2014

Methods for Integrative Analysis of Genomic Data

Paul Manser
Virginia Commonwealth University

METHODS FOR INTEGRATIVE ANALYSIS OF GENOMIC DATA

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at Virginia Commonwealth University.

by

PAUL T. MANSER

M.S., University of Virginia, 2011

B.A., University of Virginia, 2009

Director: Mark Reimers, Ph.D.,

Assistant Professor, Department of Biostatistics

Virginia Commonwealth University

Richmond, Virginia

December, 2014

Acknowledgements

I would like to thank my advisor Dr. Mark Reimers for his support and guidance and for the many research opportunities and learning experiences he has provided me. My knowledge of both statistics and neuroscience has grown greatly because of him. It has truly been a great privilege and pleasure having him as a mentor.

I would like to thank my committee members, Dr. Kellie Archer, Dr. Michael Neale, Dr. Nitai Mukhopadhyay, and Dr. Shirley Taylor, for their time and effort in reading my oral exam and dissertation and for their helpful comments and suggestions. I would like to especially thank Dr. Archer and Dr. Mukhopadhyay for their statistical advice and help proofreading. I would like to thank Dr. Neale for supporting me on his training grant and for letting me make VIPBG my second home. I would like to thank Dr. Taylor for her added biological insight.

I would like to thank the students of the Biostatistics Department and VIPBG for their help and friendship along the way. I would like to thank Dr. Vernell Williamson for her guidance and patience in helping me with data preprocessing. I would also like to thank Dr. Donna McClish and Russell Boyle for making sure I met important dead lines and actually graduated on time.

Lastly, I would like to thank my family for their unwavering support, love, and for listening to countless stressed out phone calls on Sunday nights. I couldn't have made it this far without them.

TABLE OF CONTENTS

Chapter Page

LIST OF TABLES

LIST OF FIGURES

Abstract

METHODS FOR INTEGRATIVE ANALYSIS OF GENOMIC DATA

Paul T. Manser, M.S.

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at Virginia Commonwealth University.

In recent years, the development of new genomic technologies has allowed for the investigation of many regulatory epigenetic marks besides expression levels, on a genome-wide scale. As the price for these technologies continues to decrease, study sizes will not only increase, but several different assays are beginning to be used for the same samples. It is therefore desirable to develop statistical methods to integrate multiple data types that can handle the increased computational burden of incorporating large data sets. Furthermore, it is important to develop sound quality control and normalization methods as technical errors can compound when integrating multiple genomic assays.

DNA methylation is a commonly studied epigenetic mark, and the Infinium HumanMethylation450 BeadChip has become a popular microarray that provides

genome-wide coverage and is affordable enough to scale to larger study sizes. It employs a complex array design that has complicated efforts to develop normalization methods. We propose a novel normalization method that uses a set of stable methylation sites from housekeeping genes as empirical controls to fit a local regression hypersurface to signal intensities. We demonstrate that our method performs favorably compared to other popular methods for the array. We also discuss an approach to estimating cell-type admixtures, which is a frequent biological confound in these studies.

For data integration we propose a gene-centric procedure that uses canonical correlation and subsequent permutation testing to examine correlation or other measures of association and co-localization of epigenetic marks on the genome. Specifically, a likelihood ratio test for general association between data modalities is performed after an initial dimension reduction step. Canonical scores are then regressed against covariates of interest using linear mixed effects models. Lastly, permutation testing is performed on weighted correlation matrices to test for co-localization of relationships to physical locations in the genome. We demonstrate these methods on a set of developmental brain samples from the BrainSpan consortium and find substantial relationships between DNA methylation, gene expression, and alternative promoter usage primarily in genes related to axon guidance. We perform a second integrative analysis on another set of brain samples from the Stanley Medical Research Institute.

CHAPTER 1

INTRODUCTION

1.1 Overview of necessary molecular biology

1.1.1 The central dogma of molecular biology

The central dogma of molecular biology states that information in an organism is stored in DNA as nucleic acid sequences, but is functional in the form of protein polypeptides. Information in DNA propagates by transcription into RNA which is then translated into these proteins (Krebs, Goldstein, and Kilpatrick 2013, Ch. 1.8). Discoveries in the field of epigenetics have found that information is not only stored within DNA sequences, but also on DNA in its surrounding protein structures. In fact, a single gene may be transcribed several different ways, with modification of epigenetic factors playing a role in the process.

The transcription of DNA into RNA begins with the binding of a collection of proteins known as the transcription apparatus to an area at the beginning of a gene called the promoter. Proteins called transcription factors bind in this promoter region and potentially in associated distal regions called enhancers to initiate gene transcription. An enzyme called RNA polymerase, which actually synthesizes the resulting RNA is also part of this transcription complex. Once the transcription apparatus and transcription factors have assembled, transcription starts at the $5'$ end of the gene moving towards the $3'$ end. As RNA polymerase moves along the gene, it creates a single-stranded RNA molecule with bases complementary to the DNA sequence being transcribed, with exception of thymine being replaced by uracil.

When the polymerase reaches the end of the gene, it falls off and the RNA transcript is released (Krebs, Goldstein, and Kilpatrick 2013, Ch. 20).

Before the RNA leaves the nucleus, it undergoes several processing steps. First, a guanine base is added to the $5'$ end of the RNA (commonly referred to as the $5'$ cap) which usually occurs soon after transcription initiation. The $5'$ cap serves to protect the RNA from degradation by certain kinds of exonucleases. Once the gene has finished transcription, another polymerase called poly(A) polymerase adds a stretch of roughly 200 adenosine bases to the end of the RNA to create what is commonly referred to as the poly(A) tail. The poly(A) tail serves to insulate the coding sequence of the RNA, provide stability, and again protect it from degradation. A specific protein binds to this poly(A) tail to help further protect from degradation as well as facilitate RNA translation into protein (Krebs, Goldstein, and Kilpatrick 2013, Ch. 21). The poly(A) tail is also commonly used for identifying and isolating RNA before performing microarray and next generation sequencing (NGS) experiments.

Once RNA passes out of the nucleus into the cytoplasm, a free-floating ribosome attaches itself to the RNA molecule to begin the process of translation. Once a ribosome has initiated translation, the RNA is translated into protein one codon at a time. Codons are three-base sequences of DNA that correspond to a specific amino acid which are the building blocks of proteins. Since there are four bases, there are $4^3 = 64$ possibile codons. However, there are only 20 main amino acids used to build proteins, so several 3 base sequences can code for the same amino acid, with usually the third base being allowed to vary. Other codons indicate the start and stop sites for ribosomes to translate the protein. Once translation finishes, the ribosome falls off the protein, which then may require further processing and folding before becoming fully functional (Krebs, Goldstein, and Kilpatrick 2013, Ch. 25).

1.1.2 Introduction of epigenetics

Although DNA is popularly portrayed as existing as a lone, elegant double helix, it is in fact rarely found in this form in living cells. Instead, it is tightly packed and wound around DNA-binding proteins that give it support and structure. Histones, one major category of DNA-binding proteins, combine to form nucleosomes which function as the basic unit of DNA packaging. Modifications of these histones can locally control how DNA is packaged which determines how accessible DNA is to transcription factors and other proteins floating around in the nucleus required for transcription (Krebs, Goldstein, and Kilpatrick 2013, Ch. 29).

Another epigenetic factor affecting DNA accessibility is DNA methylation. In mammals, DNA methylation generally consists of the addition of a methyl group to a cytosine base. DNA methylation often occurs in CpG sites which are 2 base palindromes that are read as CG in either direction on the DNA, with both cytosines usually being methylated. Non-palindromic strand-specific DNA methylation can also occur in brain tissue (Lister et al. 2013). The addition of a methyl group acts as a bump on DNA that can hinder the binding of transcription factors and other proteins, although certain proteins such as MECP2 bind specifically to methylated DNA, but enhance its repressive effect. Cancer studies have shown that methylation of promoter regions of genes has a silencing effect on gene expression (Baylin et al. 2001; Warden et al. 2013), while more recent studies suggest that DNA methylation in gene bodies and other regions may play a more subtle role in gene regulation (Maunakea et al. 2013).

1.1.3 The role of epigenetics in modifying transcription

While there are currently believed to be roughly twenty thousand genes in the human genome, initial estimates were much higher. This overestimate was partly due to a phenomenon known as alternative splicing which allows a single gene to code for multiple RNAs. While an RNA is being transcribed, it can be cut apart and put back together into multiple different configurations by a complex of RNA and proteins called the spliceosome. These different isoforms of RNA can then go on to code different functional protein forms.

A typical gene consists of two major types of regions: introns and exons. While a gene exists on a single stretch of DNA, generally not all of it is ultimately translated into protein. First, DNA is transcribed into a premature RNA, which includes both introns and exons. Once the genic DNA sequence is transcribed, the spliceosome removes introns from the transcript, leaving only the exons in the final transcript.

Alternative splicing occurs when these introns and exons are excluded or included in the final RNA transcript in different combinations. Sometimes introns may not be excised, and can be included in the final transcripts. Additionally, certain exons may be removed, or some may be mutually exclusive. A simple metric for assessing alternative splicing is to look at how often an exon is included in the total number of transcripts for a gene. This can be thought of in general terms as an exon inclusion ratio or percentage. Most exons should be included in close to 100% of the transcripts, but some may be included in only 30%, or perhaps not at all in a certain tissue. Genes can also have multiple transcription start sites, where start sites can begin in the middle of the full gene and code for transcripts excluding multiple upstream exons.

Recent studies have suggested a role for DNA methylation in the regulation of

alternative splicing (Maunakea et al. 2013; Cingolani et al. 2013). Exons that are spliced out generally seem to have a lower level of DNA methylation than similar exons that are constitutively included (Maunakea et al. 2013). However, increased DNA methylation in transcription factor binding sites proximal to exons can have the reverse effect(Shukla et al. 2011). These findings are observational, and cannot establish a causal relationship between increases in exonic DNA methylation and exon inclusion. However, a study in bees showed that experimentally induced changes in DNMT3, an enzyme that catalyzes the addition of methyl groups to CpG sites, was able to change patterns of alternative splicing (Cingolani et al. 2013). As genomic technologies become more affordable and reliable, integrative studies will be able to establish relationships between gene expression, alternative splicing and DNA methylation as well as other epigenetic marks.

1.2 Overview of the Infinium HumanMethylation450 BeadChip

1.2.1 Microarray design

The Infinium HumanMethylation450 BeadChip (also known as the Illumina 450k array) is a bead-based microarray that can assess DNA methylation on a genome-wide scale at over 480,000 CpG sites (Bibikova et al. 2011). The 450k array surpasses its predecessor, the Illumina 27k array (Bing Fan 2010), by providing additional coverage of CpG sites particularly in non-promoter regions and gene bodies. It is able to accomplish this by employing a complex array design that uses multiple bead types to reduce the amount of space needed on the array. The complex design along with the popularity of the 450k array have made it a popular platform for statisticians to develop normalization methods.

The Illumina 450k array uses a bisulfite treatment to assess methylation status.

Treatment with bisulfite converts unmethylated cytosine to uracil, while methylated cytosines remain unaffected. This treatment creates what might be considered a pseudo-SNP (Single nucleotide polymorphism) with methylated loci having one allele and unmethylated loci having another. After the bisulfite conversion, DNA is amplified using the whole-genome amplification reaction, fragmented enzymatically, precipitated, and suspended in a hybridization buffer (Bibikova et al. 2011). It is then applied to the array and allowed to hybridize for twenty hours.

The microarray generates two signals for each methylation site: one for the methylated state, and one for the unmethylated state. The Illumina 450k array is a bead-based array, meaning that probes for specific DNA sequences are not directly attached to the array, but rather are attached to beads which are washed over the array and settle in wells. Beads are identified by a unique 23 base barcode "address" sequence at the base of probes. The 450k array has two bead types that both share this common mechanism of identification.

Type I beads are the older bead technology on the array, inherited from the previous 27k array. They mostly target CpGs in promoter regions of genes (Bing Fan 2010). For a given CpG, there are actually two beads, one with a sequence specific to the methylated state, and one specific to the unmethylated state. The CpG site of interest occupies the last two bases at the tail of the probe sequence. After the DNA is hybridized to the probe, a florescent base is added that is complementary to the next base after the CpG site. If the hybridized sequence matches perfectly (has the correct methylation state), then the florescent base is added on at the end, giving off a burst of light. Each bead then gives off a signal giving a measure of each of the two possible states.

Type II probes are a more recent technology, added specifically for the 450k array. Their advantage over the type I beads is that they only require one bead type

and therefore less space on the chip. The single probe type on the type II bead has a non-specific sequence that will match either of the methylated states for the targeted CpG. The last base of the probe targets the first half (the G base) of the CpG. Two florescent bases are then added to the assay and will selectively hybridize to the end of the probe depending on whether the cytosine has been bisulfite converted or not. In order for the Type II probes to work, each of the two fluorescent bases must operate in different color channels since they are competitively hybridizing to the same location. This competitive hybridization seems to result in lower data quality relative to type I probes.

Resulting output after scanning arrays and recording signal intensities are stored in Intensity Data Files (.idat). These files contain all the signal information extracted from the array including negative control probes as well as signal intensities from the unused color channel of type I probes. Several normalization methods require .idat files, but often only summary measures are available from online repositories such as the Gene Expression Omnibus (GEO). However, other repositories such as The Cancer Genome Atlas (TCGA) have .idat level data publicly available.

The 450k array is a cost effective approach for assaying DNA methylation, and samples can be run in parallel in batches of twelve. Despite the increase in coverage relative to the 27k array, the array still surveys only roughly one percent of the CpGs in the human genome. Furthermore, coverage in gene bodies can be somewhat sparse and varies from gene to gene which can narrow the scope of certain types of analyses. Nevertheless, the 450k array provides a scalable solution to assaying DNA methylation with relatively high coverage on a large number of samples.

1.2.2 Summarization methods

1.2.2.1 β-values

The β-value is the standard summarization method for the 450k array. Subsequent methods for summarization either modify or aggregate β-values in some way. β-values combine unmethylated and methylated signals into a single measure. Equation 1.1 gives the formula for β-values where M is the methylated signal intensity, U is the unmethylated signal intensity, and ϵ is a small offset parameter suggested by Illumina which is set to 100 by default that stabilizes β-values when both M and U are small.

$$\beta = \frac{M}{M + U + \epsilon} \tag{1.1}$$

β-values can be interpreted as a measure of "proportion methylated." A β-value close to zero implies the locus is not methylated, while a β-value close to one implies it is methylated. Intermediate β-values can mean several things. Some loci are imprinted and are methylated only on one chromosome which will result in a β-value near 0.5. Loci can be hemi-methylated where only one cytosine in a CpG site is methylated which can also result in a β-value near 0.5. Lastly, only a subset of cells in a sample may be methylated. If 30% of cells in a sample are methylated at a given locus, then this will result in a β-value near 0.3. It is therefore important to be careful when interpreting β-values, as they may be reflecting one or more of these phenomena.

1.2.2.2 M-values

One potential disadvantage of β-values is that their range is bounded below by zero and above by one. This boundedness can create data that violate the normality

assumption for many common statistical methods such as simple linear models and t-tests. β-values also have problems with heteroscedasticity for highly methylated or unmethylated CpG sites (Du et al. 2010). In order to transform β-values to span the real line, a logit transform using $\log_2$ is used to compute M-values (Equation 1.2). The M-value method provides better performance in terms of Detection Rate (DR) and True Positive Rate (TPR) for both highly methylated and unmethylated CpG sites (Du et al. 2010). M-values however, are not as straightforward to interpret as β-values.

$$M = \log_2 \left(\frac{\beta}{1 - \beta} \right) \tag{1.2}$$

1.2.2.3 Bump Hunting

DNA methylation can be highly correlated within local regions (Zhang et al. 2013). Therefore, nearby probes may be redundant and it may make more sense to aggregate them and fit region-level models when analyzing methylation data. Functional biological mechanisms may also correspond to regional changes rather than single CpG differences (e.g. promoter regions and CpG Islands). A bump hunting approach for performing aggregation and significance testing has recently been suggested by Jaffe et al. 2012.

The approach is as follows:

1. M-values are regressed against covariates of interest for each probe.
2. Regression coefficients are then smoothed over genomic location using a loess curve.
3. Predefined thresholds for effect sizes are then used to find contiguous regions where smoothed coefficient estimates are above the specified threshold.

4. The area under the loess curve for the contiguous region is then taken as a test statistic.

5. Significance testing is then performed by comparing the area against a null permutation distribution.

While the bump hunting method for summarization is specific to the subsequent analysis, the idea of summarizing DNA methylation locally is important. Summarization not only reduces the number of eventual significance tests, but can also reduce the correlation among these tests since correlated CpG sites are aggregated.

1.3 Overview of the Affymetrix Human Exon 1.0 ST Array

1.3.1 Microarray design

Traditionally, gene expression microarrays have targeted multiple parts of the 3′ tail end of an RNA transcript using sets of complementary probes (probesets) that are then summarized into a single measure of expression. These 3′ regions of the gene are believed to be included in all transcripts. While expression arrays give a measure of overall gene abundance, they give no insight into the types of gene modifications, such as alternative splicing or alternative transcription start sites, which may be occurring upstream from the 3′ end. The Affymetrix Human Exon 1.0 ST Array contains an increased number of probesets that target all putative exonic regions of a gene (*GeneChip Exon Array Design* 2005). For the Affymetrix Exon Array, probesets for exonic regions generally consist of a set of four probes, but longer exons or extended 3′ UTR regions may have multiple probesets. Unlike the Illumina 450k array, the Affymetrix Exon array does not use beads, but rather has a static design with each probe anchored to a fixed point on the chip with known X and Y coordinates in a grid. This makes quality control and adjustment for spatial artifacts simpler than in

the case of the 450k array.

After RNA is isolated from a sample and fragmented, it is then reverse-transcribed into complementary DNA, or cDNA. After reverse transcription, the cDNA is amplified, labeled with a fluorescent dye, and hybridized to the microarray. If a cDNA molecule binds to a probe, it fluoresces indicating the presence of that particular exon in that sample. Signal intensities are captured with a camera, and quantified. Once signal intensities are obtained, many methods exist for preprocessing, summarization, and analysis.

1.3.2 Summarization methods

A unique issue to the Affymetrix Exon array is that in order to measure alternative splicing it is necessary to obtain reliable measures of two different kinds of information: The first is a measure of overall gene expression. The second is a measure of exon-specific expression. Some models for assessing alternative splicing treat aggregate gene expression as a model parameter that is estimated rather than directly computing summary statistics for exon inclusion (Purdom et al. 2008; Cline et al. 2005). If familiar statistical methods are to be directly applied, then a direct measure of exon inclusion needs to be computed. This is commonly done by taking the ratio of exon expression levels with the aggregate gene expression level. These measures can then be interpreted as an approximate measure of how many gene transcripts contain the given exon. Here we briefly review methods for summarization for the Affymetrix Human Exon 1.0 ST Array.

1.3.2.1 RMA

Robust multi-chip average, or RMA, is a popular method for obtaining expression measures from gene expression microarrays (Irizarry et al. 2003). It performs

background correction, normalization, and summarization. RMA first performs a background correction using a normal-exponential deconvolution. The equation for background correction is given in Equation 1.3. Signal intensities y_{ijk} for probe k in probeset j on array i are modeled as a function of probe-specific signal ps_{ijk} and non-specific background bg_{ijk} .

$$y_{ijk} = ps_{ijk} + bg_{ijk} \tag{1.3}$$

Here ps is exponentially distributed, $bg \sim N(0, \sigma^2)$ and $ps \perp\!\!\!\perp bg$. Once background correction is performed, RMA then performs quantile normalization (Bolstad et al. 2003). The steps for the quantile normalization algorithm are given below.

1. Let Y_i be the vector of signal intensities for array i
2. Sort each vector Y_i from largest to smallest to obtain Y_i^*
3. Compute the mean sorted vector $\bar{Y}^* = \frac{\sum_{i=1}^{I} Y_i}{I}$
4. Replace each value of Y_i^* with the corresponding mean value from $\bar{Y}^*$
5. Unsort each vector Y_i, returning it to its original ordering

Once quantile normalization is performed, probe sets are summarized to obtain a single measure of expression using an additive linear model given in Equation 1.4.

$$y_{ijk} = \mu_j + P_{jk} + M_{ij} + \epsilon_{ijk} \tag{1.4}$$

Here μ_j denotes the overall mean for probeset j, P_{jk} is the probe-specific effect for probe k, and M_{ij} is the sample effect on the probe set. The expression summary for a probe set j on array i is then given by $\hat{\mu}_j + \hat{M}_{ij}$. Tukey's median polish is then used to obtain estimates of the parameters. An implementation of RMA for the

Affymetrix ST 1.0 Exon Array exists in the oligo package in R (Carvalho and Irizarry 2010a; Carvalho and Irizarry 2010b).

1.3.2.2 Splicing Index

The splicing index, or exon inclusion ratio, is a straightforward and simple method for characterizing exon inclusion. Both the geneBASE and COSIE methods compute splicing indices using different approaches. It is a ratio of exon-specific expression to aggregate gene expression. Ideally, a measure of exon inclusion would take on a value between zero and one the way that a β-value does. However, since dynamic ranges of different exon probesets can vary substantially for purely technical reasons, the splicing index often takes on values greater than one and cannot be directly interpreted like a β-value. Instead, splicing indices are generally transformed to the $\log_2$ scale. A general form for the splicing index is given in Equation 1.5.

$$\text{Splicing Index} = \log_2 \left(\frac{\text{exon expression}}{\text{gene expression}} \right) \tag{1.5}$$

1.3.2.3 geneBASE

geneBASE uses a data-driven approach to obtain an aggregate measure for gene expression (Xing, Kapur, and Wong 2006). Rather than using the Affymetrix annotations for constitutive exons, geneBASE performs hierarchical clustering using a correlation distance metric for each gene across a tissue panel. The set of probes meeting a correlation threshold are declared as constitutive and are then summarized using the RMA linear model to get an aggregate measure of gene expression. The geneBASE paper demonstrates that Affymetrix annotations of constitutive exons are often incorrect and that their method provides a better measure of aggregate gene expression, which is important for obtaining accurate measures of exon inclusion.

1.3.2.4 COSIE

COrrected Splicing Indices for Exon arrays, or COSIE, is a method that attempts to correct for systematic biases in the detection of alternative splicing on the HT1 Exon Array (Gaidatzis et al. 2009). Since each exon is represented by a probeset that covers a small genomic region, probe sequence content can vary substantially between probesets for different exons on the same gene. Probe sequence content affects mRNA binding efficiency to microarray probes and therefore signal intensity. Gaidatzis et al. 2009 show that by simply diluting an mRNA sample many statistically significant changes in alternative splicing appear when comparing the diluted sample to the original sample. This phenomenon is due to the different probesets decreasing in signal intensity at differing non-linear rates. They were able to predict this effect moderately well using a model using only features of sequence content.

COSIE takes a simple initial approach to compute splicing indices by taking the standard RMA summary for each exon and dividing it by the mean of all exons for that gene in that sample. After computing splicing indices, additional steps are taken to remove the previously mentioned bias that occurs when gene expression differs substantially between tissues.

