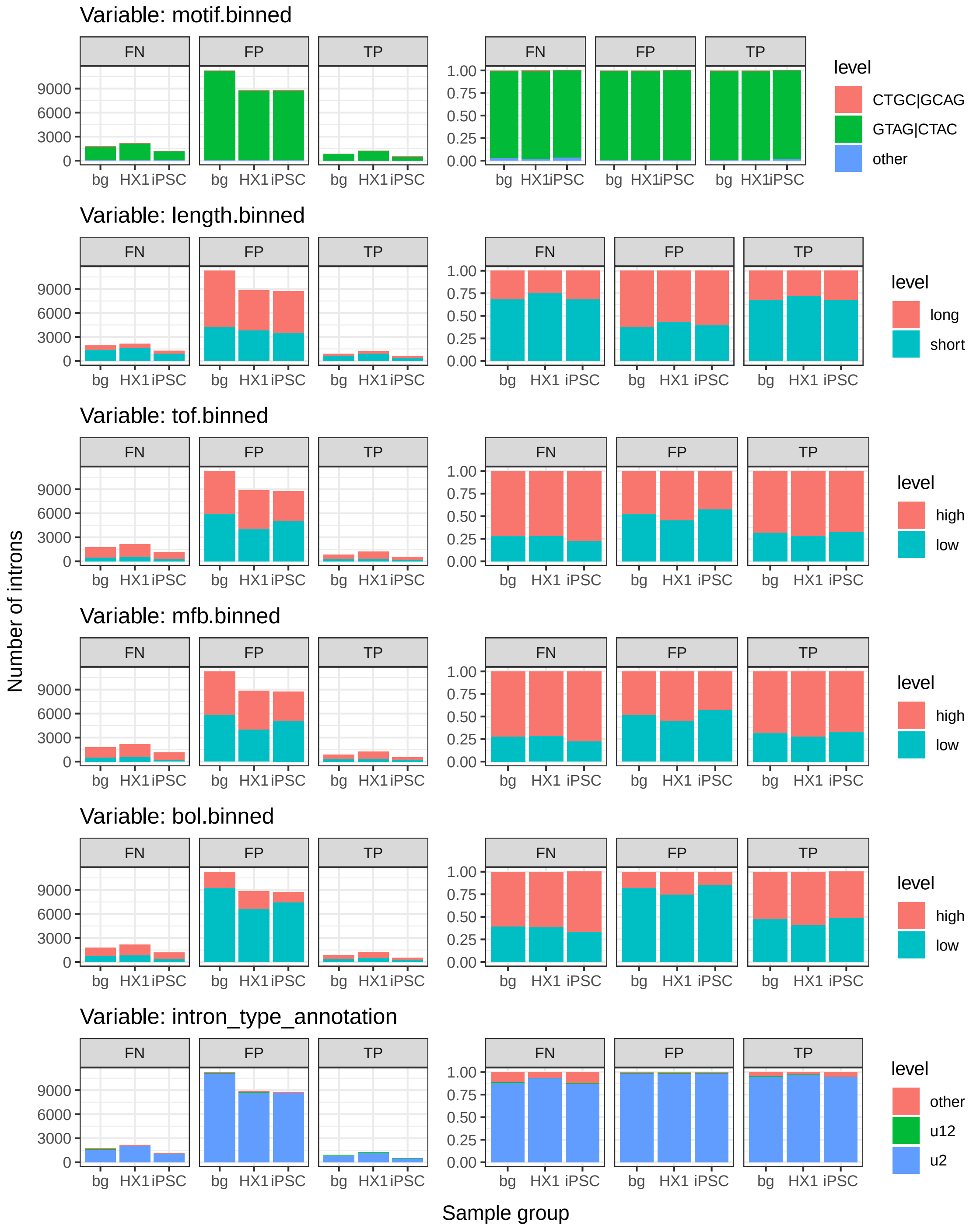


Figure D.15: Distributions of binned intron properties. Barplots of intron counts (left column) and percentages (right column) across unique levels (fill colors indicated in legends) for binned intron properties (plot titles). Results were binned by sample group types (columns, either HX1, iPSC, or the background of all unique introns) and intron 4+ truth metric categories TP, FP, and FN (ribbon labels, e.g. intron was TP in at least 4 tools for iPSC, etc.). Qualitative properties were binned by the top three most frequent levels (e.g. "intron type annotation" and "motif binned"), and quantitative properties were binned using the 50th quantile cutoff (e.g. "length", total overlapping features or "tof," max features per base or "mfb," and bases overlapped or "bol").

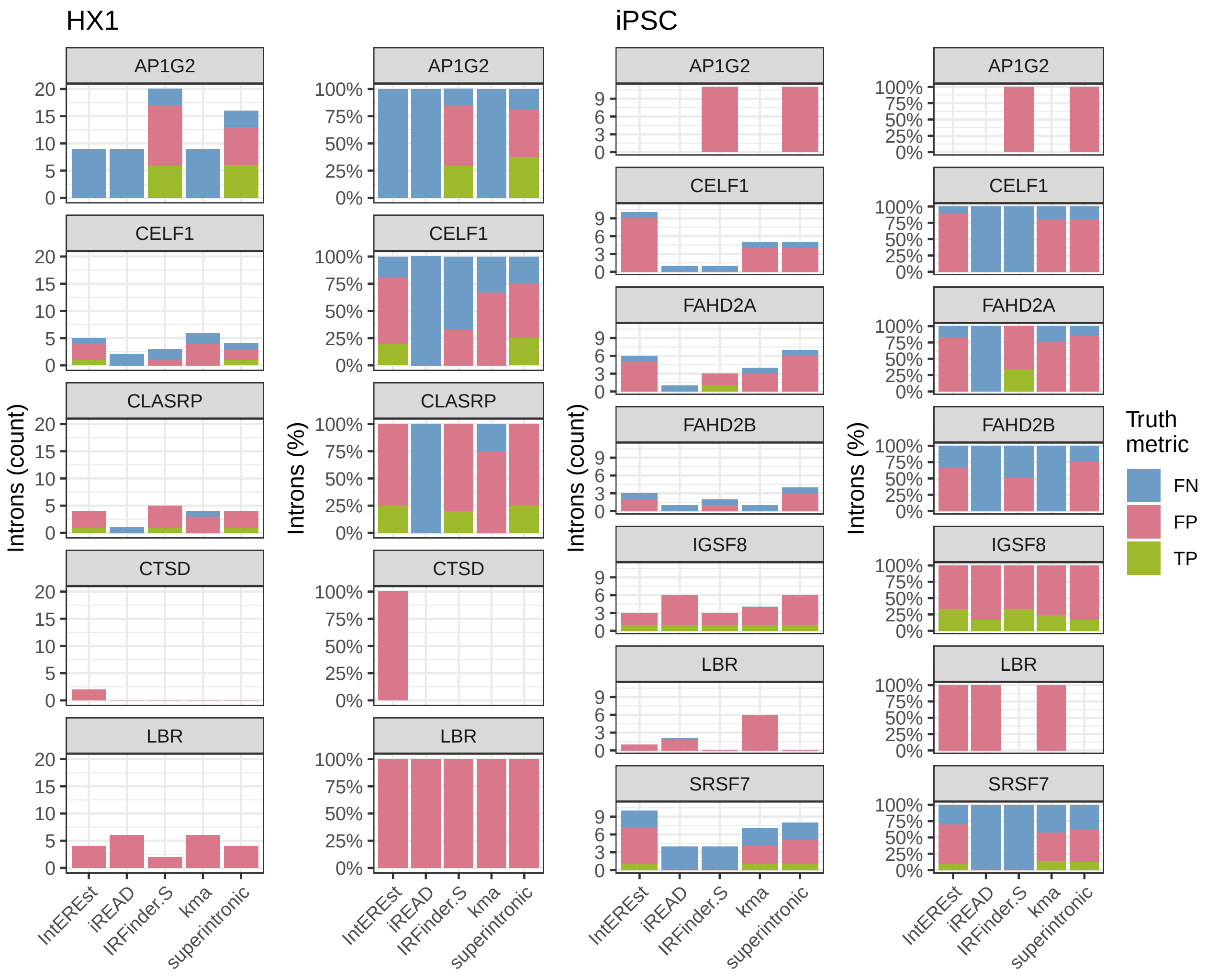


Figure D.16: Intron abundance by truth category across genes with validated RIs. Barplots show intron counts and percentages (y-axes) grouped by short-read tool (x-axes), gene (titles), for samples HX1 (left two plot columns) and iPSC (right two plot columns). Bar color fills indicate the short-read tool-specific truth category (green = TP, pink = FP, blue = FN).