In order to reduce potential biases resulting from differences in gene expression, Gaidatzis et al. 2009 fit a non-linear regression model to splicing indices as a function of aggregate gene expression across a tissue panel. Smooth systematic relationships between alternative splicing and gene expression are then subtracted out leaving residuals that should reflect only true changes in splicing. This observed bias in splicing indices seems to occur to a substantial degree only when changes in aggregate gene expression are on the order of multiple fold changes.

1.3.3 Methods for the analysis of alternative splicing

1.3.3.1 MADS

Microarray analysis of differential splicing (MADS) is a suite of methods for preprocessing, filtering, and performing inference on exon microarray data, with geneBASE being an important component of the preprocessing methods (Kapur et al. 2007; Kapur et al. 2008; Xing et al. 2008). In addition to geneBASE summarization, MADS uses a sophisticated background correction method as well as an algorithm to detect potential cross-hybridizing probes. Once pre-processing has been performed, the statistical methods used to detect differential splicing are relatively simple. Two-sample t-tests are conducted on splicing indices computed for each probe, not probe-set, in a gene and then p-values are combined using Fisher's method to create a single significance test for each gene.

1.3.3.2 ANOSVA

Analysis of Splice Variation (ANOSVA) uses a two-way ANOVA model for each gene to model log intensities of each probe in a gene (Cline et al. 2005). The linear model for the two-way ANOVA is given in Equation 1.6.

$$y_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + e_{ijk} \tag{1.6}$$

Here μ represents the baseline background intensity level for all probes, α_i represents the differing probe affinities, β_j represents the main effect for the covariate of interest, and γ_{ij} is a probe $\times$ effect interaction. ANOSVA assumes all effects are linear and additive, which is rather unrealistic. Also, individual probe signal intensities can be highly variable and have been shown to increase at different rates, even on the $\log_2$ scale (Gaidatzis et al. 2009).

1.3.3.3 FIRMA

Finding isoforms using robust multichip analysis, or FIRMA, is a method similar to ANOSVA that does not estimate the interaction γ_{ij} explicitly (Purdom et al. 2008). Instead, a main effects model is fit to $\log_2$ intensities for probe k of exon i in experiment j: y_{ijk}. Equation 1.7 gives the main effects model where c_j is the experiment effect and p_k is the probe effect. Residuals r_{ijk} are then computed from using the parameter estimates.

$$\begin{aligned} y_{ijk} &= c_j + p_k + e_{ijk} \\ r_{ijk} &= y_{ijk} - (\hat{c}_j + \hat{p}_k) \end{aligned} \tag{1.7}$$

The residual describes the discrepancy between the expected probe intensity under no alternative splicing and the observed probe intensity. A score statistic for testing for alternative splicing is then given in Equation 1.8 where the standard error s is calculated using the median absolute deviation (MAD) of the residuals (Lu, Schölkopf, and Zao 2011).

$$F_{ij} = median_{k \in exon j}(r_{ijk}/s) \tag{1.8}$$

1.4 Overview of RNA-Seq

1.4.1 Work flow

RNA-Seq (RNA-Sequencing) is a technology that uses next-generation sequencing to quantify the amount of RNA from a sample. Several of the preprocessing steps are similar to those of microarrays: coding RNA is extracted from a sample and reverse-transcribed into cDNA, amplified, and fragmented. However, once cDNA

is fragmented it is sequenced rather than hybridized to a microarray. While several protocols exist for next-generation sequencing, the end goal of all of methods is to obtain the actual nucleotide base sequences for these cDNA fragments. Generally, the whole fragments are not sequenced, but only the first 50 to 75 bases are sequenced depending on the protocol. This is usually enough to uniquely identify a large fraction of the cDNA fragments. Sequencing quality tends to decrease as more bases are added.

Once reads are sequenced, they are mapped to a reference genome using an alignment tool such as Bowtie (Langmead et al. 2009). From these reads, count data can then be obtained for genomic intervals by counting the number of reads falling in that interval. For RNA-Seq these intervals usually correspond to exons. Once raw count data is obtained, several methods exist for summarization and analysis.

1.4.2 Methods for summarization and analysis

While RNA-Seq ultimately produces count data, it is often summarized using a measure called "reads per kilobase per million," or RPKM (Mortazavi et al. 2008). RPKM scales read counts by the total number of reads in the sample as well as the size of the interval since larger exons should have more reads for the same amount of gene expression. Equation 1.9 gives the formula for RPKM.

$$\text{RPKM} = \frac{(\#\text{mapped reads})/(\text{length of transcript}/1000)}{\text{total reads in sample}/10^6} \tag{1.9}$$

The square root transformation, which is the variance stabilizing transformation for count data from a Poisson distribution, can be applied to RPKM which can then be treated as continuous for genes or exons with enough counts. Robinson and Oshlack 2010 showed that RPKM can be biased when a subset of genes are very

highly expressed in one tissue, but not another. Nevertheless, RPKM has become the standard way of summarizing RNA-Seq data.

Many statisticians have contended that since RNA-Seq data is in fact count data, is should be treated as such when performing modeling and significance testing. Therefore, methods using negative binomial generalized linear models have been developed to explicitly treat the data as counts (Robinson, McCarthy, and Smyth 2010; Anders and Huber 2010).

1.5 Overview of MBD-Seq

MBD-Seq is a cost-effective method for assaying DNA methylation on a genome-wide scale (Serre, Lee, and Ting 2010). MBD-Seq uses the methyl-CpG-binding domain (MBD) protein to extract regions of DNA containing methylated CpGs. Unlike the Illumina 450k array, MBD-Seq only obtains signals from methylated CpG sites and not unmethylated sites. After DNA fragments with methylated CpG sites are extracted, next generation sequencing is applied to map them to a reference genome as in RNA-Seq. MBD-Seq can be "tuned" to preferentially bind to areas with a given CpG density by altering the salt concentration of the buffer solution.

MBD-Seq is not a widely used assay, so few published methods exist for normalization and preprocessing (Chen et al. 2013a). For the purposes of analysis in later chapters we simply bin the data in windows of fixed width and adapt the RPKM measure from RNA-Seq.

1.6 Overview of genotyping arrays

Genotyping arrays are a type of microarray used to detect single nucleotide polymorphisms (SNPs) in DNA. A SNP is a variation in DNA sequence occurring at a single base. Different variations of a SNP are commonly referred to as alleles. Most

alleles have a common version called the major allele, and a less common version called the minor allele. These different alleles can affect phenotypes such as eye color or baldness or more complicated phenotypes such cancers or psychiatric disorders. Alleles are also used in DNA fingerprinting in forensic science.

Genotyping arrays function similarly to the Illumina 450k array, and are in essence a simpler version. For each SNP assayed by the array, a probe exists for each possible allele and a signal intensity is obtained for each. While a continuous measure is obtained for each allelle, it should hypothetically correspond to only one of three possibilities: the absence of the minor allele, presence of the minor allele on one chromosome, or the presence of the minor allele on both chromosomes. Therefore, the output of genotyping arrays for a given SNP is generally coded as an integer value 0, 1, or 2 corresponding to the three outcomes mentioned previously. These integer values are usually treated as ordinal rather than nominal when fitting statistical models.

1.7 Traditional approaches for integrative analysis

1.7.1 eQTL analysis

Quantitative trait loci (QTLs) are regions of DNA linked to genes associated with a quantitative trait. Traditionally, quantitative traits have been considered to be phenotypes such as height, blood pressure, or IQ that take on a continuous distribution. These kinds of quantitative traits are often complex and can be influenced by several genes. Studies of quantitative trait loci existed before the era of genome-wide association studies (GWAS), but assayed much smaller sets of candidate alleles (Plomin et al. 1994).

In the post-GWAS era, it is now possible to assess millions of SNPs simulta-

neously for an individual. Additionally, genomic measures can now be considered as quantitative traits. Expression quantitative trait loci (eQTLs) and methylation quantiative trait loci (mQTLs) are two examples of these new genomic QTLs (Gibbs et al. 2010). Studies of genomics QTLs involve integrating multiple types of genomic data generated by different microarrays or sequencing methods. Integrating genomic data presents several challenges, both statistical and bioinformatic.

There are various approaches to eQTL analysis. Most eQTL studies perform separate testing for all possible transcript-SNP pairs using standard linear regression or ANOVA models. For this reason, eQTL studies can be severely underpowered. A procedure for controlling false-discovery rate, such as Benjamini and Hochberg 1995, is then used to call significant eQTLs. eQTLs can be categorized into two major types: *cis*-acting and *trans*-acting. *Cis*-acting eQTLs are located within, or very close to the gene whose expression the are correlated with. *Trans*-acting eQTLs are distal SNPs affecting expression that may even be on different chromosomes.

1.7.2 Gene expression and promoter methylation

Integrating gene expression and DNA methylation has established a relationship between DNA methylation in promoter regions and gene expression in cancer studies (Baylin et al. 2001). Unlike eQTL analyses, integrating DNA methylation is more targeted and tests are conducted on a gene-by-gene basis rather than using all pair-wise combinations of expression and methylation measures. Methylation may be considered as a binary variable in some situations, but is generally treated as continuous.

Therefore, the most straightforward approach to integration is to use a Pearson's correlation coefficient for methylation/expression pairs (Warden et al. 2013). However, recent studies have demonstrated a relationship between non-promoter methy-

lation and exon inclusion (Maunakea et al. 2013). New microarray and sequencing technologies now make exploration of these kinds of relationships in large studies possible. Since most genes have multiple exons and CpG sites, alternative splicing and DNA methylation now both become multivariate continuous data and more sophisticated techniques are needed for integration.

1.8 Summary

In this section we have introduced the necessary concepts and terminology to motivate following chapters. Microarray design and summarization methods have been introduced for the Illumina 450k array. In Chapter 2, a novel robust normalization method for the 450k array will be introduced and compared to other popular normalization methods on several criteria. In Chapter 3, a novel method for genomic data integration will be introduced as a way to perform multivariate data analysis when $p > n$ with specific focus on integrating genic DNA methylation and exon inclusion. This method will then be applied in Chapter 4 to a set of developmental brain samples. In Chapter 5, a similar integrative analysis will be performed integrating gene expression and DNA methylation using brain samples taken from schizophrenic, bipolar, and neurotypical control patients.

CHAPTER 2

NORMALIZATION AND QUALITY CONTROL FOR DNA METHYLATION ARRAYS

2.1 Overview of normalization methods for 450k array

In recent years, the number of normalization methods for the Illumina 450k array has grown rapidly, and now a multitude of normalization methods and accompanying pipelines and R packages exist. Some of these methods normalize within arrays to account for the complex array design, while others focus on normalization between arrays to account for technical artifact and batch effects. Different normalization methods also operate on different levels of data summarization. Some methods require the summary level β-values, while other require the signal intensities. Some methods specifically require the raw signal .idat files which contain additional signal information that is not used in the standard Illumina summarization.

These different levels of data summarization can be problematic when choosing a normalization method. Most Gene Expression Omnibus (GEO) data sets provide raw signal intensities along with summarized β-values, but do not provide .idat files. This has reduced the pool of candidate data sets on which we are able to compare between-array normalization methods in later sections.

In the following section we provide a brief overview of popular methods in the literature. We give each method an intuitive conceptual introduction and highlight their strengths and potential shortcomings. Some of these shortcomings, particularly for quantile normalization, will be important later when motivating our new normalization method.

2.1.1 Within-array methods

As previously mentioned, there are two distinct bead types on the 450k array that have substantially differing signal characteristics as well as distribution throughout the genome. The type I beads are generally thought of as producing higher quality signals and are therefore used as a reference, or gold standard, for normalizing the type II beads in the following methods. Bibikova et al. 2011 observed that type II beads have a more compressed dynamic range than the type I probes. Teschendorff et al. 2012 showed that this compressed range in type II can result in a relative enrichment of type I beads to type II in when performing significance testing and sample clustering.

All three of the following methods attempt to make the data from type II beads look more like that from the type I. This task is complicated by the fact that the majority of type I beads have sequences lying in promoter regions, whereas type II beads are distributed throughout locations in the gene, which results in the two bead types having different overall signal distributions. Each of the following methods has a different way of normalizing the type II relative to type I, while trying to address the confounding issue of distribution of genomic location.

2.1.1.1 Peak-Based Correction (PBC)

Peak-based correction is a method that aligns the upper and lower peaks of type II beads with those of type I (Dedeurwaerder et al. 2011). This is accomplished by first computing summary β-values and transforming them into M-values using the relation: M-value $= \log_2(\beta\text{-value}/(1 - \beta\text{-value}))$. Next, a kernel density estimator is used to detect the upper and lower peaks for the type I and type II beads. Separate scaling factors are then applied to the M-values above and below zero such that

the type II peaks align with the type I peaks. These adjusted M-values are then transformed back into β-values for further analysis. The main shortcoming of this method is that it assumes a bi-modal methylation distribution with two distinct peaks. In most healthy adult tissues this is usually true, but it may not be the case for cancer samples or tissue that is a mixture of differentiated and non-differentiated cells. Samples may have more than two modes, or have wider, less-distinct peaks that may be difficult to align accurately.

2.1.1.2 Beta Mixture Quantile Normalization (BMIQ)

BMIQ performs a sophisticated quantile normalization procedure on the summary β-values by fitting a mixture of beta probability distributions (Teschendorff et al. 2012). Like peak-based correction, BMIQ uses the type I beads as a reference and normalizes the type II probes with respect to them. Rather than using a scaling factor to align peaks, BMIQ performs a quantile normalization procedure using the results of a three-state beta-mixture model that assigns CpGs as being either unmethylated, hemi-methylated, or fully methylated. After the three-state model is fit, each CpG is assigned to the most likely state. New values for the type II probes are then determined by assigning them the beta-distribution quantile from the type I density corresponding to their assignment probabilities determined from the original type II density.

BMIQ explicitly assumes that methylation values take only three possibly true underlying states. In the case of complex tissue where a β-value of 0.4 results from only 40 percent of cells being methylated at a locus, this assumption is invalid. Another weakness of both PBC and BMIQ is that they operate on the level of the β-value summary measure, and are unable to directly adjust signal intensities at a lower level before summarization.

2.1.1.3 Subset-quantile Within Array Normalization (SWAN)

SWAN performs a subset quantile normalization approach on signal intensities, rather than β-values, to make type II signals look more like type I (Maksimovic, Gordon, and Oshlack 2012). Since there are differing numbers of type I and type II probes, an average quantile distribution is first determined using a randomly selected subset of type II probes. This distribution is then quantile normalized to be identical with the type I distribution, with remaining probes adjusted by linearly interpolating the quantile distribution. This method is stratified by CpG content which is used as a proxy for biologically similar genomic regions. In practice, the adjustments made by SWAN are rather modest.

2.1.2 Between-array methods

The goal of between-array normalization is to remove artifact from signal intensities while preserving the biological signal. Several between-array methods for the 450k array have been recently developed. The complicated array design has made widely used general methods for microarray normalization, such as quantile normalization, not easily adaptable. The following between-array methods each have their own way of normalizing between arrays, while accounting for this complex design.

2.1.2.1 Subset Quantile Normalization (SQN)

Subset quantile normalization is an adaptation of the standard quantile normalization as performed in RMA (Touleimat and Tost 2012; Irizarry et al. 2003). In some ways, it is an extension of SWAN. The type I beads are used as anchors to create an average quantile distribution for several biologically distinct strata taken from the 450k array annotation file. Then both type I and type II beads are normalized with re-

spect to these average distributions using a standard quantile normalization approach on both the red and green channels. Once the stratified quantile normalization has been performed on the signals, normalized β-values are computed.

A criticism of SQN is that the fundamental assumption that all samples should have the same overall distribution, even when stratified by genomic location, can be invalid. It may be close to true for samples of healthy tissue, but it can fail for samples with aberrant methylation, or a set of samples with substantially varying cell type compositions. When this assumption fails, false apparent differences between groups can be created that may even be reproducible. More attention will be given to this phenomenon in following sections.

2.1.2.2 Normal-Exponential Using Out-of-Band Probes (Noob)

Noob is a background correction method that fits the standard normal-exponential model used by RMA, where observed signals are modeled as a convolution of a normally distributed background and exponentially distributed true signal (Irizarry et al. 2003). While a few hundred background probes exist for the 450k array to estimate parameters for the background normal density, fitting of the normal-exponential model is greatly enhanced by the use of "out-of-band probes" (Timothy J. Triche et al. 2013). These out-of-band probes are actually the signal intensities from the unused color channel of type I probes. Measures from the unused channels serve as additional measures of non-specific background hybridization, effectively increasing the number of background control probes from roughly 600 to 135,000. Noob requires the .idat signal intensity files to perform normalization, which are often unavailable as public data sets from websites such as GEO.

2.1.2.3 Functional Normalization (Funnorm)

Functional normalization uses the same principle as SQN, by using a quantile-based normalization approach stratified by genomic location (Fortin et al. 2014). However, rather than applying quantile normalization, a quantile regression method is used. First, principal component analysis is used to obtain summary measures from background and out-of-band probes. Then, the distribution quantiles are regressed against the first two principal components using a simple linear model. Since these background and out-of-band probes should not contain biologically relevant information, model fits should only be removing variation due to artifact. Quantile normalization can be seen as a special case of functional normalization that fits and subtracts out a saturated ANOVA model. Functional normalization may suffer from some of the same issues as subset quantile normalization, but they should be less severe.

2.2 Analysis of complex tissue using the 450k array

2.2.1 Complex tissues are a mixture of cell types

The human body is composed of many types of tissues such as muscle, skin, liver, and brain. Some of these tissues, such as skin and muscle, are composed mostly of a single cell type with a common origin. Therefore, we can be relatively confident that observed differences in these tissues are due to changes in methylation within the single given cell type. Even if the observed change is slight, we can be somewhat confident that some proportion of the cells are likely having a real change in methylation.

Other tissues such as brain, liver, and blood are made up of multiple different cell types with distinct methylation profiles. The brain is composed of a mixture of

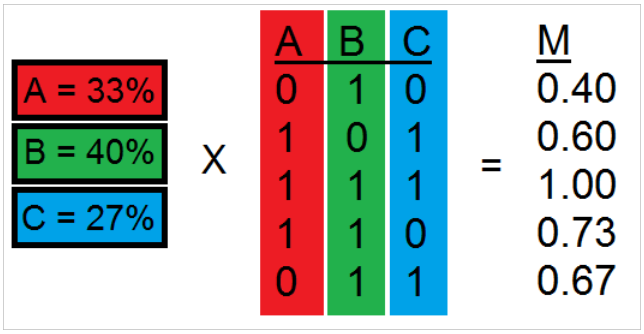

Fig. 1. An idealized example of observed intermediate β-values. An observed methylation profile of intermediate β-values M is a linear combination of β-value methylation profiles from cell types A, B, and C which are either completely methylated or unmethylated at each of five loci.

neurons, astrocytes, oligodendrocytes, and microglia. In blood, there are multiple different types of immune cells. In complex tissue, changes in the proportions of different cell types between samples can produce significant differences when no actual differential methylation within cell types is occurring. In fact, a meta-analysis of several studies using peripheral blood showed that the majority of age-related findings were in fact due to differences in cell proportions (Jaffe and Irizarry 2014).

Figure 1 gives an idealized example of a complex tissue composed of three different cell types. The three pure cell methylation profiles, given in the middle, are mixed in different proportions given on the left. The resulting β-values on the right can then take on intermediate values. More importantly, both changes within cell types as well as differences in cell proportions will produce slight changes in methylation. Changes in cell proportions will however, produce many slight changes on a global scale. Methods have been developed to estimate relative cell proportions when isolated cell methylation profiles exist (Houseman et al. 2012). Isolated cell type profiles have been obtained for both blood and brain for the 450k array using FACS (Reinius et al. 2012; Kozlenkov et al. 2013).

2.2.2 Addressing differences in cell type proportions

A method for estimating and adjusting for differences in cell type mixtures in complex tissue using isolated methylation profiles has recently been developed (Houseman et al. 2012). While originally developed for the Illumina 27k array, it has been adapted to the Illumina 450k array (Jaffe and Irizarry 2014). The method first applies a quadratic programming routine to obtain estimates of cell proportions that are constrained to sum to 1. Once these estimates are obtained, a double-bootstrap procedure is used to obtain standard errors for the estimates. Predicted methylation values from the model can then be subtracted out from the original data to produce mixture-adjusted residuals. Other more sophisticated models for incorporating estimated cell type proportions exist, but do not directly lend themselves to data integration (Guintivano, Aryee, and Kaminsky 2013). Significant differences in these residuals can then be attributed to real differences methylation, although we cannot say for sure in which cell type. A brief development of the method follows below.

Let $\boldsymbol{Y}_{0h}$ be an $m \times 1$ vector of methylation assay values from a purified cell type with the qualitative characterization given by a $d_0 \times 1$ covariate vector $\boldsymbol{w}_h$ which is generally given as a set of indicator variables for cell type. Here, $h \in \{1, ..., n_0\}$ where n_0 is the number of specimens and m corresponds to the number of CpG sites on the DNA methylation array. Then let $\boldsymbol{Y}_{1i}$ be an $m \times 1$ vector of the same CpG sites in the same order, but assayed from a sample that is a mixture of cells. Here $i \in \{1, ..., , n_1\}$ where n_1 is the number of target specimens. Let $\boldsymbol{z}_{1i}$ be $d_1 \times 1$ covariate vector representing phenotypic information. We can then posit the two following linear models describing the purified cell types and mixed samples, respectively, in Equation 2.1.

$$\begin{aligned} \boldsymbol{Y}_{0h} &= \boldsymbol{B}_0 \boldsymbol{w}_{0h} + \boldsymbol{e}_{0h} \\ \boldsymbol{Y}_{1i} &= \boldsymbol{B}_1 \boldsymbol{z}_{1i} + \boldsymbol{e}_{1i} \end{aligned} \tag{2.1}$$

We can then posit the following surrogacy relation between the two models in Equation 2.2.

$$\boldsymbol{B}_1 = \mathbf{1}_m \gamma_0^T + \boldsymbol{B_0}\boldsymbol{\Gamma} + \boldsymbol{U} \tag{2.2}$$

Here $\boldsymbol{\Gamma}$ is a $d_0 \times d_1$ matrix summarizing associations between the rows of $\boldsymbol{B}_{0j}$ and $\boldsymbol{B}_{1i}$ and $\boldsymbol{U}$ is a matrix of errors. Substituting Equation 2.2 into the second part of Equation 2.1 yields the following in Equation 2.3.

$$\boldsymbol{Y}_{1i} = \sum_{l=0}^{d_0} \boldsymbol{b}_{0l}(\gamma_l^T \boldsymbol{z}_{1i}) + (1_m \gamma_0^T + \boldsymbol{U})\boldsymbol{z}_{1i} + \boldsymbol{e}_{1i} \tag{2.3}$$

Estimation of $\boldsymbol{B}_0$ and $\boldsymbol{B}_1$ procedes by applying an appropriate linear or mixed effects linear model. Estimates of γ_0 and $\boldsymbol{\Gamma}$ are then obtained by projecting $\hat{\boldsymbol{B}}_1$ onto the column space of $\widetilde{\boldsymbol{B}}_0 = (1_m, \hat{\boldsymbol{B}}_0)$ using a constrained linear programming routine.