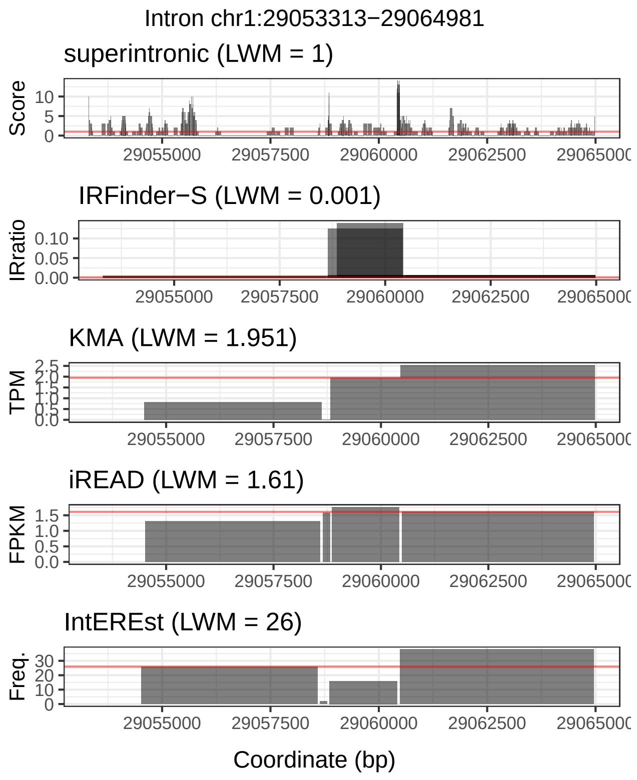
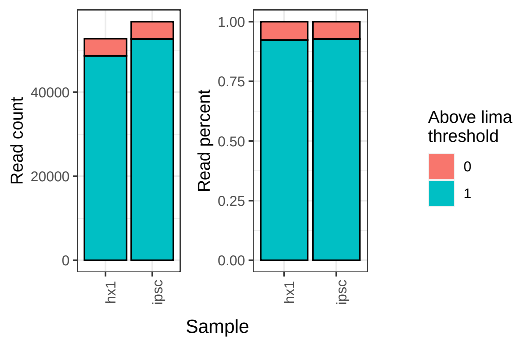
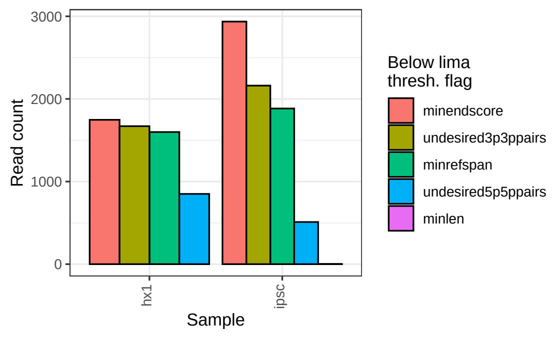


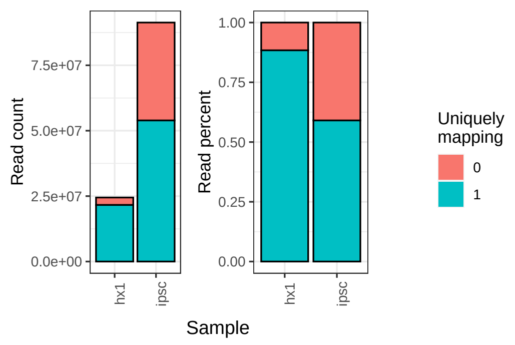
Figure D.17: Example LWM at intron *chr1*:29053313-29064981 (see 4.2.14). Intron expression (y-axes) is shown for genomic coordinates (x-axes), where expressed regions are represented by semi-transparent black rectangles which overlap the target intron. Results are grouped by each of the five short-read tools studied, and the LWM value calculated for each tool is shown in the plot titles and horizontal red lines.



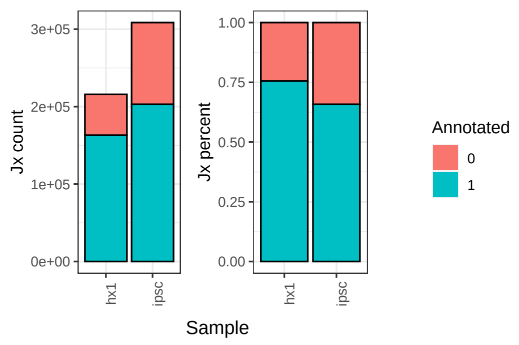
(a)



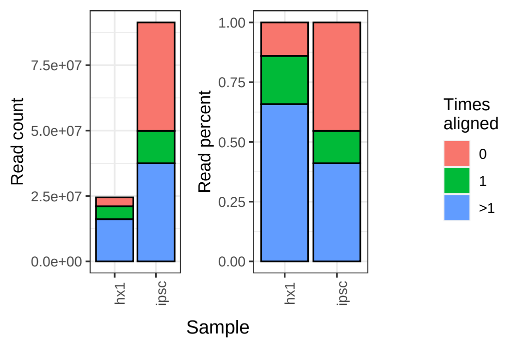
(b)



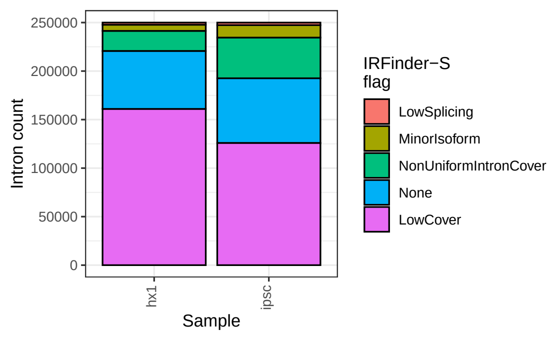
(c)



(d)



(e)



(f)