The mixture coefficients $\omega_l^{(\boldsymbol{z})}$ can then be recovered from $\boldsymbol{\Gamma}$ by $\omega_l^{(\boldsymbol{z})} = \gamma_l^T \boldsymbol{z}_{1i}$. To impart a biological interpretation, we can say that the observed methylation profiles arise as a mixture of cell types whose isolated methylation profiles have coefficients given by $\boldsymbol{b}_{0l}$ in proportions given by $\omega_l^{(\boldsymbol{z})}$ and some residual mixture of unobserved cell type proportions and true methylation differences $\xi^{(\boldsymbol{z})}$. Equation 2.4 gives the relationship explicitly below.

$$E(\boldsymbol{Y}_{1i} | \boldsymbol{z}_{1i} = \boldsymbol{z}) = \xi^{\boldsymbol{z}} + \sum_{l=1}^{d_0} \boldsymbol{b}_{0l} \omega_l^{(\boldsymbol{z})} \tag{2.4}$$

A straightforward method for downstream analysis is using the residuals $\xi^{\boldsymbol{z}}$ as a

measure of remaining real methylation changes after accounting for differences in cell proportions. Remaining significant findings in these residuals may then be attributed to real methylation differences although the specific tissue or tissues where the changes are occuring is not specified.

2.2.3 Complex tissue and microarray normalization

Complex tissues also pose a problem for many common normalization methods. A common assumption of many normalization methods for genomic data is that the majority of observations should be similar between samples. This assumption is necessary in order to have additional points of reference for comparison between samples aside from the background probes. Quantile normalization goes so far as to enforce the empirical distributions of samples to be identical (Touleimat and Tost 2012; Bolstad et al. 2003). This assumption is often untrue, but is particularly problematic in the case of complex tissues where differences in cell proportions can result in global changes in the overall methylation distribution.

Quantile normalization of samples with different β-value distributions can lead to *reproducible* false differences. Figure 2 shows density plots for average β-values from 69 technical replicates of a liver sample and 55 placenta (Aryee et al. 2014). After quantile normalization, intermediate β-values in placenta must be increased and β-values closer to one in liver must be decreased in order to match the two distributions. This warping not only changes the resulting mean methylation profiles, but also their relationship to each other. A CpG site that has a mean β-value $= 0.5$ in both tissues may have a statistically significant mean difference after quantile normalization. If we perform this analysis on independent subsets of the data, the same result will occur. This phenomenon could cause two completely separate and independent microarray studies using the same study design to *replicate* false discoveries!

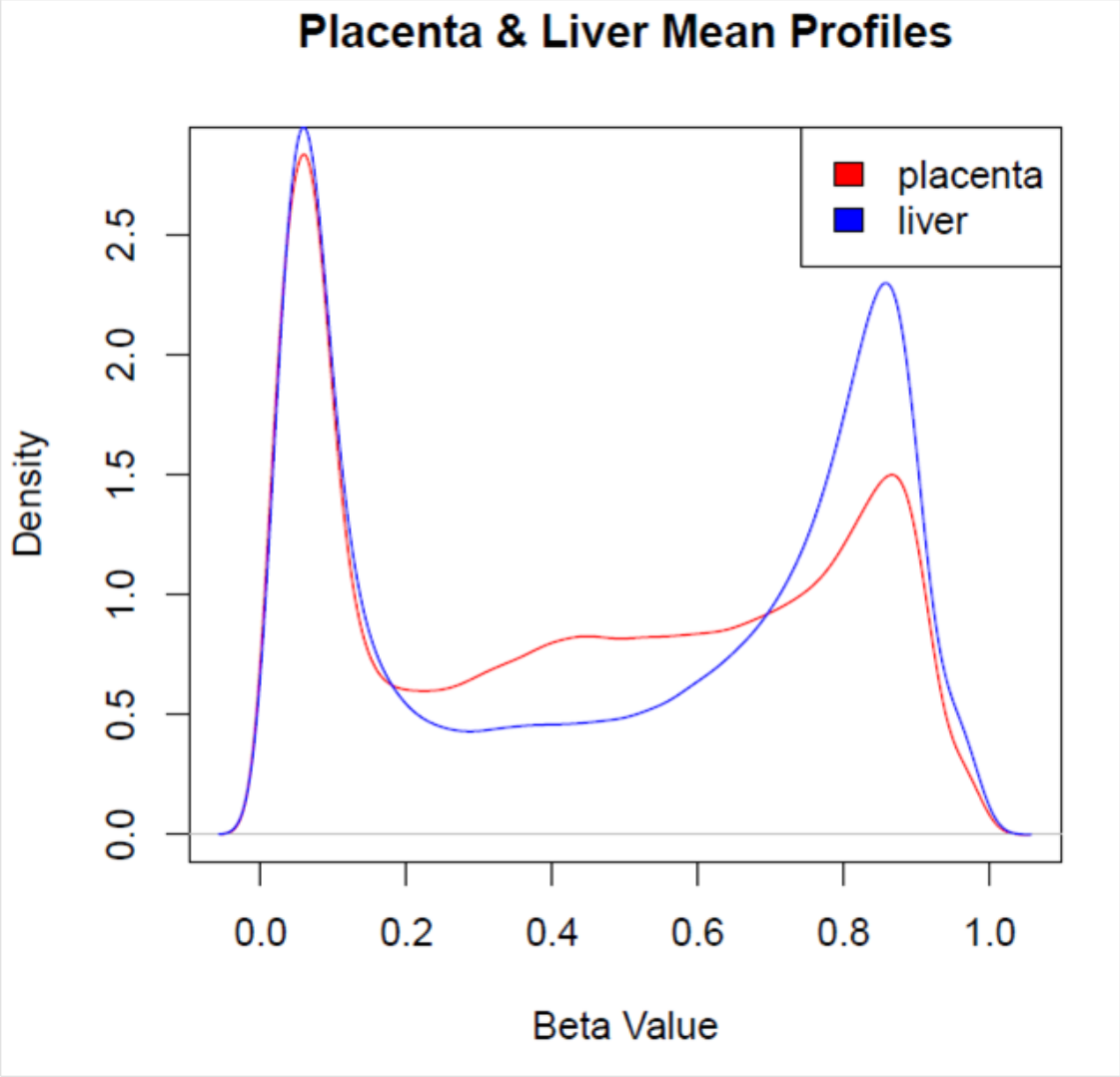

Fig. 2. Average methylation profiles for 69 technical replicates of liver and 55 technical replicates of placenta

In order to determine the extent to which quantile normalization warps mean β-value distributions of these two tissue types, we plot mean differences after normalization against each other. Ideally, a normalization method shouldn't change the mean profiles of technical replicates by much in any direction, but if it does it shouldn't do so in opposing directions in different tissues. Equation 2.5 gives the formula for computing differences in mean β-values (Δ_{ik}) for technical replicates $j \in \{1, ..., J\}$ of tissue type k at probe i.

$$\Delta_{ik} = \frac{\sum_{j=1}^{J} \beta_{ijk}^{Norm} - \sum_{j=1}^{J} \beta_{ijk}^{Raw}}{J} \tag{2.5}$$

Quantile normalization was applied to all samples in aggregate. While samples can be normalized separately by tissue type, which will avoid the problem in this scenario, this approach is not a cure-all and should ideally not be necessary. In the case of confounding continuous covariates such as differing cell proportions over age, a stratified normalization approach is not directly applicable.

Figure 3 plots changes in mean β-values in placenta after quantile normalization against changes in mean β-values in liver after quantile normalization for CpGs that were only significantly different between the two tissues after quantile normalization. Significant differences were determined using a two-sample t-test on β-values from each CpG site and controlling FDR=0.1 using the Benjamini-Hochberg method. Points are colored by global average β-value before normalization in the left panel, and by probe type in the right panel. Again, changes in mean methylation profiles after normalization should be minimal. However, we observe changes in β-values that can be almost as big as $\Delta_{ik} = 0.3$. Points that are far from the $y = x$ line indicate CpG sites where mean differences between the two tissues change, sometimes almost as much as $|\Delta_{i1} - \Delta_{i2}| = 0.2$. We can see that type two probes are being adjust almost

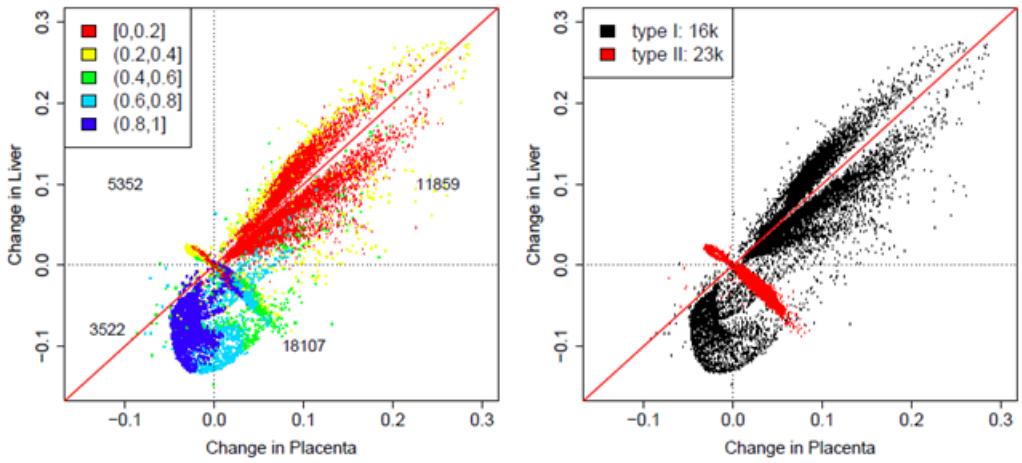

Fig. 3. Changes in average methylation profiles in liver and placenta after subset quantile normalization. Points in the left panel are colored by mean β-value. Intermediate beta values are more strongly affected. The number of points lying in each quadrant are given. The right panel is colored by probe type on the array. Many Type I probes and nearly all Type II probes are adjusted in opposing directions, creating false differences that did not exist before normalization.

exclusively in opposing directions between the two tissues, creating substantial mean differences that were not present before.

This is perhaps an extreme example of what can happen if distributional differences are ignored during normalization. This warping effect can be particularly problematic in methylation data where the data have a bi-modal distribution and differences in heights of modes between samples can have global effects on intermediate methylation values. Ideally, a normalization method should be robust to these kinds of distributional differences, whether they arise from differences in cell proportions in complex tissues or some other mechanism. Our proposed normalization method is designed with this exact goal of being robust to distributional differences.

2.3 Normalization using local regression on empirical controls

We propose a normalization method "Flexible local Regression on Empirically Selected COntrol probes," or fresco, as a supervised model-based normalization method for the Illumina 450k array. fresco was developed with the goal of creating a method that is robust to samples with varying methylation profiles as in Figure 2. The method uses a stable subset of CpGs, called empirical control probes, to fit a non-linear local regression hyper-surface to model signal intensities as a function of known covariates. fresco adjusts probe signal intensities, but does not require .idat files, so GEO data sets providing raw signals can be used. Using raw signals allows for normalization of red and green channels separately, which have been shown to have differing properties (Bibikova et al. 2011). The method has proceeds in three steps which are detailed below.

2.3.1 Selection and filtering of empirical controls

Empirically selected control probes are CpG sites taken from regions of the genome that should generally be consistent between all samples. The concept of empirical controls has been used by Gagnon-Bartsch and Speed 2011 for gene expression microarrays, but using a different statistical model. Unlike negative control probes on a microarray, the full set of empirical control probes employed by our method is representative of the entire range of variation in signal intensities. This is accomplished by having three subsets of empirical controls: methylated, unmethylated, and hemi-methylated.

It is has been generally observed that active genes have mostly unmethylated promoters and mostly methylated gene bodies. The so-called housekeeping genes are prime candidates for genes that should be active across all cell-types. Eisenberg and

Levanon 2013 identified a list of 3804 housekeeping genes using RNA-seq. While RNA gene expression patterns may be less consistent across samples due to a myriad of other factors such as lncRNAs, RNA binding proteins, RNA degradation, or distal enhancer activity, their methylation should be relatively more stable. From these housekeeping genes we obtain a set of negative controls from promoters that have beta values near zero, and a set of positive controls that have beta values near one.

In order to have a truly representative subset of empirical control CpGs, we still need to have a set of CpGs that cover the intermediate range of methylation values. To accomplish this we use the set of known imprinted genes whose promoters should be methylated on one chromosome and unmethylated on the other, therefore producing intermediate beta values (Jirtle 2012). Once the full set of empirical controls are obtained, an additional quality control step is taken.

Once candidate empirical control probes are selected, a filtering and quality control check is performed to ensure that they are indeed stable. First, we filter out probes containing known SNPs within their target sequences as well as probes that have been shown to cross-hybridize with sites on sex chromosomes which can lead to false autosomal gender differences (Chen et al. 2013b). After this initial filtering, the empirical control CpGs are then filtered for stability across a tissue panel composed of healthy adult tissues: brain, blood, and liver (Reinius et al. 2012; Aryee et al. 2014). Probes with standard deviations falling below a cut-off of $\sigma = 0.1$ are then included in the final set of empirical controls. Figure 4 illustrates the empirical control filtering process. This same filtering step is available for new data sets as a part of the normalization function.

Once we have selected the final set of empirical controls, we want to ensure that they are indeed representative of the overall range of possible signal intensities. Figure 5 shows the range of coverage for empirical control probes for both type I and

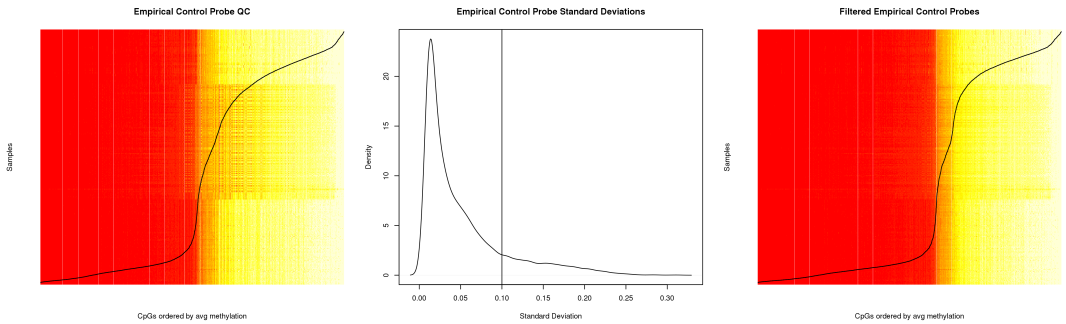

Fig. 4. Filtering empirical controls across a tissue panel by standard deviation. The left-most panel gives a heat map of β-values where each column is a CpG site, and each row corresponds to a sample. CpGs are sorted according to average methylation, with average methylation level given by the black line on a scale from 0 to 1. The middle panel gives the density of standard deviations of β-values and the threshold used to discard empirical controls. The right-most panel is the same as the first panel with the more variable probes removed

type II probe signal intensities. For each set of probes $\log_2$(Unmethylated Signals) are plotted against $\log_2$(Methylated Signals). Additionally, $\log_2$ signal intensities are plotted against target GC content, a covariate of interest. Empirical control probes are colored by their type: Methylated, Unmethylated, and Hemi-methylated.

2.3.2 Alignment and scaling

Although β-values taken from empirical control probes are filtered to be similar, the distributions of signal intensities taken from these empirical controls can still differ substantially. The variability in these signal intensities is likely largely a function of technical artifact rather than true biological signal. Therefore, we perform our normalization method on signal intensities from these sets of empirical control probes and extend model fits to remaining probes.

After empirical control probes are selected, an initial alignment and scaling procedure is performed before fitting the local regression hyper-surface. Since the 450k

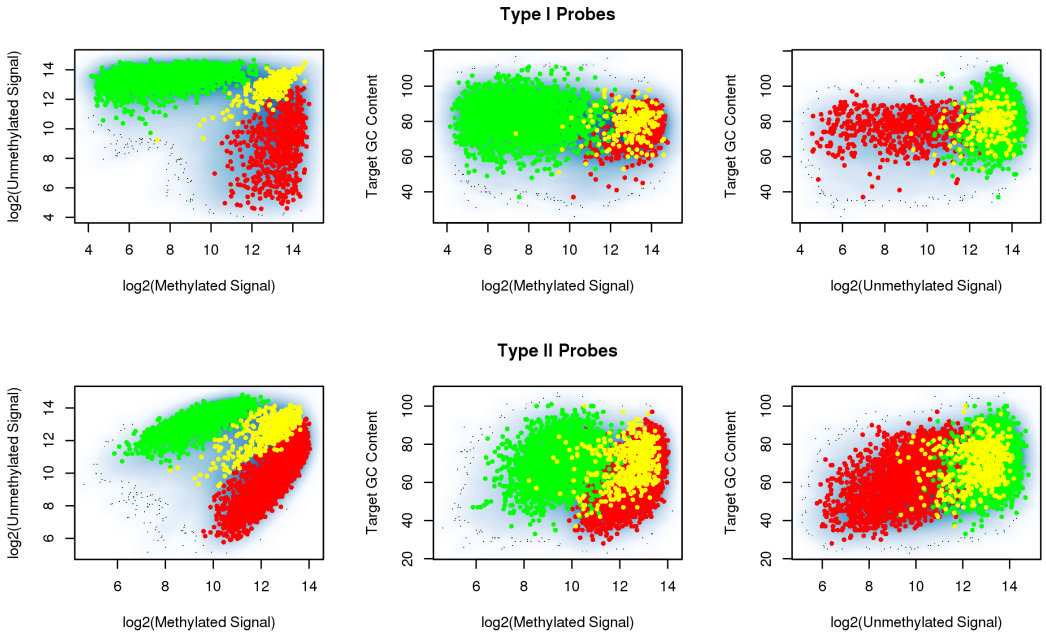

Fig. 5. Empirical controls span the range of microarray signal intensities and CG content. Green, yellow, and red points correspond to unmethylated, hemi-methylated, and methylated controls, respectively.

array contains two bead types, all subsequent normalization steps are stratified by type I and type II beads, essentially conducting two completely separate normalization procedures in parallel with each other. For the alignment step, a kernel density estimate is fit for the distribution of empirical control probes for each of the methylated and unmethylated control probe signal densities. The lower peaks of the densities are calculated by finding the max of each of the kernel densities and then subtracting it out so that they all share a common lower peak at zero.

Once signal densities are aligned by their lower peaks, a linear scaling factor is applied to minimize the difference between each sample's empirical control density and the average empirical control density. This is done by fitting an ordinary least squares zero-intercept linear model for each sample: one for each of the type I channels, and one for each of the type II channels. Equation 2.6 gives the formula for the linear model.

For each sample $j \in \{1, ..., J\}$ and for empirical control probes $i \in \{1, ..., I\}$, we model each empirical control profile Y_{ij} as a function of the average empirical control profile $Y_{i\cdot}$ using a zero-intercept linear model. A scaling factor is then estimated for the j^{th} sample as $1/\theta_j$. Equation 2.6 gives the resulting linear model.

$$Y_{ij} = \theta_j Y_{i\cdot} + \epsilon_{ij} \tag{2.6}$$

Once scaling factors are computed and applied, the average lower peak is added back in. If there are any negative values after alignment and scaling they are set to zero. Once alignment and scaling have been performed on the empirical control probes, the same process is applied to remaining probes using the empirical control model fits. Figure 6 gives an example of the alignment and scaling procedure for the type II unmethylated channel.

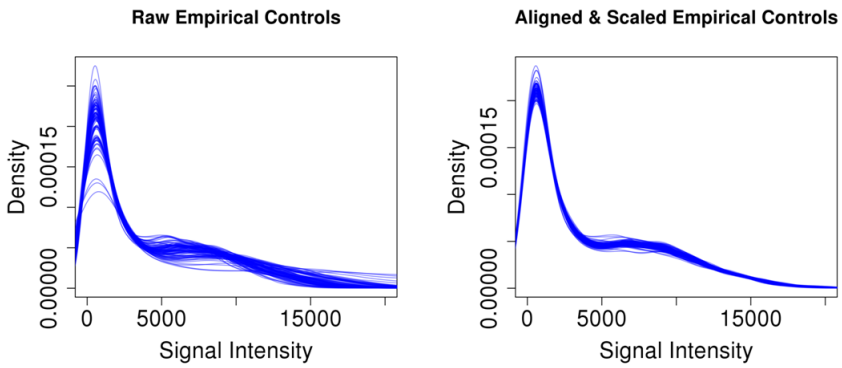

Fig. 6. Densities of signal intensities for unmethylated channel of type II probes before and after initial alignment and scaling. The left panel gives densities of raw signals. The right panel gives densities after samples are aligned by peaks and a linear scaling factor is applied.

2.3.3 Flexible local regression on technical covariates

Once signals have been aligned and scaled, a final step is performed by fitting a local polynomial regression hyper-surface to each sample to remove remaining technical artifacts. Local regression is a non-parametric regression technique that fits many linear models to local subsets of data whose sizes are determined by a span parameter λ (Cleveland, Grosse, and Shyu 1992). For each subset, observations are weighted by their proximity to the center of the subset using a kernel function, and a weighted linear model is fit. For the normalization method, surfaces are fit as a function of technical covariates which are probe-specific covariates thought to be representative of sources of technical bias. Local regression on technical covariates provides a general framework that can be adapted to other microarray or sequencing technologies. The loss function for weighted local polynomial regression for a single subset of data is given below in Equation 2.7.

$$\sum_{i=1}^{n}(Y_i - \beta \boldsymbol{x}_i - \gamma \boldsymbol{x}_i^2)^2 \omega(t_i) \tag{2.7}$$

We use a span of 15% and the tricube kernel given by Equation 2.9 when fitting the weighted local regression hyper-surface. Choice of span generally does not seem to make a large difference, but 15% seems appropriate in most cases. Smaller spans are also preferable because they result in loess fits that are less computationally intensive since fewer data points are used for each fit.

The value t_i in Equation 2.9 is a value between 0 and 1 that represents the scaled Euclidean distance of a point $\boldsymbol{x}_i$ from the center of the window. If there are $j \in \{1, ..., J\}$ technical covariates, then Equation 2.8 gives the formula for t_i where $\boldsymbol{x}^*$ contains the coordinates for the center of the window, $\boldsymbol{x}_i$ is the vector of technical covariates for data point i, and h is the window half-width.

$$t_i = \frac{\sqrt{\sum_{j=1}^{J}(x_{ij} - x_j^*)^2}}{h} \tag{2.8}$$

$$\omega(t_i) = \begin{cases} (1 - |t_i|^3)^3 & \text{if } |t_i| \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.9}$$

Technical covariates included in our model include probe target sequence GC content as well as average methylated and unmethylated signal intensities. The surface fitting step proceeds separately for each of the two channels within each probe type. First, signal intensities are transformed using $\log_2(y + 1)$ to reduce the skew in their distributions which should result in a more stable surface. Then a robust average methylation profile is computed for all probes using a 10% trimmed mean. Finally, the local regression hyper-surface is fit to deviations from the average methylation profile as a function of technical covariates (Equations 2.10 and 2.11). For either the

$\log_2$ methylated channel signal M, or $\log_2$ unmethylated channel U and probe i in sample j, deviations from the average intensity are modeled as a sample-specific local regression surface f_j that is a function of average $\log_2$ methylated signal ($\bar{M}$), average $\log_2$ unmethylated signal ($\bar{U}$), and target sequence GC content (GC).

$$U^*_{ij} = (U_{ij} - \bar{U}_i) - f_j(\bar{M}_i, \bar{U}_i, \mathrm{GC}_i) \tag{2.10}$$

$$M^*_{ij} = (M_{ij} - \bar{M}_i) - f_j(\bar{M}_i, \bar{U}_i, \mathrm{GC}_i) \tag{2.11}$$

Once the fitted surfaces have been subtracted out and the residual matrices $\boldsymbol{M}^*$ and $\boldsymbol{U}^*$ are computed, normalized signals are then computed by adding average signal profiles intensities back to obtain normalized signals on the $\log_2$ scale. These normalized $\log_2$ signals are then transformed back from the $\log_2$ scale to obtain normalized signal intensities which can be used to compute β-values. A formula for transforming the normalized residuals back into normalized β-values is given in Equation 2.12 where ϵ is a small offset suggested by Illumina to stablize β-values when both methylated and unmethylated signals are small.

$$\beta^*_{ij} = \frac{2^{M^*_{ij}+\bar{M}_i}}{2^{M^*_{ij}+\bar{M}_i} + 2^{U^*_{ij}+\bar{U}_i} + \epsilon} \tag{2.12}$$

2.4 Performance assessment

To assess the performance of the fresco normalization method, we compare it with other popular between-array normalization methods across several metrics. While the goal of within-array normalization methods is to reduce the enrichment bias of type I probes relative to type II, the goal of between-array normalization is to improve the biological signal-to-noise ratio. Since most normalization methods are performed

without incorporating information on phenotype (with Surrogate Variable Analysis being an exception), we believe improvement in signal-to-noise ratio should be a result of reduction in noise, rather than amplification of biological signal (Leek and Storey 2007).

It is also important that a normalization method does not over-fit or over-adjust in such a way that a substantial amount of true biological variability is tampered with or removed. As mentioned previously, our goal is to have an effective method that avoids over-fitting by using a very flexible model on a subset of data that should be stable between samples. In the following sections we detail the data sets being used, the metrics being used to assess performance, and results on the effects of normalization. All data sets are read in and preprocessed from the .idat files using the minfi package in R (Aryee et al. 2014).

2.4.1 Overview of data sets

The first data set from the BrainSpan Consortium contains 93 post-mortem human brain samples taken from 6 individuals. Each individual is sampled at sixteen brain regions. Of the original 96, three samples did not pass an initial quality control check and were discarded. Two samples were outliers and had low signal intensities for many probes. One cortical sample seemed to be mislabeled and clustered with the cerebellum samples, which have a very distinct methylation profile. Brains were sampled from eleven different cortical regions and 5 sub-cortical regions. Table I gives a summary of the distinct brain regions sampled and their abbreviations. Samples were randomized across eight batches of size twelve.

The second data set comes from a set of six peripheral blood samples taken from six healthy males (Reinius et al. 2012). This set of six samples is assayed several times after applying centrifugation and FACS in different combinations to isolate various

Table I. Brain regions assayed in BrainSpan data

Brain Region	**Abbreviation**	**Location**
Primary Auditory Cortex	A1C	Temporal Lobe
Amygdala	AMY	Sub-cortical
Cerebellum	CBC	Sub-cortical
Dorsolateral Prefrontal Cortex	DFC	Frontal Lobe
Hippocampus	HIP	Sub-cortical
Inferior Parietal Cortex	IPC	Parietal Lobe
Inferior Temporal Cortex	ITC	Temporal Lobe
Thalamus	MD	Sub-cortical
Primary Motor Cortex	M1C	Frontal Lobe
Medial Prefrontal Cortex	MFC	Frontal Lobe
Orbitofrontal Cortex	OFC	Frontal Lobe
Primary Somatosensory Cortex	S1C	Parietal Lobe
Superior Temporal Cortex	STC	Temporal Cortex
Striatum	STR	Sub-cortical
Primary Visual Cortex	V1C	Occipital Lobe
Ventral Frontal Cortex	VFC	Frontal Lobe

Table II. Sample types in Reinius blood data

Sample Type	**Preprocessing**
Whole Blood	None
Mononuclear Cells	Centrifugation
Granulocytes	Centrifugation
CD4+ T Cells	FACS on Mononuclear Cells
CD8+ T Cells	FACS on Mononuclear Cells
CD14+ Mononuclear Cells	FACS on Mononuclear Cells
CD19+ B cells	FACS on Mononuclear Cells
CD56+ Natural Killer Cells	FACS on Mononuclear Cells
Neutrophils	FACS on Granulocytes
Eosinophils	FACS on Granulocytes

cell sub-populations. Table II gives an overview of the sample types for each of the 6 samples and how they were obtained. There are sixty arrays in total. Samples were randomized across five batches of size twelve.

The third data set is a collection of 175 liver samples generated by the TCGA Research Network (**http://cancergenome.nih.gov**). Samples are a mixture of 50 healthy livers and 123 livers with hepatocellular carcinoma that come from multiple medical centers involved with TCGA. Table III gives details of the samples taken from each study. Samples are randomized over 21 batches, but three batches contain only HCC samples.

Table III. Overview of hepatocellular carcinoma samples from TCGA data set

Center	Number of Controls	Number of Cases
Alberta Health Services	2	17
Asterand	0	3
Christiana Care Health System	2	7
Fox Chase Cancer Center	0	1
ILS Bioservices	0	13
International Genomics Consortium	2	11
Mayo Clinic	28	47
Ontario Institute for Cancer Research	0	1
Saint Joseph's University	0	3
University of Florida	2	1
University of Minnesota	0	1
University of North Carolina	12	15
University of Pittsburgh	2	3

2.4.2 Methods for comparison

2.4.2.1 Reduction in batch effect

One intuitive method for assessing the effectiveness of a normalization method is to see how well it is able to reduce batch effects (Chen et al. 2011). Batch effects are significant differences among samples that occur across batches. If proper randomization and experimental design are performed, and biological factors of interest are mostly orthogonal to batch assignment, then batch effects are a good measure of technical variability. Methods have been developed to specifically target batch effects using empirical Bayes methods (Johnson and Li 2006). If it is the case that batches are confounded with phenotypes, then reduction in batch effect is not as easily interpreted as a reduction in technical variability. There is minimal confounding between batch and covariates of interest in the three data sets used for comparison

of normalization methods.

For assessing reduction of batch effects, a one-way ANOVA is fit to each CpG on the array with batch as a predictor. This then produces a distribution of p-values for all CpG sites. If no batch effects are present, the p-value distribution should be as close to uniform as possible. Deviations from the uniform density with an increased number of small p-values indicates presence of batch effects. Empirical cumulative p-value distributions are then obtained for the raw data and each of the normalization methods for visual comparison.

2.4.2.2 Increase in apparent significance

The goal of genomic microarray studies is usually to compare samples across some set of biological conditions. Therefore, it is desirable to have as much power as possible to detect these differences. However, since we don't know the true methylation states of assayed CpGs, we have no way of objectively measuring power. Due to the intricacy of some normalization methods, it is difficult to simulate a realistic scenario. We can instead take the increase in the number of CpG sites as an indirect measure of power, but this naïve approach is not soley sufficient, as will be detailed in the next section.

We can begin to justify this approach by claiming that under appropriate experimental design, technical artifacts, such as batch effects should be mostly orthogonal to biological signal. If framed in the context of a t-test or F-test, technical variability should be mostly contributing to the denominator, or error term of the test statistic. Therefore, an increase in the significant number of CpG sites when testing across covariates of interest should indicate a reduction in batch effects and also less easily characterized sources of technical variability that are contributing to the error term.

In order to assess increases in apparent significance, a similar approach is taken to the one for assessing batch effects. One-way ANOVAs are fit using β-values for

each CpG and p-values are computed for each site. P-values are then adjusted using the Benjamini-Hochberg procedure for controlling False Discovery Rates (Benjamini and Hochberg 1995). If one particular normalization method improves the signal-to-noise ratio better than another, then there should be more significant CpG sites after normalization for the same FDR. However, an increase in significance does not guarantee that the increase is due to a reduction in technical variability and more real differences are being discovered. As mentioned previously, quantile normalization methods can create reproducible false differences when comparing samples that have different overall distributions of beta values.

2.4.2.3 Sensitivity of methods to distributional differences

While an improved signal-to-noise ratio is the main goal of a between-array normalization, it is important to ensure that the resulting improvement is valid. The noise component of the signal to noise ratio in microarray experiments is generally not i.i.d. and has some kind of structure that can be attributed to technical artifact such as cross-hybridization, probe GC content, or spatial variability on the chip. The goal of normalization is to remove components of this structured noise. Aside from quantile normalization, normalization methods generally use information that should be independent of biology so as to not tamper with true biological signal. Funnorm and Noob use the out-of-band probes. Our method uses technical covariates. Therefore, a normalization that has a substantial effect on biological effect sizes should be treated as suspect, especially if it is affecting a large number of CpG sites. In order to assess whether increases in signficiance are due to a reduction in noise, or an increase in effect size, we create what will be referred to as composite F-statistics. Equation 2.13 shows the standard formula for an F-statistic from a one-way ANOVA.

$$F_{pq} = \frac{\frac{\sum_{i=1}^{K} n_i(\bar{Y}_{i\cdot} - \bar{Y})}{K-1}}{\frac{\sum_{i=1}^{K}\sum_{j=1}^{n_i}(\bar{Y}_{ij} - \bar{Y}_{i\cdot})}{N-K}} = \frac{\frac{SSR}{K-1}}{\frac{SSE}{N-K}} = \frac{MSR}{MSE} \tag{2.13}$$

From looking at Equation 2.13, it is clear that there are two ways an F-statistic can become larger, and therefore more significant: by an increase in the MSR (effect size), or a decrease in the MSE (error noise). By picking apart these two pieces, we can determine if an increase in effect size or a decrease in error noise is what is driving the increase in significance after normalization.

To accomplish this, we create two composite F-scores using different components of the F-statistic from before and after normalization. Let SSR_{Raw} and SSE_{Raw} be the numerator and denominator, respectively, of the F-statistic (given in Equation 2.13) computed from the raw data. Similarly, let SSR_{Norm} and SSE_{Norm} be the numerator and denominator of the test statistic computed from the same set of data after performing a normalization. We can then define two composite F-statistics F_{ES} and F_{Err} given in Equation 2.14. F_{Err} reflects the effect of normalization on significance by reducing the error term, which we can think of as being reflective of removing technical artifact. F_{ES} reflects the effect of normalization on signifiance by increasing the observed effect size. Systematic increases in F_{ES} are probably due to over-fitting of the normalization procedure rather than true increases in the biological component of signals.

$$\text{F}_{\text{ES}} = \frac{SSR_{\text{Norm}}}{SSE_{\text{Raw}}} \qquad \text{F}_{\text{Err}} = \frac{SSR_{\text{Raw}}}{SSE_{\text{Norm}}} \tag{2.14}$$

In order to assess if normalization procedures are increasing apparent significance by reducing the error variability or increasing effect sizes, we plot results from the two sets of composite test statistics against results from the original statistics. Specifically, we plot the $-log_{10}$(p-values) from the two composite test statistics against each other.

Resulting increases in significance in the composite F-statistics will result in points falling above the $y = x$ line.

2.4.3 Results

2.4.3.1 BrainSpan

The BrainSpan data consists of 93 samples randomized across 12 batches. While batch effects should be mostly orthogonal to covariates of interest, it is still desirable to mitigate batch effects to minimize the error variance. Figure 7 gives empirical cumulative p-value distributions (ECDFs) from the one-way ANOVA testing for batch effect.

Interestingly, Funnorm seems to substantially increase batch effects relative to the raw data. Noob, and the various versions of our normalization procedure perform similarly. SQN and the fresco using a 15% span provide the greatest reduction in batch effects.

The ultimate goal of normalization is not to specifically remove batch effects, but to improve the signal-to-noise ratio and overall data quality. Figure 8 gives a plot of proportion of CpGs declared significant at a given FDR where p-values were adjusted using the Benjamini-Hochberg method (Benjamini and Hochberg 1995). We can see that most normalization methods perform similarly, calling roughly 20% of CpGs significantly different across brain regions at an FDR = 0.1. All methods seem to perform slightly better than Funnorm.

Figure 9 gives scatter plots of $-\log_{10}$(p-values) from composite F-scores plotted against the original $-\log_{10}$(p-values) testing for differences in brain region. The top row of figures plots F_{Err} on the Y-axis and the bottom row plots F_{ES}.

From looking at the scatter plots of F_{Err}, it appears that quantile normalization

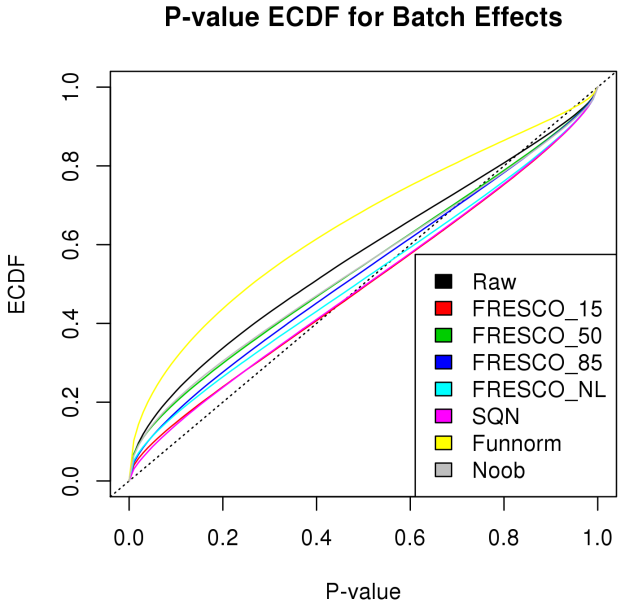

Fig. 7. Empirical cumulative p-value distributions from one-way ANOVAs for batch effect in the BrainSpan data.

provides the greatest increase in power through reduction in the error variance term of the F-statistics from the one-way ANOVA for brain region. Funnorm provides a weaker, but similar reduction in the error variance. Our method seems to provide a more moderate decrease in error variance.

The scatter plots of F_{ES} reveal that much of the observed increase in significance from quantile normalization in Figure 8 is likely due to an increase in the numerator of the F-statistics from the warping of overall β-value distributions. Surprisingly, none of the methods, even Noob which is a background correction, are completely immune to this phenomenon. Quantile normalization, and Funnorm to a lesser degree, appear to reduce larger effect sizes. When incorporating the local regression hyper-surface, our method also seems to suffer from over-fitting, in spite of the usage of empirical controls.

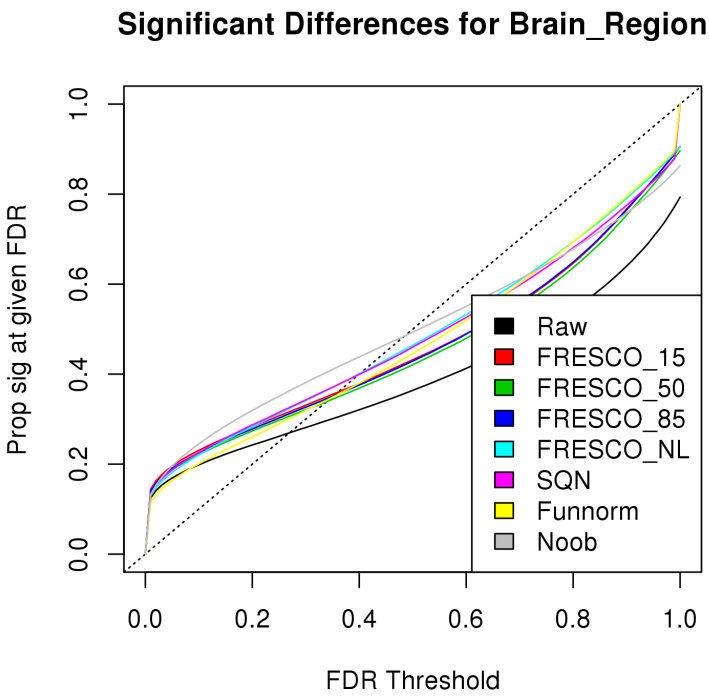

Fig. 8. Proportion of CpGs called significant for different FDR thresholds in the BrainSpan data. Quantile normalization creates more apparent significance relative to other methods.

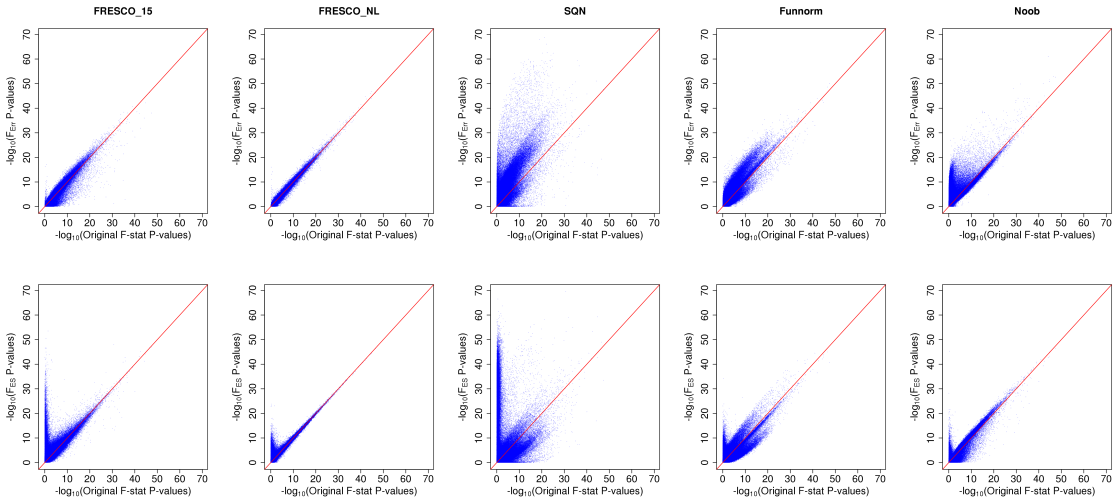

Fig. 9. Scatter plots of $-\log_{10}$(p-values) from composite F-statistics for regional differences the BrainSpan data. Subset quantile normalization can create apparent significance by increasing effect sizes alone.

2.4.3.2 Reinius flow-sorted blood

Figure 10 gives empirical cumulative p-value distributions for batch effects in the Reinius flow-sorted blood data which was mostly randomized across five batches. The fifth batch consists exclusively of granulocyte samples and is excluded when assessing reduction in batch effects due to confounding. Interestingly, Noob actually seems to substantially increase the significance of batch effects relative to the raw data. fresco methods using the local regression surface and SQN provide the greatest reduction in batch effects, with smaller loess spans providing a greater reduction.

Figure 11 gives a plot of the proportion of CpGs declared significant at a given FDR where p-values were adjusted using the Benjamini-Hochberg method (Benjamini and Hochberg 1995). Noob appears to clearly outperform all methods in this context. SQN offers the second best improvement in power, but as we will see again, much of this increase in power is likely due to bias incurred in CpGs with intermediate methylation values.

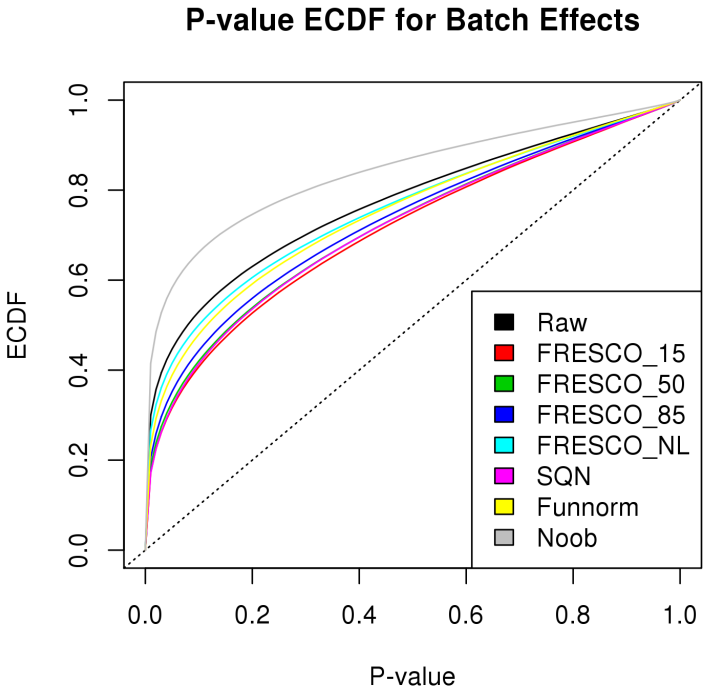

Fig. 10. Empirical cumulative p-value distributions from one-way ANOVAs for batch effect in the Reinius data.

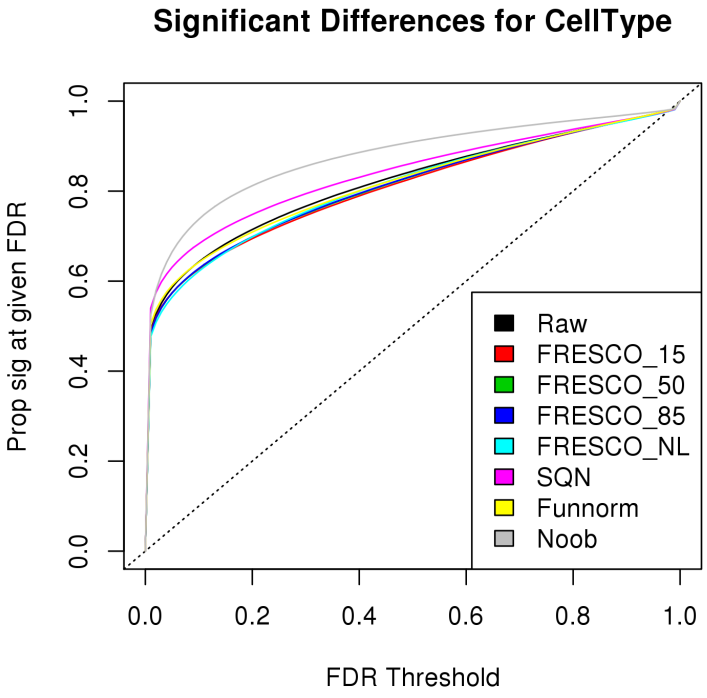

Fig. 11. Proportion of CpGs called significant for different FDR thresholds in the Reinius data. Quantile normalization and noob create more apparent differences.

It is odd that Noob seems to both improve power to detect real differences as well as significance of batch effects. This is likely due not to technical artifact *between* arrays, but perhaps to a lower overall level of signal relative to background in all samples. In the context of overall weaker biological signals, a good background correction may be more effective than a normalization method adjusting for technical artifacts between arrays.