Figure D.18: Processing and alignment quality control. Results (y-axes, fill colors) across long-read (A-B) and short-read (C-F) data runs for the samples HX1 and iPSC (x-axes) as follows: (a) `lima` quality among long-reads. Barplot y-axes quantify long-read counts (left) and percentages (right) relative to the quality threshold (blue = above, pink = below), where medians across all runs are shown for iPSC. (b) `lima` flags among long-reads. Barplot y-axes quantify long-reads, where bar colors and x-axes indicate one of the five quality flags (magenta = below minimum length or "minlen", blue = undesired 5-prime 5-prime pairs as "undesired5p5ppairs", green = below reference span as "minrefspan", yellow = undesired 3-prime 3-prime pairs as "undesired3p3ppairs", and pink = below minimum end score as "minendscore"). (c) Unique mapping among STAR-aligned short-reads. Barplot y-axes quantify short-read counts (left) and percentages (right) by mappability (blue/1 = uniquely mapping, pink/0 = not uniquely mapping). (d) Annotation among STAR-aligned short-reads. Barplot y-axes as in (c) with color indicating annotation (blue/1 = annotated, pink/0 = not annotated). (e) Alignment counts among bowtie2-aligned short-reads. Barplot y-axes as in (c), where bar colors show alignment counts (blue = $>$ 1 times, green = 1 time, pink = 0 times). (f) IRFinder-S flag quantities. Barplot y-axes as in (c), where bar colors show flag (magenta = low coverage as "LowCover", blue = none, green = non-uniform intron coverage as "NonUniformIntronCover", yellow = minor isoform presence as "MinorIsoform", pink = low splicing as "LowSplicing").

D.2 Supplementary Tables

Table D.1: Description of sequencing data used in this chapter.

Sample name in paper	iPSC	HX1
Sample type	Induced pluripotent stem cell line	Whole blood (non-cancer)
Biosample ID	SAMN07611993	SAMN04251426
SRA Study ID	SRP098984	SRP065930
Long read platform	PacBio Iso-Seq RSII	PacBio Iso-Seq RSII
Size fractionated	No	Yes
Iso-Seq runs	27	46
Aligned long reads	839,558	945,180
Short-read platform	Illumina NextSeq 500	Illumina HiSeq 2000
Short read runs	1	1
Aligned short reads (% uniquely aligned)	91,330,785 (59%)	24,463,210 (88%)

Table D.2: Performance metrics for called RIs across persistence thresholds. Green indicates the highest value per threshold and sample for each metric.

sample	short read detection tool (RIs detected)		long read persistence threshold:									
			>0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
HX1		long read RI count	4585	1342	675	458	334	249	181	135	106	72
	iREAD (533)	precision	0.317	0.077	0.032	0.023	0.019	0.013	0.009	0.006	0.002	0.002
		recall	0.037	0.031	0.025	0.026	0.030	0.028	0.028	0.022	0.009	0.014
		f-score	0.066	0.044	0.028	0.024	0.023	0.018	0.014	0.009	0.003	0.003
	IntEREst (3,177)	precision	0.265	0.064	0.037	0.023	0.016	0.010	0.008	0.005	0.003	0.002
		recall	0.184	0.151	0.172	0.162	0.153	0.124	0.144	0.119	0.094	0.083
		f-score	0.217	0.090	0.060	0.041	0.029	0.018	0.015	0.010	0.006	0.004
	superintronic (3,184)	precision	0.444	0.120	0.063	0.040	0.028	0.019	0.014	0.010	0.008	0.004
		recall	0.308	0.285	0.295	0.277	0.269	0.249	0.249	0.237	0.226	0.194
		f-score	0.364	0.169	0.103	0.070	0.051	0.036	0.027	0.019	0.015	0.009
	kma (3,084)	precision	0.258	0.064	0.034	0.021	0.014	0.008	0.006	0.003	0.002	0.001
		recall	0.174	0.146	0.154	0.144	0.126	0.100	0.099	0.067	0.066	0.056
		f-score	0.208	0.089	0.055	0.037	0.025	0.015	0.011	0.006	0.004	0.003
	IRFinder-S (2,603)	precision	0.509	0.175	0.086	0.057	0.036	0.022	0.012	0.009	0.007	0.003
		recall	0.289	0.340	0.330	0.323	0.281	0.229	0.177	0.170	0.179	0.125
		f-score	0.368	0.231	0.136	0.097	0.064	0.040	0.023	0.017	0.014	0.007
iPSC		long read RI count	3194	782	327	212	176	140	109	75	61	40
	iREAD (367)	precision	0.237	0.063	0.014	0.008	0.008	0.008	0.005	0.003	0.003	0.003
		recall	0.027	0.029	0.015	0.014	0.017	0.021	0.018	0.013	0.016	0.025
		f-score	0.049	0.040	0.014	0.010	0.011	0.012	0.008	0.005	0.005	0.005
	IntEREst (4,083)	precision	0.179	0.035	0.013	0.009	0.006	0.004	0.002	0.001	0.001	0.000
		recall	0.229	0.182	0.159	0.165	0.136	0.121	0.083	0.067	0.066	0.025
		f-score	0.201	0.058	0.024	0.016	0.011	0.008	0.004	0.002	0.002	0.000
	superintronic (2,956)	precision	0.344	0.073	0.028	0.019	0.013	0.010	0.006	0.004	0.003	0.002
		recall	0.318	0.276	0.254	0.259	0.222	0.207	0.165	0.173	0.148	0.125
		f-score	0.331	0.116	0.051	0.035	0.025	0.019	0.012	0.009	0.006	0.003
	kma (3,676)	precision	0.148	0.032	0.012	0.008	0.005	0.004	0.002	0.001	0.001	0.000
		recall	0.171	0.150	0.131	0.132	0.102	0.100	0.083	0.053	0.049	0.025
		f-score	0.159	0.052	0.021	0.014	0.009	0.007	0.005	0.002	0.002	0.001
	IRFinder-S (783)	precision	0.542	0.192	0.079	0.047	0.036	0.029	0.019	0.011	0.008	0.004
		recall	0.133	0.192	0.190	0.175	0.159	0.164	0.138	0.120	0.098	0.075
		f-score	0.213	0.192	0.112	0.074	0.058	0.050	0.034	0.021	0.014	0.007

Table D.3: Properties and sources of experimentally validated RIs studied.