Figure 12 gives $-\log_{10}$(p-values) from composite F-scores. We again observe quantile normalization providing the greatest overall increase in power due to reduction of error variance. When examining F_{Err}, Noob provides a large increase in power for less significant CpGs, while having a minimal effect on CpGs that already show

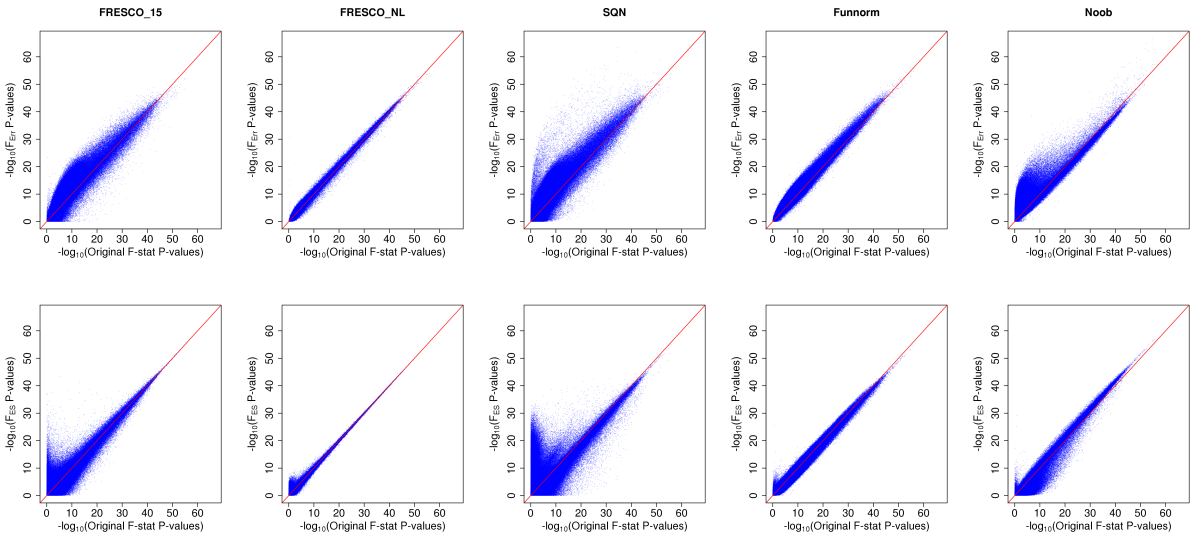

Fig. 12. Scatter plots of $-\log_{10}$(p-values) from composite F-statistics for cell type differences the Reinius data. Subset quantile normalization can create apparent significance by increasing effect sizes alone.

some significance. This result agrees with the idea that many of these CpGs may have signals that are weak relative to background levels, and that a background correction provides a better improvement in signal quality than adjusting between samples.

If we look at F_{ES}, we again observe a similar phenomenon in quantile normalization. Quantile normalization again appears to produce false significance from overfitting of the model.

2.4.3.3 TCGA Hepatocellular carcinoma

Figure 13 gives empirical cumulative p-value distributions for batch effects in the TCGA cancer data. Samples taken from multiple medical centers were randomized across 21 batches of varying size. Funnorm and Noob perform almost identically when reducing significance of batch effects. The fresco method omitting the loess surface fitting seems to perform the best. We should expect the cancer samples to have substantially differing methylation profiles across samples, so this is a situation

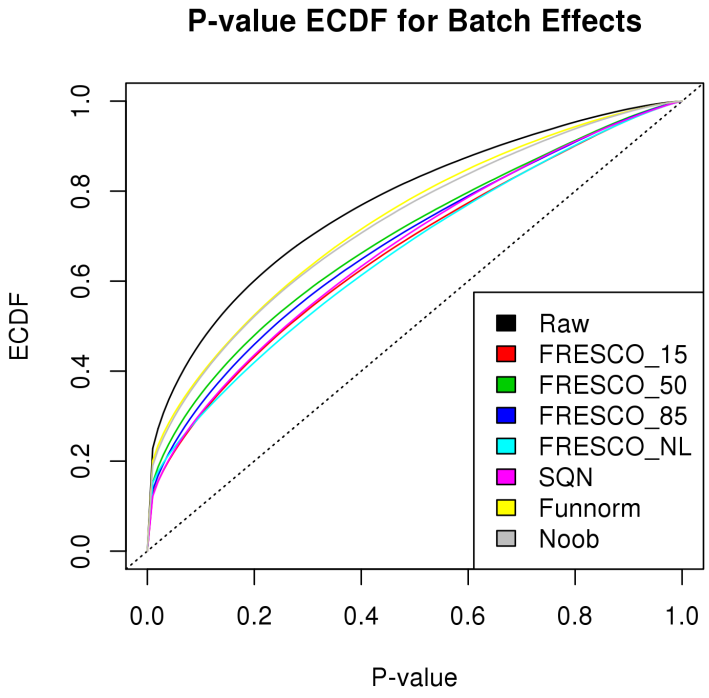

Fig. 13. Empirical cumulative p-value distributions from one-way ANOVAs for batch effect in the TCGA data.

where fresco should perform well.

Figure 14 gives a plot of the proportion of CpGs declared significant at a given FDR where p-values were adjusted using the Benjamini-Hochberg method (Benjamini and Hochberg 1995). Quantile normalization gives the greatest improvement in apparent significance, with fresco omitting surface fitting and Noob performing the second best. Interestingly, Funnorm, which is advertised as being specifically a method for cancer studies, performs less well in this scenario relative to the other methods.

Figure 15 gives $-\log_{10}$(p-values) from composite F-scores for the TCGA data. SQN and Funnorm appear to provide the greatest reduction in error variance. However, when looking at the plots for F_{ES}, it appears that SQN may call a substantially different set of CpGs significant after normalization due to over-fitting. Funnorm also

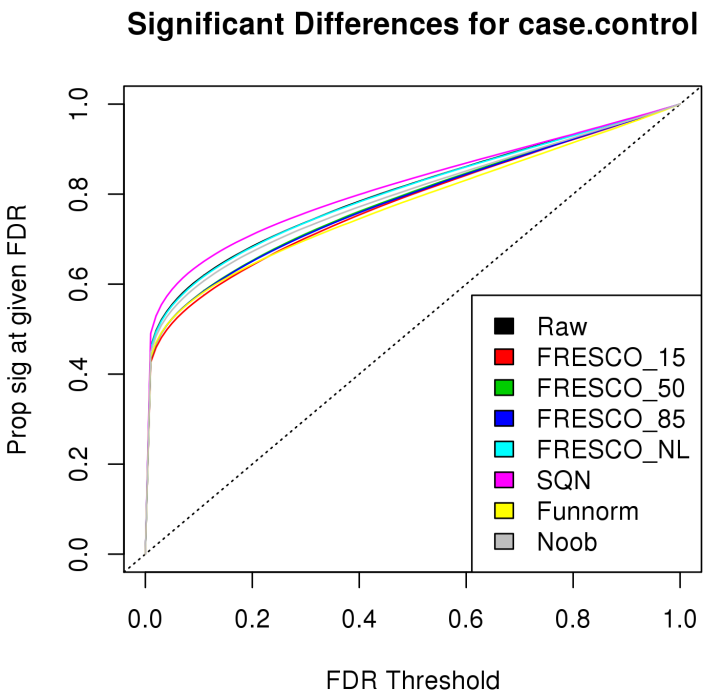

Fig. 14. Proportion of CpGs called significant for different FDR thresholds in the TCGA data. Quantile normalization creates more apparent significant differences.

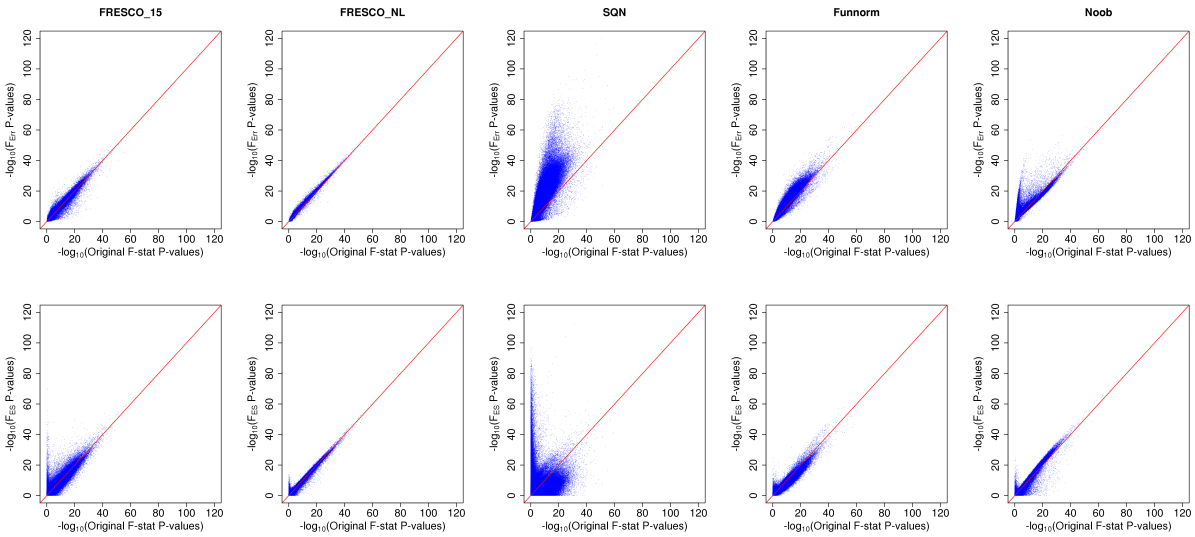

Fig. 15. Scatter plots of $-\log_{10}$(p-values) from composite F-statistics for differences between cancer and control in the TCGA data. Subset quantile normalization can create apparent significance by increasing effect sizes alone, and dampen real differences.

generally seems to suppress effect sizes in spite of reducing error variance.

2.5 Summary

Proper normalization of methylation samples with global distributional differences is important to obtain accurate results in epigenetic studies. We have demonstrated that, particularly in cancer studies, our method is robust to these kinds of biologically meaningful global differences. Quantile-based methods, particularly SQN, are prone to over-fitting and may not only muffle real biological variability, but create reproducible false differences. In the case of weaker biological signals, Noob appears to be a very effective method for background correction.

CHAPTER 3

METHODS FOR INTEGRATIVE ANALYSIS

3.1 Statistical issues in integrative genomic analysis

An important statistical issue when dealing with genomic data is that the number of samples (n) is generally much smaller than the number of parameters, or probes on the microarray (p). For example, a set of $n = 50$ gene expression microarrays may have $p = 20,000$ or many more measures of gene expression. In this situation where thousands of statistical models are being fit and significance tests are being performed, classical methods that control for the family-wise error rate are far too conservative (Nadon and Shoemaker 2002). Rather, methods have been developed that can select a set of significant genes while allowing for some false positives, but while asserting some control on the overall false-discovery rate, or FDR (Benjamini and Hochberg 1995; Storey 2003). In the situation of working with exon-level data and DNA methylation, each gene can now have tens and even hundreds of measurements which result in multiplying the number of potential tests by roughly an order of magnitude.

The increased number of probes on the DNA methylation array not only creates a larger number of overall potential significance tests, but these tests can be highly correlated. For a single gene, there may be multiple highly correlated CpG sites within 1 kilobase (kb) of each other around a gene promoter that could be adequately represented by simply using a summary measure such as their mean. However, this is not always the case and specific CpG sites in a given region may be discrepant with their surrounding neighbors. Jaffe et al. 2012 developed a method for automatic

summarization of correlated nearby CpG sites to help reduce the overall number of tests. While this is method is designed for specific scenarios just comparing DNA methylation data, to our knowledge no analogous method for integrative analysis has been developed. One goal of our proposed method is to provide dimension reduction to reduce the overall number of tests.

Another issue is that genes come in different sizes and have different numbers of exons and CpG sites. The Illumina 450k microarray only provides partial coverage of genic CpGs and completeness can vary from gene to gene. Despite the multivariate nature of genes, most standard downstream analyses after significance testing assume a single p-value or summary measure for each gene. These methods include different kinds of biological enrichment/pathway analyses such as Gene Set Enrichment Analysis (GSEA: Subramanian et al. 2005), topGO (Alexa, Rahnenfhrer, and Lengauer 2006), and Ingenuity Pathway Analysis (Abatangelo et al. 2009). Therefore, it is desirable to have a method that is able to take a set of heterogeneous genes, conduct the same omnibus test on each, and produce similar sets of results for each while not suffering too much bias due to differences in size between genes. At the same time, this omnibus test should not hinder identifying where within the gene the significant associations are occurring.

The method we propose is a two-step process that first uses Principal Component Analysis (PCA) for dimension reduction, and then Canonical Correlation Analysis (CCA) for significance testing and subsequent biological interpretation. As part of this research we have made the following methods presented in this chapter available in the R package "gdi" (Genomic Data Integration) which is currently available as a developmental version on GitHub (**https://github.com/paulmanser/gdi**).

3.2 Prerequisite statistical methods

3.2.1 Principal component analysis

The goal of principal component analysis is to represent the majority of variability in a data set in a lower-order linear subspace (Hotelling 1933). This is done by computing orthogonal linear combinations of variables, with the first linear combination explaining the maximum amount of variability. This can be thought of as fitting an ellipsoid to the data set with the axes of the ellipsoid corresponding to the vectors used when creating the linear combinations.

Generally the first k of these linear combinations, called principal component scores, are then kept as composites of the original variables. These composite principal component scores retain some portion of the total variability in the data. Principal component analysis is often the first step in an analysis, as in ours, before performing another statistical method such as linear regression. Principal components can be interpreted by how the linear combinations were computed, where variables with higher coefficient loadings are seen as contributing more to that set of scores. There are multiple ways of computing principal component scores including the Eigenvalue decomposition and the Singular Value Decomposition (SVD). For our method, we use the SVD which is detailed below.

Let X be an $m \times k$ matrix of real numbers. Then it can be decomposed in such a way that there exists an $m \times m$ orthogonal matrix U and $k \times k$ orthogonal matrix W obeying Equation 3.1

$$A = U\Sigma W^T \tag{3.1}$$

where the $m \times k$ matrix Σ has all diagonal entries $\sigma_i \geq 0$ for $I \in \{1, ..., \min(m, k)\}$ and

the other entries are zero. The positive constants σ_i are called singular values. In R, the singular value decomposition is computed using the dgesvd routine in LAPACK, which uses a QR algorithm (Anderson et al. 1999).

The singular values σ_i, which are equivalent to the square root of the eigenvalues of X^TX when X^TX is positive definite, can be interpreted as the standard deviations of each of the principal components. They can be useful in determining how many principal components to keep. Equation 3.2 below gives a measure of the amount of variability kept in the first k of K possible components. One can then choose the number of components to keep in order keep a certain amount of the total variability.

$$\text{Variance Retained in first } k \text{ of } K \text{ PCs} = \frac{\sum_{i=1}^{k} \sigma_i^2}{\sum_{i=1}^{K} \sigma_i^2} \tag{3.2}$$

Once the SVD has been performed, principal component scores T are then obtained by taking linear combinations of the original data X using either the W or U matrices given in Equation 3.3.

$$T = XW = (U\Sigma W^T)W = U\Sigma(W^TW) = U\Sigma \tag{3.3}$$

3.2.2 Canonical correlation analysis

Canonical correlation analysis is an analogous method to principal component analysis for finding linear combinations of data that explain the maximum amount of correlation between *two* sets of variables (Hotelling 1936). It can also be seen as a dimension reduction technique. Canonical correlation is a general method, and many well-known statistical methods such as multiple linear regression can be considered as special cases of canonical correlation.

The goal of canonical correlation analysis is to obtain sets of canonical covari-

ate scores for each of the two data sets that are maximally correlated. These scores can then be interpreted to find which variables are responsible for covariance between the two data sets by interpreting the communalities for each. Like principal components analysis, covariance matrices can be decomposed using the SVD or the spectral decomposition when computing the canonical covariates. For our purposes, we use the classical formulation of canonical correlation analysis using the spectral decomposition as detailed below.

Let X be a random vector of p variables and Y be a random vector of q variables. We can then define their cross-covariance as $\Sigma_{XY} = \text{cov}(X, Y)$ which is an $p \times q$ matrix whose (i, j) entry is $\text{cov}(x_i, y_j)$. Σ_{XY} can also be thought of as the off-diagonal component of the variance-covariance matrix when combining X and Y into a single random vector Z as in Equation 3.4.

$$\underset{(p+q)\times 1}{Z} = \begin{pmatrix} X \\ Y \end{pmatrix} \tag{3.4}$$

Σ_{XY} is then the off-diagonal of Σ_Z given in Equation 3.5.

$$\Sigma_Z = \begin{pmatrix} \Sigma_{XX} & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_{YY} \end{pmatrix} \tag{3.5}$$

The goal of canonical correlation analysis is to then find linear combinations $a^T X$ and $b^T Y$ such as to maximize ρ in $\text{cor}(a^T X, b^T Y) = \rho$. The linear combinations $U = a^T X$ and $V = b^T Y$ can be re-written as linear combinations of the standardized variables. For $i \in \{1, ..., \min(p, q)\}$ sets of canonical covariate vectors U_i and V_i can be written as $U_i = \boldsymbol{e}_i^T \Sigma_{XX}^{-1/2} X$ and $V_i = \boldsymbol{f}_i^T \Sigma_{YY}^{-1/2} Y$. Here ρ_i^2 are the first $\min(p, q)$ eigenvalues of $\Sigma_{XX}^{-1/2} \Sigma_{XY} \Sigma_{YY}^{-1} \Sigma_{YX} \Sigma_{XX}^{-1/2}$ and $\Sigma_{YY}^{-1/2} \Sigma_{YX} \Sigma_{XX}^{-1} \Sigma_{XY} \Sigma_{YY}^{-1/2}$. Lastly, $\boldsymbol{e}_i^T$ and $\boldsymbol{f}_i^T$ are the eigenvectors from the last two expressions in the previous sentence, re-

spectively. The canonical covariates have the following properties given in Equation 3.6:

$$\begin{aligned} \text{Var}(U_i) = \text{Var}(V_i) = 1 \\ \text{Cov}(U_i, U_j) = \text{Cor}(U_i, U_j) = 0 \quad i \neq j \\ \text{Cov}(V_i, V_j) = \text{Cor}(V_i, V_j) = 0 \quad i \neq j \\ \text{Cov}(U_i, V_j) = \text{Cor}(U_i, V_j) = 0 \quad i \neq j \end{aligned} \tag{3.6}$$

These sets of canonical covariates can then be interpreted by examining their loadings L_{U_i} and L_{V_i}, which are a measure of how much each original variable is related to canonical covariate i. The vector of loadings for canonical covariate U_i is given in Equation 3.7.

$$L_{U_i} = \text{Cor}(U_i, X) = \begin{pmatrix} \text{Cor}(U_i, X_1) \\ \vdots \\ \text{Cor}(U_i, X_n) \end{pmatrix} \tag{3.7}$$

Aside from being able to interpret canonical covariates using their loadings, we want to be able to quantify how much variance each canonical covariate explains in the original data. This is analogous to an R^2 measure, except that it is asymmetric. The correlation between U_i and V_i is given by ρ_i, but U_i and V_i are different linear combinations of the original data. Therefore, we compute what is called a redundancy coefficient (RC) separately for X and Y. The amount of variability explained in X and Y by canonical covariates V_i and U_i are given by $R^{*2}_{X_i}$ and $R^{*2}_{Y_i}$, respectively, in Equation 3.8.

$$R_X^{*2} = \rho_i^2 \sum_{j=1}^{p} L_{U_i}^2$$
$$R_Y^{*2} = \rho_i^2 \sum_{k=1}^{q} L_{V_i}^2 \tag{3.8}$$

3.3 A gene-level likelihood ratio test for association

3.3.1 Development

Here we present a general method for testing for association between two multivariate data sets, with a specific application to alternative splicing and DNA methylation. Another genomic mark such as histone modification or DNase-I hypersensitivity could be substituted for DNA methylation to conduct a similar analysis within the same framework. We first introduce the statistical method used, and then justify certain empirical assumptions using real data and simulation studies.

3.3.1.1 A likelihood ratio test for CCA

When performing canonical correlation analysis, we are using linear combinations $a^T X$ and $b^T Y$ to perform dimension reduction and model the cross-covariance Σ_{XY} from Equation 3.5. However, if $\Sigma_{XY} = 0$ then there is no reason to perform canonical correlation analysis. Therefore, we would like to have a statistical significance test with $H_0 : \Sigma_{XY} = \mathbf{0}$ and $H_1 : \Sigma_{XY} \neq \mathbf{0}$. If we take $\boldsymbol{S}_Z$, $\boldsymbol{S}_{XX}$, $\boldsymbol{S}_{YY}$, and $\hat{\rho}_i^{*2}$ as sample estimates of the population parameters Σ_Z, Σ_{XX}, Σ_{YY}, and ρ_i^{*2} respectively, then we can formulate the following likelihood ratio test statistic in Equation 3.9.

$$-2\ln(\Lambda) = n \ln\left(\frac{|\boldsymbol{S}_{XX}||\boldsymbol{S}_{YY}|}{|\boldsymbol{S}_Z|}\right) = -n \ln \prod_{i=1}^{\min(p,q)} (1 - \hat{\rho}_i^{*2}) \tag{3.9}$$

This test statistic is distributed as χ^2_{pq} under the null hypothesis (Kshirsagar 1972; Lawley 1959; Johnson and Wichern 2007). We will replace the multiplicative factor n in the likelihood ratio test statistic with $n - 1 + \frac{1}{2}(p + q + 1)$ to improve the χ^2 approximation as suggested by Bartlett 1939.

3.3.1.2 Using PCA for dimension reduction

The test statistic in Equation 3.9 is only feasible when $\boldsymbol{S}_Z$, $\boldsymbol{S}_{XX}$, and $\boldsymbol{S}_{YY}$ are non-singular since determinants are being computed. This is often not the case in genomic studies where a gene may have over 100 methylation loci for a given gene, but only 20 samples. Even if sample covariance matrices are non-singular, the degrees of freedom for the test statistic can become large very quickly and will vary from gene to gene, which will result in a variable loss of power across genes. For example, one gene may have $p = 50$ observed methylation sites and $q = 5$ exons, which will result in a significance test with $5 \times 50 = 250$ degrees of freedom. Another gene may have $p = 15$ observed methylation sites and $q = 3$ exons for a significance test with $15 \times 3 = 45$ degrees of freedom. In order to make many of these significance tests feasible and avoid a variable and biased loss of power across different genes, we propose a preliminary dimension reduction step using PCA before performing the likelihood ratio test.

To do this, we perform PCA using the SVD for each gene on each of the two data sets separately keeping the first k principal components from each. We can then replace $\boldsymbol{S}_Z$, $\boldsymbol{S}_{XX}$, and $\boldsymbol{S}_{YY}$ in Equation 3.9 with $\boldsymbol{S}^*_Z$, $\boldsymbol{S}^*_{XX}$, and $\boldsymbol{S}^*_{YY}$ where $\boldsymbol{S}^*_{XX}$ and $\boldsymbol{S}^*_{YY}$ are $k \times k$ covariance matrices for the first k principal component scores for methylation and alternative splicing and $\boldsymbol{S}^*_Z$ is as given in Equation 3.5, but using $\boldsymbol{S}^*_{XX}$ and $\boldsymbol{S}^*_{YY}$ and their cross-covariance. We keep $k = 3$ principal components for methylation and splicing from each gene. This choice will be justified empirically

in the following paragraphs for DNA methylation and alternative splicing. A major benefit of the dimension reduction step is that all significance tests will be similarly powered with a χ^2_9 limiting null distribution for all test statistics.

Roughly 70% of the variability in methylation can be contained in the first three principal components regardless of the number of CpG sites in the gene. Figure 16 plots the proportion of variability explained by the first three PCs in methylation in each gene versus the total number of original CpGs. The average proportion of variance we would expect to keep from independently and identically distributed normal data is given by the red line. We can see that after a certain point, roughly 70% of variability is retained regardless of the number of CpG sites in the gene. The red line was computed by conducting 500 simulations of i.i.d. normal variables and taking the average proportion of variation explained by the first three principal components. It is important to note that much of the reason for the success of PCA in this scenario is due to the fact that many of the probes are positioned very close to each other on the gene and are highly correlated.

If we plot a similar figure for the splicing index from the Affymetrix HT 1.0 Exon Array we get a slightly different picture in Figure 17. Many genes are close to the red null line. However, we should expect genes that are not alternatively spliced to have a similar amount of variability explained by the first 3 PCs as the independent data. We do see a similar phenomenon in that after about 20 exons, the average amount of variability explained remains relatively constant.

3.3.2 Controlling type I error

In order to assess the type I error rate for the likelihood ratio test, we conduct a set of simulation studies. Chi and Muller 2013 conducted similar simulation studies to test the effectiveness of performing principal component analysis as a general

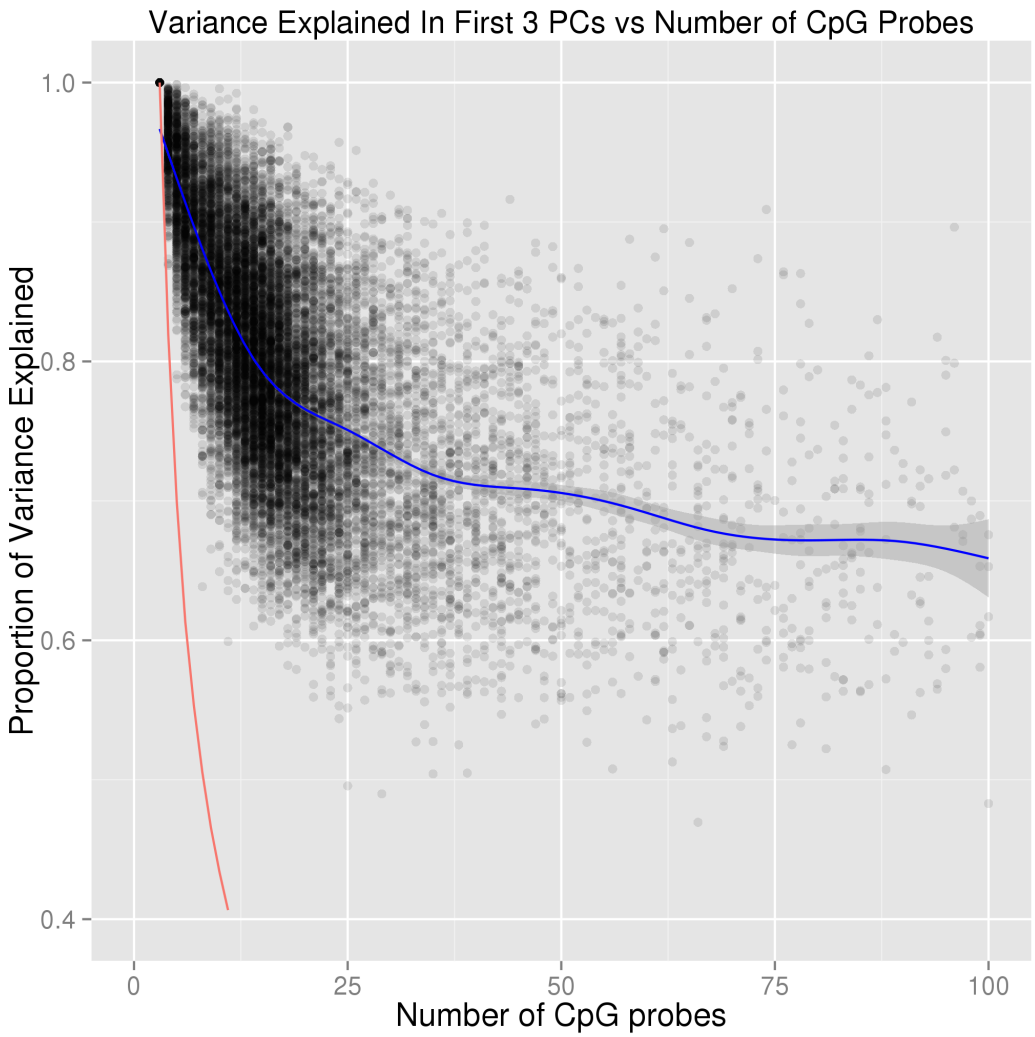

Fig. 16. Proportion of variance explained by first 3 principal components for DNA methylation. Mean results from a simulated i.i.d. normal distribution are given as the red line. The first 3 principal components are generally able to explain at least 70% of the variance in most genes regardless of the number of CpGs.

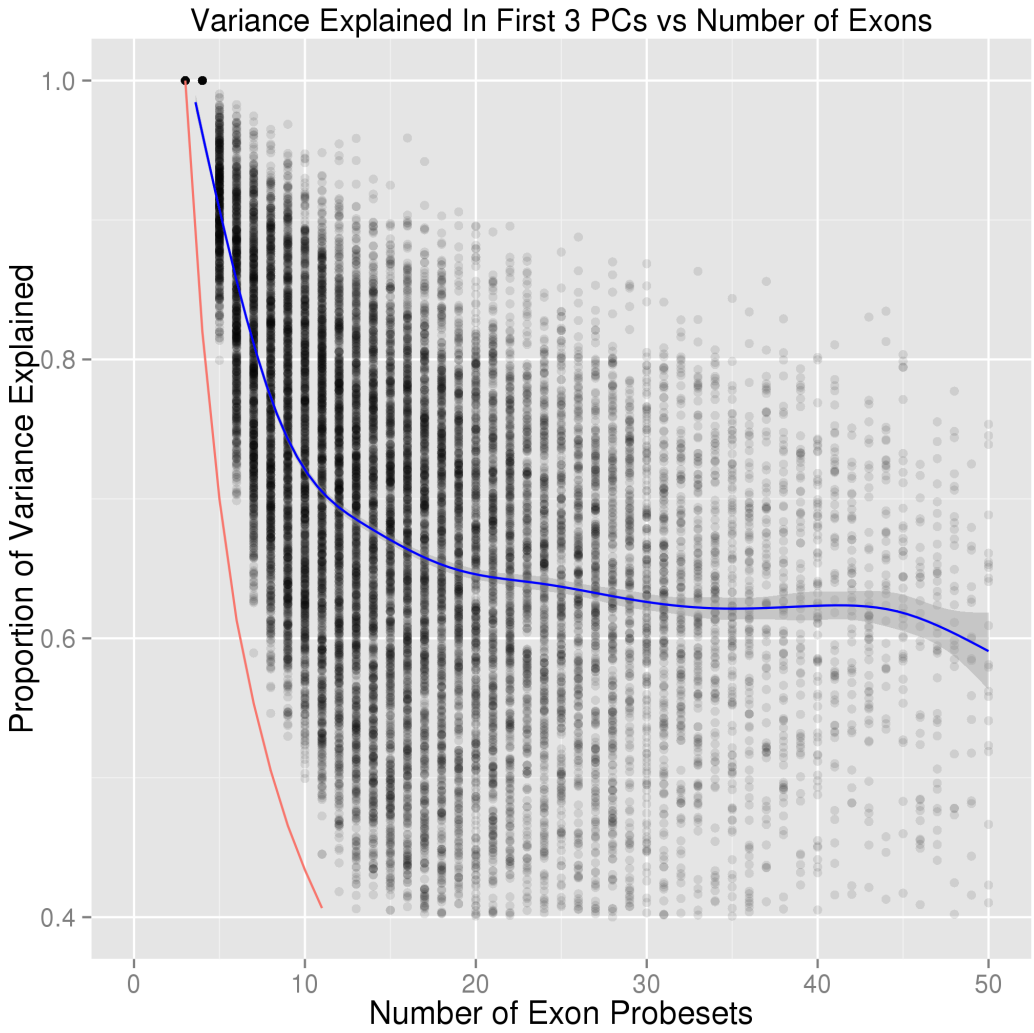

Fig. 17. Proportion of variance explained by first 3 principal components for splicing index. Mean results from a simulated i.i.d. normal distribution are given as the red line. The first 3 principal components are generally able to explain at least 65% of the variance in most genes regardless of the number of exon probesets.

dimension reduction step before performing MANOVA when the variance-covariance matrix of outcomes was singular. Our simulation studies are focused on a specific application in genomics, so the data will be simulated as if it were coming from a gene model with comparable variability to what is observed in the BrainSpan data.

Despite the overall distribution of β-values being bi-modal, individual CpGs sites generally have a distribution that is uni-modal. For simplicity, we will simulate data for both splicing indices and DNA methylation β-values as coming from a multivariate normal distribution. For a random vector $\boldsymbol{X}$ of p CpG sites and random vector $\boldsymbol{Y}$ of q exons, we simulate data using their joint covariance $\boldsymbol{\Sigma}$ in Equation 3.10.

$$\boldsymbol{\Sigma} = \begin{pmatrix} \Sigma_{XX} & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_{YY} \end{pmatrix} \tag{3.10}$$

In the case of generating null data for testing type I error rates, Σ_{XY} is set equal to zero. Σ_{XX} and Σ_{YY} are constructed using a compound symmetry correlation structure with variances that vary linearly as a function of a slope parameter θ and intercept parameter δ. The level of correlation between loci within a data modality is determined by the parameter ρ. The general form used for generating Σ_{XX} and Σ_{YY} for the case of three loci is given for Σ_{XX} in Equation 3.11.

$$\Sigma_{XX} = \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 & \rho\sigma_1\sigma_3 \\ \rho\sigma_2\sigma_1 & \sigma_2^2 & \rho\sigma_2\sigma_3 \\ \rho\sigma_3\sigma_1 & \rho\sigma_3\sigma_2 & \sigma_3^2 \end{pmatrix} \tag{3.11}$$

Here $\sigma_i^2 = \theta(i-1) + \delta$ allows for non-uniform variances such that some sites are more variable than others. Other parameters of interest are the number of subjects n, the number of methylation loci p and exons q, and the number of principal components kept after the pre-processing step k. Separate parameters determined empirically from

the BrainSpan data are used to simulate methylation and splicing data.

For σ_i^2, θ and δ are chosen so that the range of variances used in the simulations corresponds to the inter-quartile range of the variances of all CpGs from the BrainSpan samples. The within-gene correlation parameter ρ is determined from the median correlation among CpG sites within the same gene. This was computed by estimating a correlation matrix for each gene from the BrainSpan samples, concatenating all of the off-diagonal elements from all genes, and taking the median. For I methylation sites, $\theta = 0.00313/(I-1)$ and $\delta = 0.000437$ covers the inter-quartile range of methylation variances: $\{0.000437 - 0.00357\}$ and the within gene correlation is $\rho = 0.25$. For J exons, $\theta = 0.158/(J-1)$ and $\delta = 0.086$ covers the inter-quartile range of inclusion ratio variances: $\{0.086 - 0.244\}$ and the within gene correlation is $\rho = -.0678$. The within gene correlation is slightly negative for splicing due to the mean centering used to compute the splicing index. Simulation studies were conducted using above parameters for a "typical" gene with $p = 20$ CpGs and $q = 8$ exons. Simulation studies are run for sample sizes $n \in \{26, 50, 100, 200, 500\}$ and keeping $k \in \{1, 3, 5, 10, 15\}$ principal components. Table IV gives simulation results for $10,000$ simulations. Type I error appears to be conserved in nearly all of the various scenarios. Type I error appears to become inflated when keeping a large number of principal components relative to the sample size, but this is a non-issue since keeping more than three principal components markedly decreases power and is not adviseable.

3.3.3 Assessing power

In order to simulate scenarios where a relationship exists between methylation and splicing, we conduct similar simulation studies, but add in non-random coinciding changes in methylation and splicing after generating random data where no relation-

Table IV. Type I Error for n samples after retaining k principal components

n	k	**Type I Error**
26	1	0.04962
26	3	0.05164
26	5	0.05481
26	10	0.11752
26	15	0.46554
50	1	0.04966
50	3	0.04961
50	5	0.05086
50	10	0.05755
50	15	0.06932
100	1	0.04944
100	3	0.04993
100	5	0.05121
100	10	0.05055
100	15	0.05340
200	1	0.05037
200	3	0.05005
200	5	0.04904
200	10	0.04966
200	15	0.04985
500	1	0.04792
500	3	0.04891
500	5	0.04965
500	10	0.04984
500	15	0.05046

ship exists. Here, the joint variance-covariance matrix takes on a simpler form where all loci have the same variance which is estimated empirically from the data by taking the median variance of all loci. All methylation loci are given variance $\sigma^2 = 0.00123$ and exon inclusion ratios all have variance $\sigma^2 = 0.15$. Equation 3.12 gives the form of the variance-covariance matrix.

$$\Sigma_{XX} = \begin{pmatrix} \sigma^2 & \rho\sigma^2 & \rho\sigma^2 \\ \rho\sigma^2 & \sigma^2 & \rho\sigma^2 \\ \rho\sigma^2 & \rho\sigma^2 & \sigma^2 \end{pmatrix} \tag{3.12}$$

After multivariate normal random data has been generated, non-random differences are then added in to the data. When adding in these changes, two parameters must be set for both methylation and splicing: the effect size and number of loci affected. For the simulation study, we take the simplest approach of having the change occur across two groups (i.e. case vs control). For example, a set of 3 CpGs will be more methylated in one group than another which corresponds to an increase in exon inclusion in a single exon. The "typical gene" from the previous simulation studies with $p = 20$ CpGs and $q = 8$ exons is again used. In this scenario the parameters that are allowed to vary are:

1. Number of PCs kept: k
2. Number of samples: n
3. Number of CpG sites affected: p^*
4. Methylation effect size: M_Δ
5. Number of exons affected: q^*
6. Splicing effect size: E_Δ

The effect size for methylation is set to $M_\Delta = 0.2$, and $E_\Delta = 1.2$ for splicing. Results are given in Table V. The likelihood ratio test appears to be generally underpowered in situations where a single CpG is correlated with a single exon. However,

more pervasive effects that exist in multiple CpGs and exons are detected more often, particularly when three principal components are kept. This kind of scenario is realistic in the case of alternative promoter usage resulting in the inclusion/exclusion of multiple exons. These results echo those of Chi and Muller 2013, who found that performing PCA as a dimension reduction step before MANOVA was most effective when three principal components were kept. For sample sizes less than 50, it is likely that we may miss sparse, specific correlations between methylation and splicing such as the inclusion of a single cassette exon.

3.4 Interpreting results using canonical correlation

3.4.1 Canonical covariate regression

While the likelihood ratio test from the previous section provides a general test for examining relationships between two sets of covariates, it does not provide information on what may be responsible for these associations. Once a significant result is found, the next logical step is to determine if differences in methylation and splicing are co-occurring across covariates of interest. Canonical correlation analysis is able to reduce the relationship between the two sets of principal component scores used for the likelihood ratio test into pairs of canonical covariate vectors that are maximally correlated with each other. Bartlett suggests a sequential set of χ^2 tests for determining the number of canonical covariate pairs to keep, but the overall significance level α is difficult to determine (Bartlett 1939; Johnson and Wichern 2007). We only use the first set of canonical covariates for our purposes, but more could certainly be included.

Now that we have a single set of scores each for methylation and splicing, we can take a model-based approach to interpret the first set of canonical scores. This is

Table V. Power to detect case vs control relationships. For n samples with p^* CpG sites with a $M_\Delta = 0.2$ case-control difference and q^* exons with a $E_\Delta = 1.2$ case-control difference, k principal components are retained.

n	k	p^*	q^*	**Power**
26	1	1	1	0.33199
26	2	1	1	0.21412
26	3	1	1	0.14699
26	5	1	1	0.10506
26	1	3	4	0.87727
26	2	3	4	0.45073
26	3	3	4	0.25421
26	5	3	4	0.14981
50	1	1	1	0.33868
50	2	1	1	0.38685
50	3	1	1	0.27998
50	5	1	1	0.17378
50	1	3	4	0.99627
50	2	3	4	0.96162
50	3	3	4	0.80318
50	5	3	4	0.46619
100	1	1	1	0.12307
100	2	1	1	0.42592
100	3	1	1	0.33774
100	5	1	1	0.22755
100	1	3	4	0.99768
100	2	3	4	0.99754
100	3	3	4	0.98152
100	5	3	4	0.84821

accomplished by regressing them against covariates of interest such as brain region or age. Since the two sets of canonical covariates U_1 and V_1 are by definition maximally correlated with each other, we cannot fit two separate models and combine their p-values using Fisher's method, which assumes independent tests (Fisher 1973). We instead fit a single linear mixed effects model for both sets of canonical scores with a random effect to account for their correlation. The model is fit using maximum likelihood estimation. Equation 3.13 gives an example model for the i^{th} observation y_i in $Y = \{U_1, V_1\}$ with continuous covariate x_i and categorical covariate at level j in canonical covariate pair k.

$$y_{ijk} = \alpha_j + x_i\beta + b_k + e_i \tag{3.13}$$

Here α is the parameter for a categorical covariate and β is the coefficient for a continuous covariate x_i. Since there is a pair of correlated canonical scores from methylation and expression for each sample, a random effect b_k is included. If canonical covariates are arranged as $Y = \{u_1, v_1, ..., u_n, v_n\}$, then adding the random effect b is equivalent to specifying a block diagonal covariance structure in blocks of size two for the linear model. Although maximum likelihood estimation gives biased estimates of random effects in mixed effects models, it allows for testing of fixed effects using a likelihood ratio test for nested models. The full model in Equation 3.13 can then be compared to a reduced model omitting one or both α or β using the likelihood ratio test given in Equation 3.14

$$2\left(\hat{\ell}_{\text{full}} - \hat{\ell}_{\text{reduced}}\right) \sim \chi^2_{(p-q)} \tag{3.14}$$

where $\hat{\ell}_{\text{full}}$ and $\hat{\ell}_{\text{reduced}}$ are the maximized log-likelihoods for the full and reduced models, respectively and $p - q$ is the difference between the number of parameters in

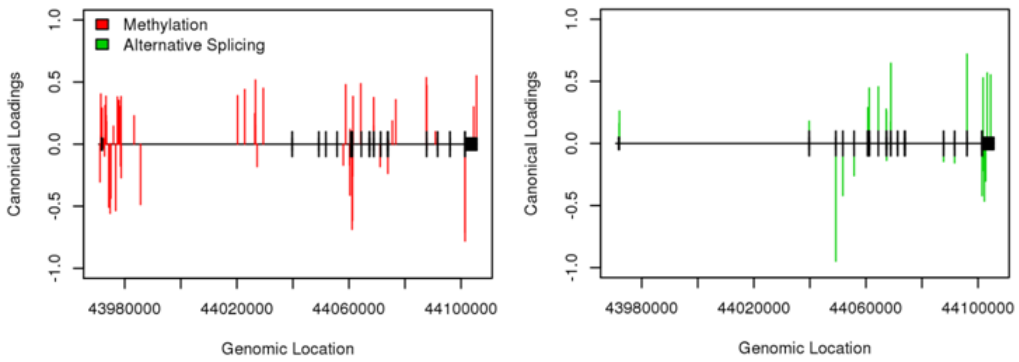

Fig. 18. Example of canonical loadings plotted over a gene model for DNA methylation and alternative splicing

the full and reduced models. If a coefficient is found to be statistically significant, then the relationship between alternative splicing and methylation can be attributed to their covariation across that variable. The χ^2 approximation to the likelihood ratio can be anti-conservative and a bootstrapping approach may need to be implemented to obtain credible results.

3.4.2 Interpreting canonical loadings

Once a significant relationship is established via the likelihood ratio test, and a putative mechanism for the relationship is determined from the linear mixed effects model, the final step is to determine if specific CpG sites in the gene are related to specific exons. A straightforward way to do this is to examine the canonical loadings from Equation 3.7. Since each element of the canonical loading vectors corresponds to a CpG site or exon with a specfic location in the gene, we can generate a bar plot of the canonical loadings with bars positioned at their corresponding genic locations. A gene model can then be added to aid in the visualization of where in the gene strongly loading CpGs and exons exist and if they co-localize. Figure 18 gives an example of a bar plot of canonical loadings with a gene model.

While this is a convenient way to interpret results from a single gene, it is not a viable option for interpreting the loadings of thousands of genes. In the following section we introduce a permutation test that automatically tests for statistically significant co-localization of high loadings on the gene model.

3.5 A gene-level permutation test for spatial co-localization

Once a set of genes have met some FDR threshold for significance from the likelihood ratio test, we would like to have an automated way of testing whether the relationships between alternative splicing and methylation co-localize at specific places in the gene. An example of co-localization would be differential methylation in the third exon affecting that exon's inclusion ratio. Alternatively, differential methylation in the promoter region could somehow be associated with the inclusion ratio of the last exon. For this second case, it is perhaps less straightforward to give a putative biological explanation. Therefore, we would like to establish a statistical test to be able to distinguish these two kinds of scenarios. We propose two similar tests to do this: the first test is a global test for co-localization. The second test is specific to a set of canonical covariates. This allows for further interpretation of canonical covariates. We can establish a putative mechanism via the linear mixed model from Equation 3.13, and then test whether the relationship appears to be *cis*-acting. To be explicit, the null and alternative hypotheses are given in Equation 3.15.

$$\begin{aligned} H_0 &: \text{Exon and CpG locations are interchangeable} \\ H_1 &: \text{Exons and CpG sites have } \textit{cis} \text{ relationships} \end{aligned} \tag{3.15}$$

3.5.0.1 A permutation test on R^2 matrices

For a given gene, let R^2_{XY} be an $p \times q$ matrix whose ij^{th} entry corresponds to the coefficient of determination r^2 between the i^{th} CpG site and j^{th} exon. Similarly, let D be the $p \times q$ genomic distance matrix between each CpG site and exon. When calculating distance between exons and CpG sites, the center of the exon and the cytosine base in CpG sites are used as points of reference. For each exon/CpG site pair we can then get a weighted measure of association t_{ij} that is a product of the coefficient of determination r^2_{ij} and a function of genomic distance $\omega(d_{ij})$ in Equation 3.16.

$$t_{ij} = r^2_{ij}\omega(d_{ij}) \tag{3.16}$$

We then specify $\omega(\cdot)$ as an exponentially decaying function given in Equation 3.17. Since the Illumina 450k array provides incomplete coverage, a single CpG site is often the only information available for whole regions of a gene. However, correlation among CpG sites decays rapidly as a function of genomic distance with correlation decreasing to approximately 0.4 after 400 bp (Zhang et al. 2013). Therefore, the half-life of the exponential decay function $\frac{\ln(2)}{\lambda}$ is specified to be 400 bp since we cannot be confident that relationships between methylation and splicing much greater than 400 bases reflect a *cis*-acting effect. For different genomic assays, different appropriate half-lives may be chosen.

$$\omega(x_{ij}) = e^{-\lambda d_{ij}} \tag{3.17}$$

Once the half-life has been specified, a test statistic T can be computed by simply taking the sum of the individual t_{ij}. This sum is an aggregate measure of spatial co-localization between methylation and splicing. A permutation test is then performed

by permuting the rows and columns of D, which is effectively permuting the locations of exons and CpG sites. A formula for the permutation test statistic T_k is given in Equation 3.18 where D^* is the matrix of permuted distances.

$$T_k = \sum_{i=1}^{p} \sum_{j=1}^{q} \sigma_{ij} \, \omega(d^*_{ij}) \tag{3.18}$$

A permutation p-value can then be computed from the permutation distribution of K permutation test statistics given in Equation 3.19. Permutation testing is performed on several genes, and a distribution of permutation p-values is ultimately obtained. Permutation p-values from different genes have different non-uniform null distributions on different discrete supports. A method for estimating false discovery rates for sequential permutation p-values has been proposed, which is similar to this scenario (Bancroft, Du, and Nettleton 2013).

$$p_{perm} = \frac{\sum_{k=1}^{K} \mathbf{1}_{T_k > T}}{K} \tag{3.19}$$

3.5.0.2 A permutation test on canonical communalities

A similar permutation test can also be performed for a specific pair of canonical covariates. The permutation test is almost identical, except the matrix R^2_{XY} is replaced by the outer product of canonical loadings $\Psi = L_{U_1} L_{V_1}^T$ from the first pair of canonical covariates. A formula for the modified test statistic is given in Equation 3.20.

$$T_k = \sum_{i=1}^{p} \sum_{j=1}^{q} \psi_{ij} \omega(x^*_{ij}) \tag{3.20}$$

Performing a permutation test on canonical loadings allows the added benefit of combining permutation test results and significance testing from the linear mixed

effects model from Equation 3.13 to make claims about *cis*-acting relationships occurring across specific factors such as brain region. It is possible to make these claims because, while permutation testing using R^2_{XY} is a general test of loci-specificity, the test on canonical communalities is specific only to that set of canonical covariates that have associated scores. Therefore, if one or more signifiance testing results on parameters from Equation 3.13 are significant, we can say a *cis*-acting relationship occurs across those covariates.