Gene	Source	Intron coordinates	Discovery assay	Validation assay	Disease or cell type association	Samples with gene expression	Sample intron persistence
AP1G2	Jeong 2021 [310]	chr14:23565702-23565815 (intron 5)	short-read RNA-seq	RT-PCR	mesenchymal stem cell	HX1, iPSC	0.06, 0
CELF1	Li 2021 [312]	chr11:47478953-47482694; chr11:47478941-47482694	short-read RNA-seq	Nanostring	Alzheimer's disease	HX1, iPSC	0, 0
CLASRP	Li 2021 [312]	chr19:45069249-45070021	short-read RNA-seq	Nanostring	Alzheimer's disease	HX1	0
CTSD	Wong 2013 [272]	chr11:1755029-1757323 (intron 5)	short-read RNA-seq	RT-PCR, RNA-seq	granulocyte	HX1	0
FAHD2A	Li 2021 [312]	chr2:95412765-95412894	short-read RNA-seq	Nanostring	Alzheimer's disease	iPSC	0.06
FAHD2B	Li 2021 [312]	chr2:97083818-97083947	short-read RNA-seq	Nanostring	Alzheimer's disease	iPSC	0.05
IGSF8	Li 2021 [312]	chr1:160094172-160094868	short-read RNA-seq	Nanostring	Alzheimer's disease	iPSC	0.02
LBR	Wong 2013 [272]	chr1:225410417-225411336 (intron 9)	short-read RNA-seq	RT-PCR, RNA-seq	granulocyte	HX1, iPSC	0.02, 0.01
SRSF7	Lejeune 2001 [311]	chr2:38748654-38749528 (intron 3)	*in vitro* splicing assays	Northern blot	–	iPSC	0.17

Table D.4: HX1 called RIs. Preview of supplemental table; full table available online.

longread_persistence_threshold	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
longread_intron_count	4585.00	1342.00	675.00	458.00	334.00	249.00	181.00	135.00	106.00	72.00
all_IRFinder-S_RIs	2603.00	2603.00	2603.00	2603.00	2603.00	2603.00	2603.00	2603.00	2603.00	2603.00
IRFinder-S_true_positives	1324.00	456.00	223.00	148.00	94.00	57.00	32.00	23.00	19.00	9.00
IRFinder-S_false_positives	1279.00	2147.00	2380.00	2455.00	2509.00	2546.00	2571.00	2580.00	2584.00	2594.00
IRFinder-S_false_negatives	3261.00	886.00	452.00	310.00	240.00	192.00	149.00	112.00	87.00	63.00
IRFinder-S_precision	0.51	0.18	0.09	0.06	0.04	0.02	0.01	0.01	0.01	0.00
IRFinder-S_recall	0.29	0.34	0.33	0.32	0.28	0.23	0.18	0.17	0.18	0.13
IRFinder-S_fscore	0.37	0.23	0.14	0.10	0.06	0.04	0.02	0.02	0.01	0.01
all_superintronic_RIs	3184.00	3184.00	3184.00	3184.00	3184.00	3184.00	3184.00	3184.00	3184.00	3184.00
superintronic_true_positives	1413.00	383.00	199.00	127.00	90.00	62.00	45.00	32.00	24.00	14.00
superintronic_false_positives	1771.00	2801.00	2985.00	3057.00	3094.00	3122.00	3139.00	3152.00	3160.00	3170.00
superintronic_false_negatives	3172.00	959.00	476.00	331.00	244.00	187.00	136.00	103.00	82.00	58.00
superintronic_precision	0.44	0.12	0.06	0.04	0.03	0.02	0.01	0.01	0.01	0.00
superintronic_recall	0.31	0.29	0.29	0.28	0.27	0.25	0.25	0.24	0.23	0.19
superintronic_fscore	0.36	0.17	0.10	0.07	0.05	0.04	0.03	0.02	0.01	0.01
all_iREAD_RIs	533.00	533.00	533.00	533.00	533.00	533.00	533.00	533.00	533.00	533.00
iREAD_true_positives	169.00	41.00	17.00	12.00	10.00	7.00	5.00	3.00	1.00	1.00
iREAD_false_positives	364.00	492.00	516.00	521.00	523.00	526.00	528.00	530.00	532.00	532.00
iREAD_false_negatives	4416.00	1301.00	658.00	446.00	324.00	242.00	176.00	132.00	105.00	71.00
iREAD_precision	0.32	0.08	0.03	0.02	0.02	0.01	0.01	0.01	0.00	0.00
iREAD_recall	0.04	0.03	0.03	0.03	0.03	0.03	0.03	0.02	0.01	0.01
iREAD_fscore	0.07	0.04	0.03	0.02	0.02	0.02	0.01	0.01	0.00	0.00
all_kma_RIs	3084.00	3084.00	3084.00	3084.00	3084.00	3084.00	3084.00	3084.00	3084.00	3084.00
kma_true_positives	796.00	196.00	104.00	66.00	42.00	25.00	18.00	9.00	7.00	4.00
kma_false_positives	2288.00	2888.00	2980.00	3018.00	3042.00	3059.00	3066.00	3075.00	3077.00	3080.00
kma_false_negatives	3789.00	1146.00	571.00	392.00	292.00	224.00	163.00	126.00	99.00	68.00
kma_precision	0.26	0.06	0.03	0.02	0.01	0.01	0.01	0.00	0.00	0.00
kma_recall	0.17	0.15	0.15	0.14	0.13	0.10	0.10	0.07	0.07	0.06
KMA_fscore	0.21	0.09	0.06	0.04	0.02	0.02	0.01	0.01	0.00	0.00
all_IntEREst_RIs	3177.00	3177.00	3177.00	3177.00	3177.00	3177.00	3177.00	3177.00	3177.00	3177.00
IntEREst_true_positives	843.00	203.00	116.00	74.00	51.00	31.00	26.00	16.00	10.00	6.00
IntEREst_false_positives	2334.00	2974.00	3061.00	3103.00	3126.00	3146.00	3151.00	3161.00	3167.00	3171.00
IntEREst_false_negatives	3742.00	1139.00	559.00	384.00	283.00	218.00	155.00	119.00	96.00	66.00