3.6 Implementation

We implemented the above methods in R package "gdi" whose developmental version is currently available on GitHub (**https://github.com/paulmanser/gdi**). Since genomic data sets can be very large, the ff package is used to store genomic data sets out of memory (Adler et al. 2014). Data are then read into memory in chunks, analysis is performed, and results are written back out to disk. Conducting thousands of tests can be quite computationally and time intensive. This is particularly true for the permutation tests, especially when the number of permutations is large. To speed up significance testing, the foreach package for parallel computing is used (Analytics and Weston 2014). The ff and foreach packages play well together in that multiple copies of the data are not created when performing parallel programming, each core only takes what it needs, writes back to disk, and then reads in the next small piece. Together, they allow the methods introduced in this chapter to scale to large data sets both in terms of memory storage and computation time.

3.7 Summary

In this chapter we present a set of methods for integrating two different genomic data sets on a gene-by-gene basis. These methods proceed in three steps for each gene.

First, a likelihood ratio test on the sample variance-covariance matrix is performed to test for general association between the two data sets. Next, canonical correlation analysis is performed, and resulting canonical covariates are regressed against covariates of interest using a linear mixed effects model to test whether general associations occur across factors of interest. Lastly, a permutation test is performed to test whether significant relationships between the two data sets co-localize on specific genic regions in a *cis*-acting manner.

CHAPTER 4

INTEGRATIVE ANALYSIS OF DEVELOPMENTAL BRAIN DATA

4.1 Overview of neuroscience and neurogenomics

4.1.1 Major neural cell types

While the neuron is the most commonly known and well-studied neural cell type, they actually comprise substantially less than half of cells in the human brain. Brain tissue consists of several other types of cells called glia. Glial cells support, protect, and supply nutrients to neurons. Once thought of as simply the glue that held the brain together, recent research has begun to show glial cells may be more important than once thought (Fields 2009). The data used in this chapter arises from microarray experiments performed on frozen post-mortem brains, and therefore assayed tissue samples reflect an aggregate signal obtained from all cell types. While most results in this chapter are interpreted from a neuron-centric point of view, it is important to acknowledge that observed changes may not necessarily be the results of differences within or between neurons. Therefore, it is important to give a brief introduction to neurons as well as the different glial cell types and their respective roles in brain tissue.

Neurons are specialized cells in the human brain that transmit signals via electrical and chemical means. While there are many types and subtypes of neurons, they all share a similar basic morphology. Neurons have three major components: the cell body or soma, a single cellular extension called an axon that sends signals out to other neurons, and a collection of thin branching structures called dendrites

that receive information from other neurons. Information flow in neurons is unidirectional. There are two major categories of neurons: excitatory neurons which transmit information from neuron to neuron, and inhibitory neurons which act to inhibit the activity of excitatory neurons. During development of the cerebral cortex, neurons migrate and arrange themselves in specialized layers. Neurons connect with each other via synapses which are junctions between an axon of one neuron and a dendrite, or sometimes an axon, of another.

Oligodendrocytes are a type of glia that serve to protect and insulate neurons. Specifically, they form a myelin sheath that wraps around axons to insulate them, which improves axon efficiency. A single oligodendrocyte can insulate axons from several neurons. Multiple sclerosis, a disease of the nervous system, is characterized by the destruction of myelin sheaths of neurons.

Astrocytes are star-shaped glial cells that have several functions in the brain. They are the most abundant cells in the brain and provide structural support. They help to regulate the environment of neurons by regulating extracellular ion concentrations, as well as functioning in neurotransmitter re-uptake and release. Astrocytes also provide metabolic support and nutrients to neurons.

Microglia are the resident macrophages of the brain. Due to the brain being separated from the rest of the body by the blood-brain barrier, the microglia comprise the brain's own separate unique immune system. Besides defending against infectious agents, microglia also function to remove unwanted cellular matter in the brain such as damaged or dead cells as well as neurofibrillary tangles and function in synaptic pruning.

4.1.2 Issues in neurogenomics

Analysis of genomic data poses many unique problems in statistics, experimental design, and data quality control. In addition to these problems, analysis of genomic brain data creates additional concerns and caveats. Unlike other tissues, brain samples cannot be taken longitudinally from living subjects over time in the way blood samples or cancer biopsies can be. Even in the case of animal models, animals are generally sacrificed and the whole brain is recovered. This poses a problem in longitudinal studies, like BrainSpan, because individual differences are then confounded with temporal differences. In order to address this issue, additional constraints need to be in place when analyzing the data. For example, fitting a saturated ANOVA model treating time as categorical will result in many false positives that are actually due to individual differences. If time is treated as continuous, a model with a reasonable amount of smoothness such as a quadratic or cubic linear regression model should yield fewer false positives due to individual differences.

Not only are brains donated once, but the patient must die in order to donate. In the case of human brains, the cause and circumstances of death can have an effect on data quality. This is particularly true for RNA, which is a less stable molecule than DNA. The post mortem interval (PMI) is the interval of time between death and when the brain is frozen to be used later. Longer PMIs are generally correlated with lower RNA integrity which is measured using an RNA integrity number (RIN). A lower RIN corresponds to smaller more fragmented pieces of RNA which are more difficult to uniquely identify via sequencing or microarray hybridization. PMI and RIN can be included as additional covariates in a linear model to mitigate for these effects.

When comparing brains across psychiatric conditions, such as schizophrenia,

there are additional concerns for confounding. Psychiatric patients are usually taking one or more medications to treat their condition such as anti-psychotics. This usually results in almost complete confounding between medication regimen and condition, although amount and length of treatment will vary between patients. Additionally, schizophrenic patients may not take good care of themselves and are more prone to smoking cigarettes or having a poorer diet. Like PMI and RIN, if information on medical history, smoking status, and diet are available they can also be included as covariates in statistical models.

Given the additional caveats and confounding effects that come along with analyzing post mortem brain samples, it is still possible to obtain meaningful results. Proper experimental design, normalization, and quality control can help to ensure data quality. Choosing appropriate statistical methods and including important covariates can help mitigate confounding factors.

4.2 Estimating cell type admixtures in brain tissue

4.2.1 Estimating the neuronal fraction

In order to estimate the relative abundance of neurons in the BrainSpan developmental samples, we use data coming from orbitofrontal cortex that has been sorted using FACS to separate out NeuN+ cells (Kozlenkov et al. 2013). The data set provides methylation profiles for 2 replicates of Neun+ and Neun- samples each from six brains for a total of 24 samples. Since cerebellum is so distinct from other brain regions, it is omitted from the cell proportion estimation procedure as estimates may be unreliable. Average Neun+ and Neun- methylation profiles are then used as predictors in a linear model where estimates of the neuronal and non-neuronal proportions are constrained to sum to one (Houseman et al. 2012). Figure 19 gives box

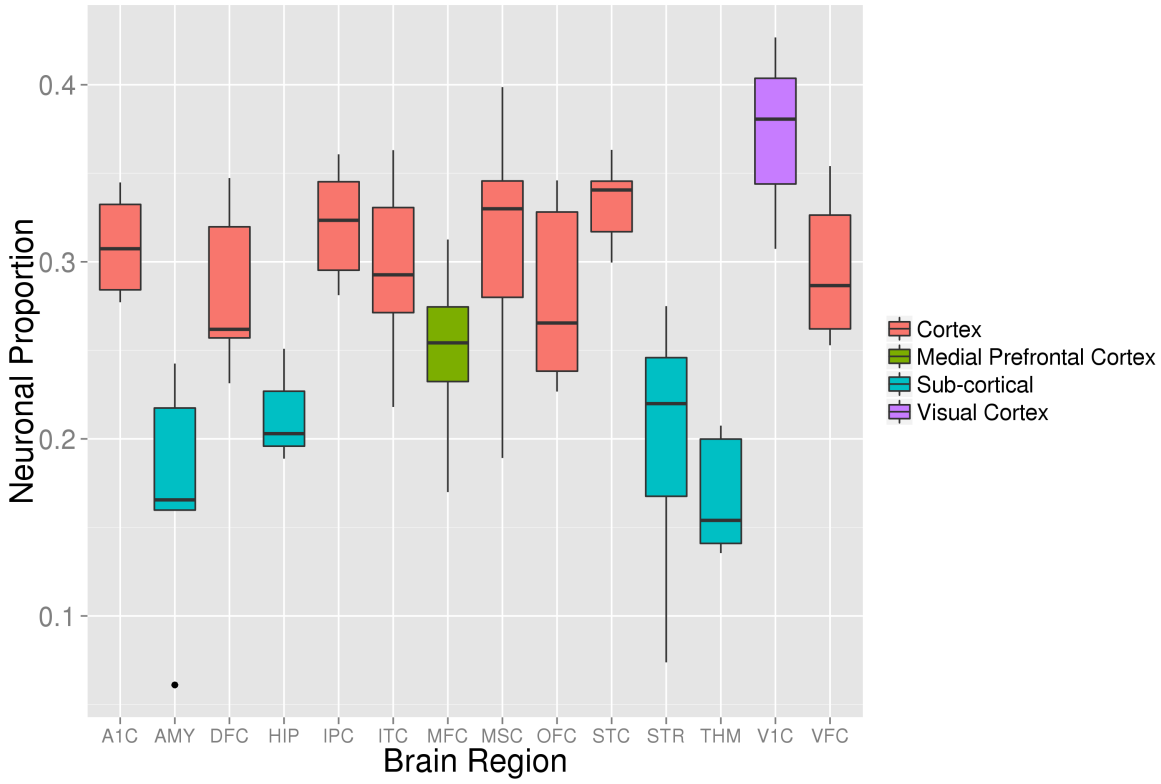

Fig. 19. Box plots of estimates of neuronal proportions by brain region in the BrainSpan data

plots estimates of the neuronal fraction plotted by brain region using the regional abbreviations from Table I.

Notably, visual cortex has the highest neuron density which is in agreement with recent findings from a primate study (Collins et al. 2010). This increase in neuron density can be attributed to a decrease in the average size of neurons, allowing more of them to be packed into the same amount of volume. Sub-cortical regions tend to have lower estimated neuronal densities than cortical regions. If we instead plot these same estimates against age in Figure 20, we are able to observe a decreasing trend in neuron density in both cortical and sub-cortical regions. This is likely a combination of both astrocyte proliferation and neuron death.

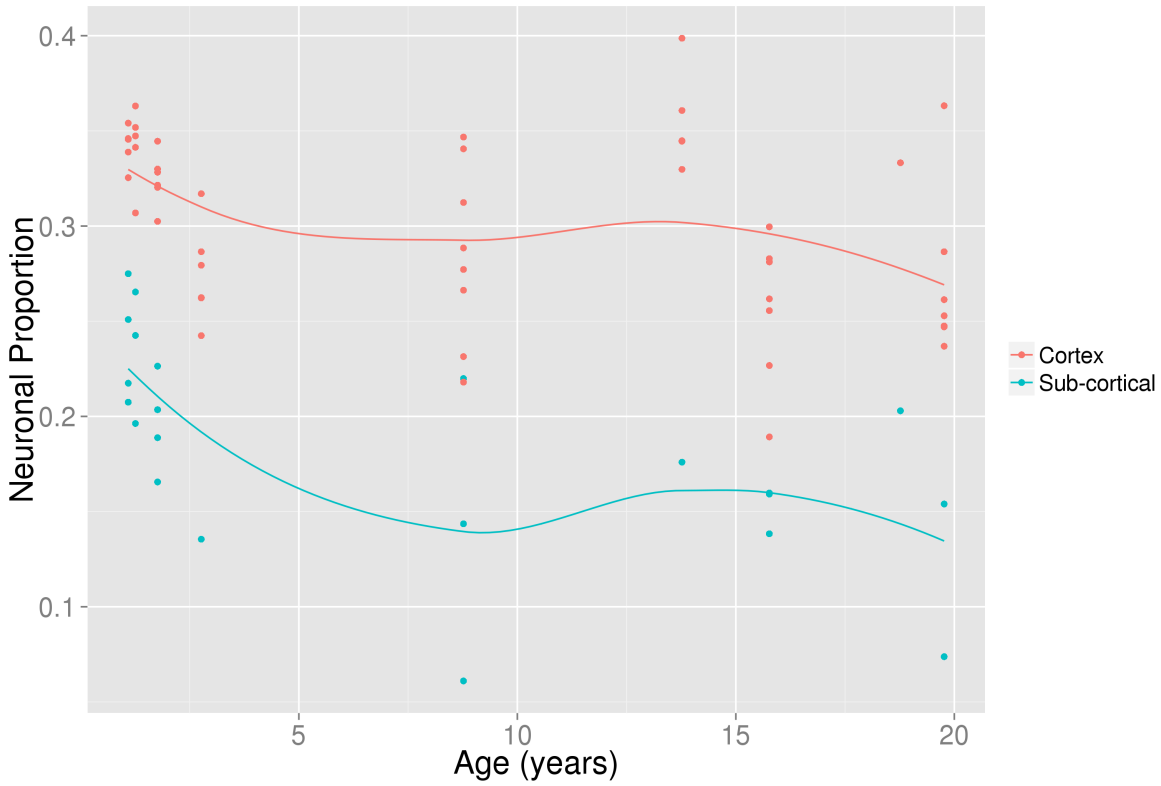

Fig. 20. Estimates of neuronal proportions by age in years in the BrainSpan data

4.2.2 Estimating proportions of microglia

Compared to the other neural cell types in the brain, microglia are relatively small and likely generate less RNA than large neurons. However, they should have a very distinct expression and methylation profiles with many unique immunological genes being solely expressed in microglia. However, unlike in the previous section, we do not have isolated methylation profiles on the Illumina 450k array for microglia. This problem is further complicated by the fact that resting microglia, known as ramified microglia, may have different markers than reactive microglia, which are the active macrophages these resting microglia proliferate and transform into in response to pathogens. In spite of this complication we use Integrin alpha M (ITGAM), also known as CD11b, which is commonly used as a marker for microglia in an attempt to characterize microglia proportions. The ITGAM protein is presnt on the surface of many leukocytes involved in the innate immune system and regulates leukocyte adhesion and migration.

Since ITGAM should be exclusively expressed in microglia, we might expect its promoter to be methylated in all other cell types. If roughly 10-15% of cells are microglia, then this should result in β-values for promoter CpGs in the range of 0.8 to 0.9. We are able to obtain 5 CpG sites from the ITGAM promoter and its CpG sites do fall within this range. If we take $1 - \bar{\beta}$ as a very rough estimate of microglia proportion, where $\bar{\beta}$ is the average promoter β-value for a sample, the average proportion estimate for all samples is 16.3% with a standard deviations of SD = 2.3%. Estimates do not seem to vary much over aging or brain region, but vary somewhat between individuals. Estimates of microglia proportions using methylation data correlated very poorly with ITGAM expresssion $r = 0.061$ which is expressed at moderate levels and slightly increases with age $r = 0.23$. Given these results, it

is difficult to make strong claims about the relative proportions of microglia. More and better markers, or isolated methylation profiles of microglia will greatly improve reliability of these results.

4.3 Overview of developmental BrainSpan data

BrainSpan is a multi-institute consortium devoted to studying the transcriptional mechanisms of human brain development. Not only do they conduct and publish their own research (Kang et al. 2011), but also host what is referred to as the Atlas of the Developing Human Brain (**www.brainspan.org**). BrainSpan provides a wealth of information including imaging, in situ hybridization, exon-level gene expression microarrays, RNA-Seq, as well as methylation microarrays from donated brain samples from 16 brain regions spanning prenatal and developmental periods to old age.

For the purposes of this analysis, we use the publicly available DNA methylation and gene expression data sets. Unfortunately, the DNA methylation microarrays are available only for a subset of the samples at this time, so only nine individuals have both methylation and expression data. Figure 21 provides an overview of available samples. Currently, 87 samples with paired data spanning ages one to twenty years old are available. For the remainder of this section be will perform a brief exploratory analysis for each platform before looking more closely at changes in prefrontal cortex over brain development.

4.3.1 DNA methylation

For an initial exploratory analysis of the DNA methylation data, we perform clustering using multidimensional scaling (MDS) to observe how variable the data are across age and brain region. Methylation was quantified using β-values. Figure 22 gives two MDS plots from autosomal regions of the methylation samples. The left

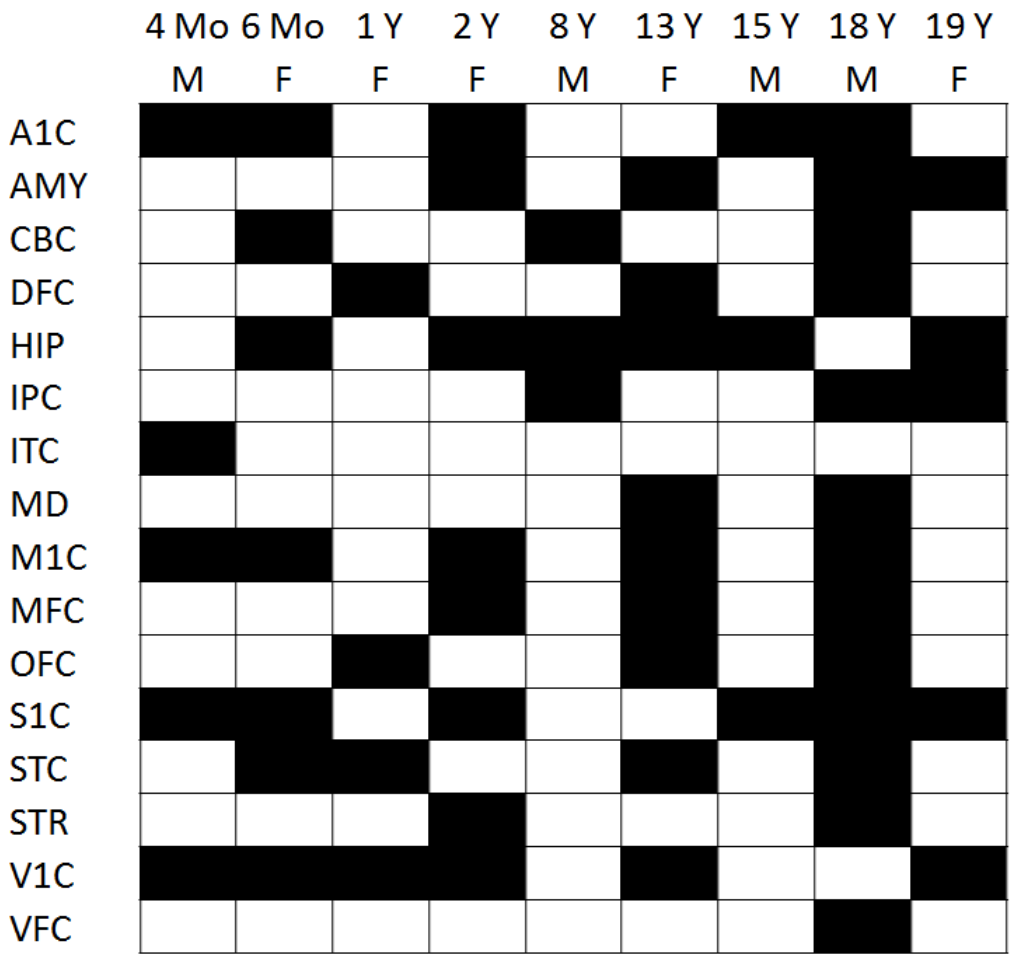

Fig. 21. Publicly available BrainSpan samples that have paired data from methylation and exon-level gene expression. Samples that are not filled in are available. Age and sex are given along the top of the table. Brain regions using abbreviations from Table I are given along the left side.

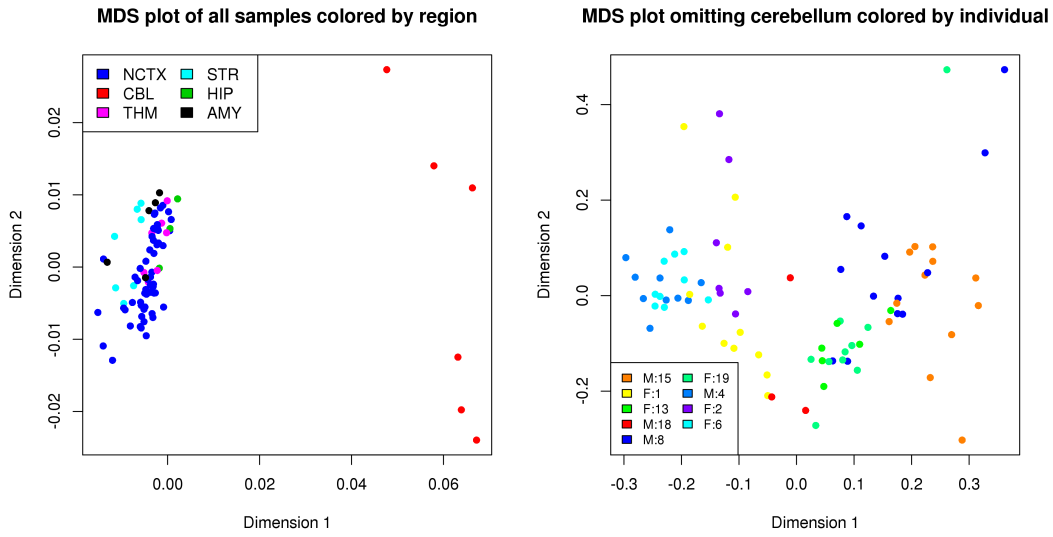

Fig. 22. Multidimensional scaling figures for methylation in the BrainSpan developmental samples. All samples are plotted in the left panel colored by brain region. All samples ommitting cerebellum are plotted in the right panel colored by individual.

panel plots all samples colored by general brain region: neocortex (NCTX), cerebellum (CBL), thalamus (THM), striatum (STR), hippocampus (HIP), and amygdala (AMY). We can see that cerebellum is very distinct from all other samples. The right panel plots samples excluding cerebellum colored by individual. Samples seem to cluster by individual in a way that corresponds with age to some degree, with lower ages tending to the left side of the plot, and older ages to the right side. However, they are not strictly ordered left to right, as both the lowest and highest ages are closer to the middle.