Table D.5: HX1 potential RIs. Preview of supplemental table; full table available online.

longread_persistence_threshold	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80
longread_intron_count	4585.00	1342.00	675.00	458.00	334.00	249.00	181.00	135.00	106.00
all_IRFinder-S_RIs	12772.00	12772.00	12772.00	12772.00	12772.00	12772.00	12772.00	12772.00	12772.00
IRFinder-S_true_positives	3479.00	967.00	468.00	312.00	219.00	153.00	99.00	72.00	58.00
IRFinder-S_false_positives	9293.00	11805.00	12304.00	12460.00	12553.00	12619.00	12673.00	12700.00	12714.00
IRFinder-S_false_negatives	1106.00	375.00	207.00	146.00	115.00	96.00	82.00	63.00	48.00
IRFinder-S_precision	0.27	0.08	0.04	0.02	0.02	0.01	0.01	0.01	0.00
IRFinder-S_recall	0.76	0.72	0.69	0.68	0.66	0.61	0.55	0.53	0.55
IRFinder-S_fscore	0.40	0.14	0.07	0.05	0.03	0.02	0.02	0.01	0.01
all_superintronic_RIs	9208.00	9208.00	9208.00	9208.00	9208.00	9208.00	9208.00	9208.00	9208.00
superintronic_true_positives	2536.00	613.00	282.00	178.00	124.00	83.00	57.00	42.00	33.00
superintronic_false_positives	6672.00	8595.00	8926.00	9030.00	9084.00	9125.00	9151.00	9166.00	9175.00
superintronic_false_negatives	2049.00	729.00	393.00	280.00	210.00	166.00	124.00	93.00	73.00
superintronic_precision	0.28	0.07	0.03	0.02	0.01	0.01	0.01	0.00	0.00
superintronic_recall	0.55	0.46	0.42	0.39	0.37	0.33	0.31	0.31	0.31
superintronic_fscore	0.37	0.12	0.06	0.04	0.03	0.02	0.01	0.01	0.01
all_iREAD_RIs	4657.00	4657.00	4657.00	4657.00	4657.00	4657.00	4657.00	4657.00	4657.00
iREAD_true_positives	736.00	179.00	82.00	48.00	26.00	18.00	12.00	4.00	2.00
iREAD_false_positives	3921.00	4478.00	4575.00	4609.00	4631.00	4639.00	4645.00	4653.00	4655.00
iREAD_false_negatives	3849.00	1163.00	593.00	410.00	308.00	231.00	169.00	131.00	104.00
iREAD_precision	0.16	0.04	0.02	0.01	0.01	0.00	0.00	0.00	0.00
iREAD_recall	0.16	0.13	0.12	0.10	0.08	0.07	0.07	0.03	0.02
iREAD_fscore	0.16	0.06	0.03	0.02	0.01	0.01	0.00	0.00	0.00
all_kma_RIs	12533.00	12533.00	12533.00	12533.00	12533.00	12533.00	12533.00	12533.00	12533.00
kma_true_positives	1779.00	430.00	191.00	117.00	77.00	48.00	32.00	22.00	18.00
kma_false_positives	10754.00	12103.00	12342.00	12416.00	12456.00	12485.00	12501.00	12511.00	12515.00
kma_false_negatives	2806.00	912.00	484.00	341.00	257.00	201.00	149.00	113.00	88.00
kma_precision	0.14	0.03	0.02	0.01	0.01	0.00	0.00	0.00	0.00
kma_recall	0.39	0.32	0.28	0.26	0.23	0.19	0.18	0.16	0.17
KMA_fscore	0.21	0.06	0.03	0.02	0.01	0.01	0.01	0.00	0.00
all_IntEREst_RIs	19384.00	19384.00	19384.00	19384.00	19384.00	19384.00	19384.00	19384.00	19384.00
IntEREst_true_positives	2876.00	670.00	291.00	176.00	115.00	74.00	51.00	35.00	28.00
IntEREst_false_positives	16508.00	18714.00	19093.00	19208.00	19269.00	19310.00	19333.00	19349.00	19356.00
IntEREst_false_negatives	1709.00	672.00	384.00	282.00	219.00	175.00	130.00	100.00	78.00
IntEREst_precision	0.15	0.03	0.02	0.01	0.01	0.00	0.00	0.00	0.00
IntEREst_recall	0.63	0.50	0.43	0.38	0.34	0.30	0.28	0.26	0.26

Table D.6: iPSC called RIs. Preview of supplemental table; full table available online.

longread_persistence_threshold	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80	0.90
longread_intron_count	3194.00	782.00	327.00	212.00	176.00	140.00	109.00	75.00	61.00	40.00
all_IRFinder-S_RIs	783.00	783.00	783.00	783.00	783.00	783.00	783.00	783.00	783.00	783.00
IRFinder-S_true_positives	424.00	150.00	62.00	37.00	28.00	23.00	15.00	9.00	6.00	3.00
IRFinder-S_false_positives	359.00	633.00	721.00	746.00	755.00	760.00	768.00	774.00	777.00	780.00
IRFinder-S_false_negatives	2770.00	632.00	265.00	175.00	148.00	117.00	94.00	66.00	55.00	37.00
IRFinder-S_precision	0.54	0.19	0.08	0.05	0.04	0.03	0.02	0.01	0.01	0.00
IRFinder-S_recall	0.13	0.19	0.19	0.17	0.16	0.16	0.14	0.12	0.10	0.08
IRFinder-S_fscore	0.21	0.19	0.11	0.07	0.06	0.05	0.03	0.02	0.01	0.01
all_superintronic_RIs	2956.00	2956.00	2956.00	2956.00	2956.00	2956.00	2956.00	2956.00	2956.00	2956.00
superintronic_true_positives	1017.00	216.00	83.00	55.00	39.00	29.00	18.00	13.00	9.00	5.00
superintronic_false_positives	1939.00	2740.00	2873.00	2901.00	2917.00	2927.00	2938.00	2943.00	2947.00	2951.00
superintronic_false_negatives	2177.00	566.00	244.00	157.00	137.00	111.00	91.00	62.00	52.00	35.00
superintronic_precision	0.34	0.07	0.03	0.02	0.01	0.01	0.01	0.00	0.00	0.00
superintronic_recall	0.32	0.28	0.25	0.26	0.22	0.21	0.17	0.17	0.15	0.13
superintronic_fscore	0.33	0.12	0.05	0.03	0.02	0.02	0.01	0.01	0.01	0.00
all_iREAD_RIs	367.00	367.00	367.00	367.00	367.00	367.00	367.00	367.00	367.00	367.00
iREAD_true_positives	87.00	23.00	5.00	3.00	3.00	3.00	2.00	1.00	1.00	1.00
iREAD_false_positives	280.00	344.00	362.00	364.00	364.00	364.00	365.00	366.00	366.00	366.00
iREAD_false_negatives	3107.00	759.00	322.00	209.00	173.00	137.00	107.00	74.00	60.00	39.00
iREAD_precision	0.24	0.06	0.01	0.01	0.01	0.01	0.01	0.00	0.00	0.00
iREAD_recall	0.03	0.03	0.02	0.01	0.02	0.02	0.02	0.01	0.02	0.03
iREAD_fscore	0.05	0.04	0.01	0.01	0.01	0.01	0.01	0.00	0.00	0.00
all_kma_RIs	3676.00	3676.00	3676.00	3676.00	3676.00	3676.00	3676.00	3676.00	3676.00	3676.00
kma_true_positives	545.00	117.00	43.00	28.00	18.00	14.00	9.00	4.00	3.00	1.00
kma_false_positives	3131.00	3559.00	3633.00	3648.00	3658.00	3662.00	3667.00	3672.00	3673.00	3675.00
kma_false_negatives	2649.00	665.00	284.00	184.00	158.00	126.00	100.00	71.00	58.00	39.00
kma_precision	0.15	0.03	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.00
kma_recall	0.17	0.15	0.13	0.13	0.10	0.10	0.08	0.05	0.05	0.03
KMA_fscore	0.16	0.05	0.02	0.01	0.01	0.01	0.00	0.00	0.00	0.00
all_IntEREst_RIs	4083.00	4083.00	4083.00	4083.00	4083.00	4083.00	4083.00	4083.00	4083.00	4083.00
IntEREst_true_positives	731.00	142.00	52.00	35.00	24.00	17.00	9.00	5.00	4.00	1.00
IntEREst_false_positives	3352.00	3941.00	4031.00	4048.00	4059.00	4066.00	4074.00	4078.00	4079.00	4082.00
IntEREst_false_negatives	2463.00	640.00	275.00	177.00	152.00	123.00	100.00	70.00	57.00	39.00
IntEREst_precision	0.18	0.03	0.01	0.01	0.01	0.00	0.00	0.00	0.00	0.00
IntEREst_recall	0.23	0.18	0.16	0.17	0.14	0.12	0.08	0.07	0.07	0.03

Table D.7: iPSC potential RIs. Preview of supplemental table; full table available online.

longread_persistence_threshold	0.00	0.10	0.20	0.30	0.40	0.50	0.60	0.70	0.80
longread_intron_count	3194.00	782.00	327.00	212.00	176.00	140.00	109.00	75.00	61.00
all_IRFinder-S_RIs	10434.00	10434.00	10434.00	10434.00	10434.00	10434.00	10434.00	10434.00	10434.00
IRFinder-S_true_positives	2147.00	501.00	199.00	122.00	99.00	77.00	59.00	40.00	33.00
IRFinder-S_false_positives	8287.00	9933.00	10235.00	10312.00	10335.00	10357.00	10375.00	10394.00	10401.00
IRFinder-S_false_negatives	1047.00	281.00	128.00	90.00	77.00	63.00	50.00	35.00	28.00
IRFinder-S_precision	0.21	0.05	0.02	0.01	0.01	0.01	0.01	0.00	0.00
IRFinder-S_recall	0.67	0.64	0.61	0.58	0.56	0.55	0.54	0.53	0.54
IRFinder-S_fscore	0.32	0.09	0.04	0.02	0.02	0.01	0.01	0.01	0.01
all_superintronic_RIs	7465.00	7465.00	7465.00	7465.00	7465.00	7465.00	7465.00	7465.00	7465.00
superintronic_true_positives	1499.00	296.00	99.00	60.00	43.00	32.00	20.00	13.00	9.00
superintronic_false_positives	5966.00	7169.00	7366.00	7405.00	7422.00	7433.00	7445.00	7452.00	7456.00
superintronic_false_negatives	1695.00	486.00	228.00	152.00	133.00	108.00	89.00	62.00	52.00
superintronic_precision	0.20	0.04	0.01	0.01	0.01	0.00	0.00	0.00	0.00
superintronic_recall	0.47	0.38	0.30	0.28	0.24	0.23	0.18	0.17	0.15
superintronic_fscore	0.28	0.07	0.03	0.02	0.01	0.01	0.01	0.00	0.00
all_iREAD_RIs	11225.00	11225.00	11225.00	11225.00	11225.00	11225.00	11225.00	11225.00	11225.00
iREAD_true_positives	1373.00	250.00	74.00	44.00	33.00	22.00	13.00	7.00	6.00
iREAD_false_positives	9852.00	10975.00	11151.00	11181.00	11192.00	11203.00	11212.00	11218.00	11219.00
iREAD_false_negatives	1821.00	532.00	253.00	168.00	143.00	118.00	96.00	68.00	55.00
iREAD_precision	0.12	0.02	0.01	0.00	0.00	0.00	0.00	0.00	0.00
iREAD_recall	0.43	0.32	0.23	0.21	0.19	0.16	0.12	0.09	0.10
iREAD_fscore	0.19	0.04	0.01	0.01	0.01	0.00	0.00	0.00	0.00
all_kma_RIs	8352.00	8352.00	8352.00	8352.00	8352.00	8352.00	8352.00	8352.00	8352.00
kma_true_positives	1023.00	202.00	66.00	39.00	28.00	22.00	13.00	6.00	5.00
kma_false_positives	7329.00	8150.00	8286.00	8313.00	8324.00	8330.00	8339.00	8346.00	8347.00
kma_false_negatives	2171.00	580.00	261.00	173.00	148.00	118.00	96.00	69.00	56.00
kma_precision	0.12	0.02	0.01	0.00	0.00	0.00	0.00	0.00	0.00
kma_recall	0.32	0.26	0.20	0.18	0.16	0.16	0.12	0.08	0.08
KMA_fscore	0.18	0.04	0.02	0.01	0.01	0.01	0.00	0.00	0.00
all_IntEREst_RIs	12743.00	12743.00	12743.00	12743.00	12743.00	12743.00	12743.00	12743.00	12743.00
IntEREst_true_positives	1608.00	301.00	84.00	48.00	34.00	22.00	13.00	7.00	6.00
IntEREst_false_positives	11135.00	12442.00	12659.00	12695.00	12709.00	12721.00	12730.00	12736.00	12737.00
IntEREst_false_negatives	1586.00	481.00	243.00	164.00	142.00	118.00	96.00	68.00	55.00
IntEREst_precision	0.13	0.02	0.01	0.00	0.00	0.00	0.00	0.00	0.00
IntEREst_recall	0.50	0.38	0.26	0.23	0.19	0.16	0.12	0.09	0.10