4.3.2 Gene expression

We create a similar figure for aggregate measures of gene expression using multidimensional scaling. Aggregate gene expression measures were computed by taking

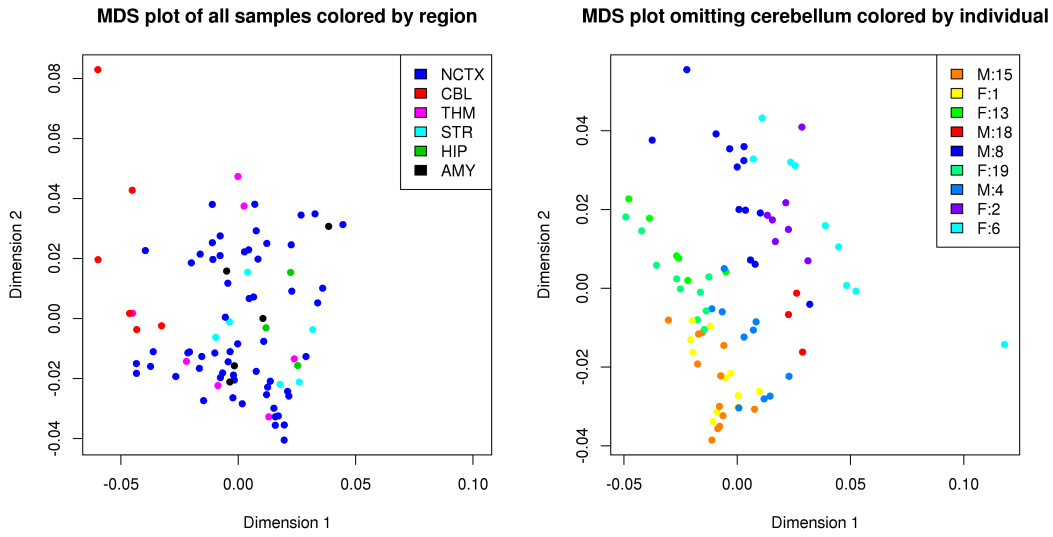

Fig. 23. Multidimensional scaling figures for gene expression in the BrainSpan developmental samples. All samples are plotted in the left panel colored by brain region. All samples ommitting cerebellum are plotted in the right panel colored by individual

the mean of all exon-level RMA summarized signals for each gene in each sample. Figure 23 gives the results. We observe somewhat similar clustering as in the methylation data. In the left panel, cerebellum again appears to be distinct from other samples, but to a lesser degree. When omitting cerebellum in the right panel, the samples again cluster by individual, but in a different way. Specifically, they do not seem to correspond to age or sex in any discernable way.

4.3.3 Exon inclusion

Alternative splicing patterns should not necessarily correspond to differences in gene expression, so it is important to observe how samples cluster by alternative splicing and compare the results to gene expression. Alternative splicing was quantified using the the splicing index from Equation 1.5 using RMA summaries of exon expres-

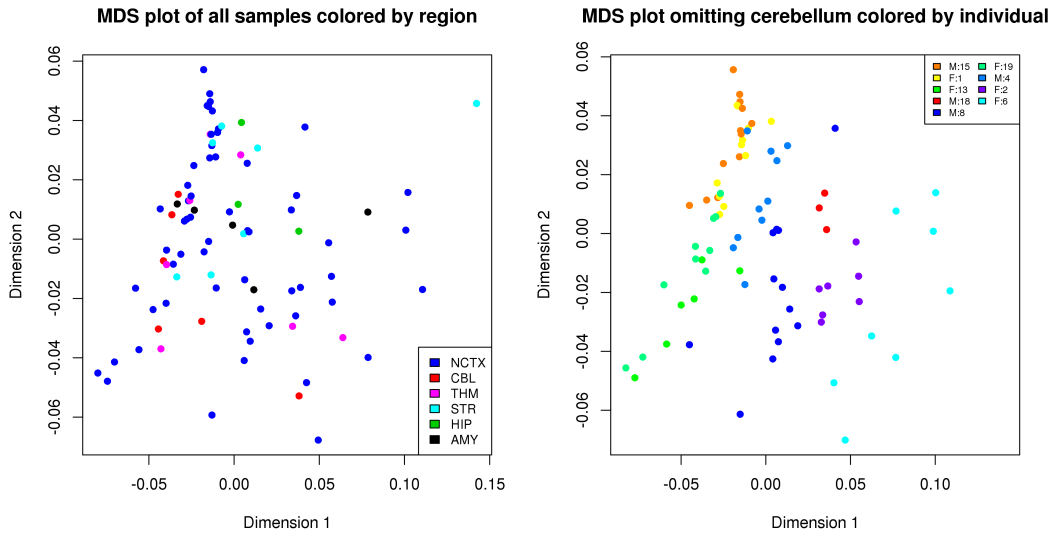

Fig. 24. Multidimensional scaling figures for splicing index in the BrainSpan developmental samples. All samples are plotted in the left panel colored by brain region. All samples ommitting cerebellum are plotted in the right panel colored by individual. Cerebellum is not so distinct as it was in MDS plots of expression

sion and the aggregate gene expression measure as computed in the previous section. Figure 24 gives the resulting MDS plot. In the left panel, we can see that cerebellum now clusters with the rest of the samples and is not distinct as it was before. This suggests that gene expression differences, not differences in alternative splicing, are what make cerebellum distinct. However, in the left plot, individuals cluster almost identically to as before with gene expression.

4.3.4 Brain samples are clustered by individual

One issue in analyzing brain samples over time, as mentioned in Section 1.1.2, is that individual differences are confounded with temporal differences. In the BrainSpan samples, individuals are also sampled at several brain regions which are correlated with each other. In a standard multiple regression setting where the outcome is

univariate, a mixed effects model is sufficient to address this issue. We employ this method when integrating DNA methylation and gene expression. However in the situation of multivariate analysis, techniques such as canonical correlation analysis and partial least squares (PLS) do not have widely used analogous methods for dealing with clustered data. This becomes an issue when integrating alternative splicing and DNA methylation.

A simple approach in the multivariate setting might be to perform a permutation test using the test statistic from the likelihood ratio test proposed in Chapter 3 (Equation 3.9). However, since the samples are not independent, a simple permutation test is not valid as the samples are not fully exchangeable under the null hypothesis. It then might be possible to perform a blocked permutation test, where samples are permuted within individuals to preserve the correlation structure. However, permuting within individuals also preserves the effect of time since time and individual are confounded. Therefore, if we are interested in testing for temporal relationships, a permutation test is not appropriate in this scenario.

In order to get a sense of how individual clustering affects the asymptotic null distribution of the likelihood ratio test statistic from Equation 3.9, we simulate data and create an empirical null distribution. This is done by simulating the first three principal components of both DNA methylation and exon inclusion from a multivariate normal distribution. If we assume for simplicity that all principal component scores follow the same multivariate normal distribution under the null hypothesis (which is somewhat reasonable), we can estimate the parameters for the multivariate normal distribution from the data. The mean vectors for principal components are effectively zero since they are constrained to sum to one, so they are simply set to zero in simulations. All that remains then is to estimate covariance matrices for each of the first three principal components from methylation and alternative splicing.

Covariance matrices are estimated using the following procedure:

- Perform principal component analysis on a gene-by-gene basis for both DNA methylation and exon inclusion for all genes.
- Retain the first three sets of PC scores from each gene for both DNA methylation and exon inclusion.
- For each principal component, estimate a single covariance matrix using that principal component's scores from all genes.
- For each covariance matrix, set all entries corresponding to covariance *between* different individuals equal to zero.

The resulting set of covariance matrices obtained from this procedure can then be used to simulate sets of principal component scores that are clustered within individuals similar to observed data, but with no significant correlation *between* individuals. Principal components from methylation are independent to those from expression. A second simulated null distribution is computed using only the diagonal terms from the covariances to simulate the scenario with no clustering, but with the same variances. Densities of 10,000 simulated test statistics from these two scenarios are plotted against the theoretical χ^2_9 null distribution in Figure 25. We can see that the distribution simulated under independence is well-approximated by the χ^2_9 distribution. However, the distribution simulated with clustering has a substantially heavier tail.

This heavier tail is a result of an over-representation of the effective sample size in the test statistic. This means that since samples are not independent, we get a smaller amount of total variability in the data than would be expected from 26 independent samples. Therefore, we effectively have fewer than 26 samples when computing the

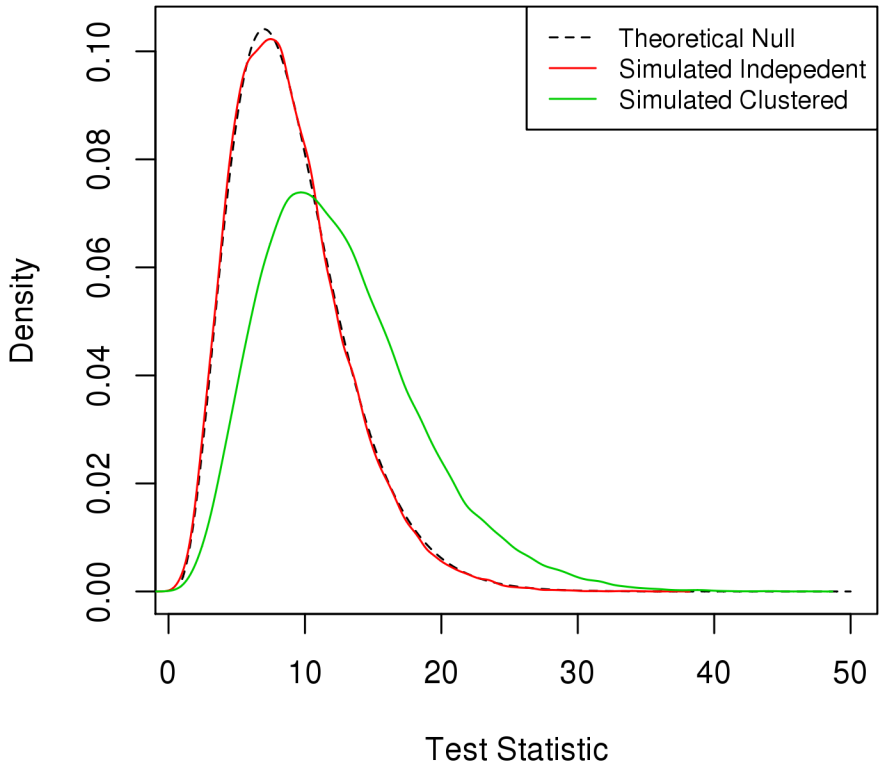

Fig. 25. Densities of 10,000 test statistics simulated from a clustered null distribution. The theoretical χ^2_9 distribution is given as the dashed line. In the case of independence (red), the simulated distribution is well-approximated by the theoretical. In the case of clustered data, the simulated distribution has a much heavier tail

likelihood ratio test statistic. To account for this we can adjust the effective sample size used in the test statistic. To do this we can fit a zero-intercept linear model to compute a scaling factor θ as given in Equation 4.1 modeling the i^{th} quantile of the theoretical distribution χ^2_{9i} as a function of the i^{th} quantile of the clustered simulated distribution $\hat{f}_i$.

$$\chi^2_{9i} = \theta \hat{f}_i + \epsilon_i \tag{4.1}$$

After fitting the linear model, we get an estimate of $\theta = 0.7145$. We can do a little algebra and solve for n^* in Equation 4.2 to obtain the effective sample size. The right side gives the formula for the corrected sample size used by Bartlett 1939 with the estimated scaling factor θ applied where the number of parameters (principal components) $p = q = 3$ and $n = 26$. The left side gives the formula, but with the effective sample size n^* rather than using the scaling factor θ.

$$n^* - 1 - \frac{1}{2}(1 + p + q) = \theta \left(n - 1 - \frac{1}{2}(1 + p + q)\right) \tag{4.2}$$

Solving for n^* gives an effective sample size of $n^* = 19.55$. If we replace the original sample size in the correlated data with the adjusted effective sample size, the χ^2_9 approximation becomes appropriate. Figure 26 gives the density of the test statistics simulated from correlated data using n^* instead of n. The theoretical χ^2_9 distribution is included as reference. For further analyses, we can instead use the effective sample size ($n^* = 19.55$) rather than the actual sample size ($n = 26$) when conducting the likelihood ratio test from Equation 3.9.

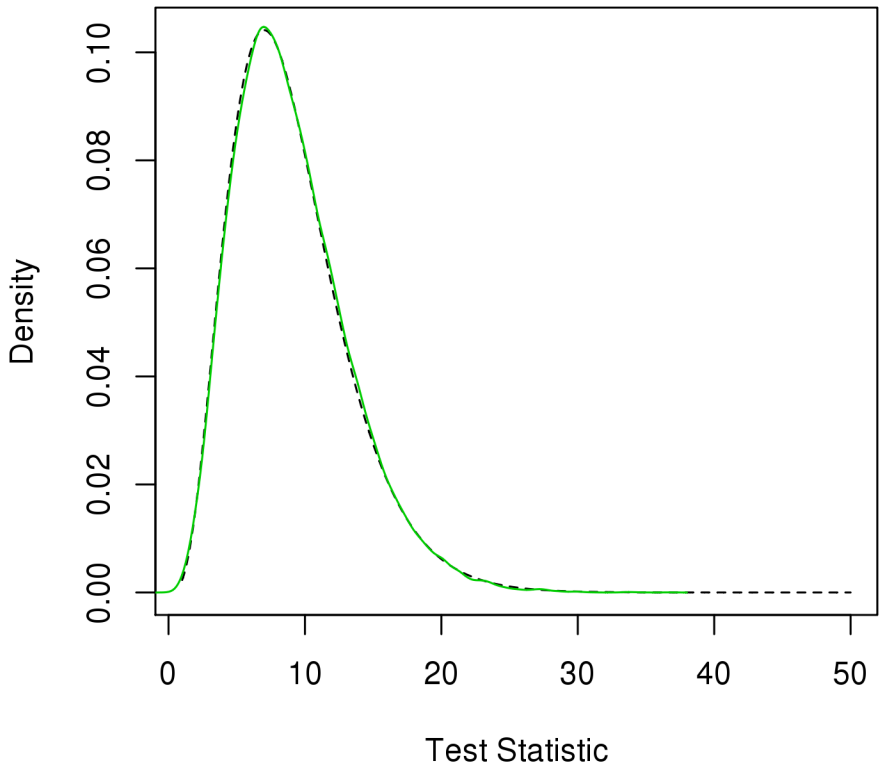

Fig. 26. Density of test statistics simulated from correlated data using the effective sample size $n^* = 19.55$. Once the effective sample size is used, the χ^2_9 approximation becomes appropriate.

4.4 Integrating exon inclusion and DNA methylation

To analyze the relationship between alternative splicing and DNA methylation in developing prefrontal cortex we apply a multi-step exploratory approach. First, we test for overall association using the likelihood ratio test from Equation 3.9. We then use the permutation test on R^2 matrices to test for associations that co-localize on the gene more than would be expected by chance. Finally we use the mixed effects linear model from Equation 3.13 including both linear and quadratic effects for age to test whether associations in the first set of canonical covariates occur across development. Finally, we perform a reanalysis using methylation data that has been adjusted for differences in tissue admixtures estimated in Section 4.2.1 to demonstrate that significant findings are unlikely to be confounded with differences in proportions of neural cell types.

As mentioned previously in Section 4.3.4, the 26 brain samples used in this analysis are clustered by individual and cannot be treated as independent. However, independence is an an assumption made by the likelihood ratio test from Equation 3.9 makes. To address this issue, we use an adjusted sample size of 19.55 when computing test statistics. Figure 27 gives results from the likelihood ratio test on all genes using both the original sample size as well as the adjusted effective sample size computed in Section 4.3.4. The distribution of p-values obtained by using the original sample size in the left panel are anti-conservative. However, when using the adjusted effective sample size, the resulting p-value distribution in the right panel is much more reasonable and some significance is still retained.

Once we have selected a set of significant genes, we can then ask whether the association between alternative splicing and DNA methylation seems to co-localize to the same regions on the genome. Figure 28 gives the density of permutation p-values

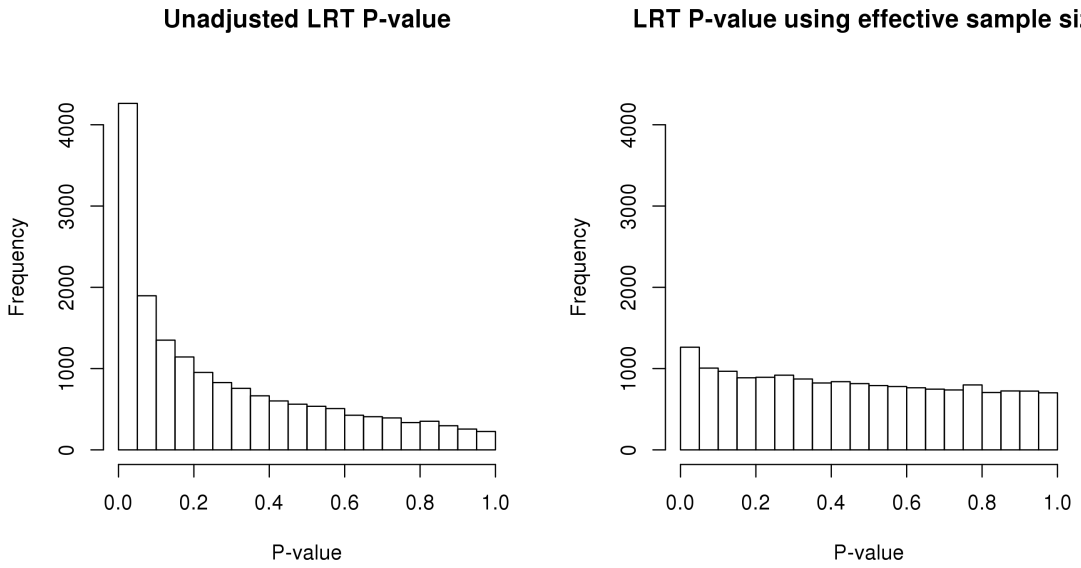

Fig. 27. Histogram of p-values from likelihood ratio test for association. P-values using the original sample size of 26 are given in the left panel. P-values using the effective sample size of 19.55 are given in the right panel.

obtained using the R^2 matrix in the left panel and plots the $-\log_{10}$(p-values) from the permutation test against the $-\log_{10}$(p-values) from the likelihood ratio test. A small offset of 10^{-6} was added to permutation p-values so permutation p-values equal to zero could be plotted. Since both the permutation test and likelihood ratio test seem to be somewhat under-powered due to the small sample size, we use a joint threshold of $p < 0.01$ for both the likelihood ratio test and permutation test to selection a set of genes that seem to have significant, *cis*-acting relationships between methylation and splicing. Many of these genes seem to have changes that may be associated with brain development, too. Table VI gives a list of these genes along with p-values and a brief description.

While several of the genes from Table VI seem to be associated with aging, there may be genes that are have a significant result from the likelihood ratio test

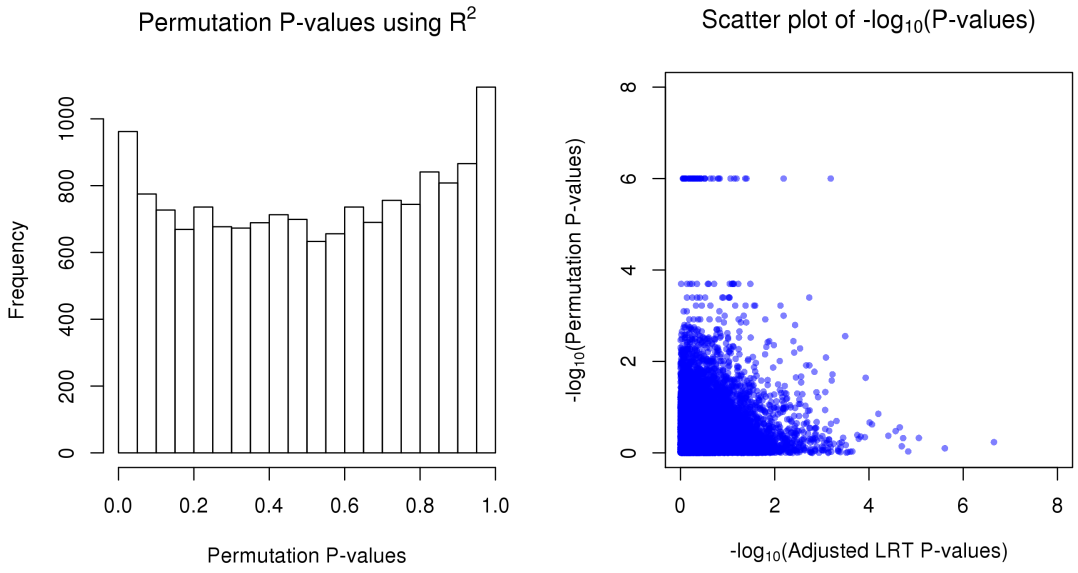

Fig. 28. Results from permutation testing using R^2 values between CpG sites and exons. The left panel gives the histogram of permutation p-values. The right panel plots $-\log_{10}$(permutation p-values) against $-\log_{10}$(LRT p-values). A small offset of 10^{-6} was added to permutation p-values to plot p-values equal to zero.

Table VI. Genes meeting threshold for significance from LRT and permutation test

Gene	LRT P-val	Perm P-val	Mixed Effects P-val	Details
CNTNAP2	0.000320	0.0028	0.109643	Contactin-associated protein-like 2
RPL10	0.000648	0.0000	0.176863	60S ribosomal protein L10
ST18	0.004022	0.0036	0.013308	Suppression of tumorigenicity 18 protein
NFIA	0.006474	0.0010	0.005944	Nuclear factor 1 A-type
KALRN	0.006474	0.0000	0.045387	Kalirin
DIO2	0.001853	0.0004	0.002720	Type II iodothyronine deiodinase
IL32	0.007823	0.0098	0.155751	Interleukin-32
TMEM144	0.000817	0.0082	0.003309	Transmembrane protein 144
ROBO1	0.003698	0.0016	0.052201	Roundabout, axon guidance receptor, homolog 1
URB1	0.007552	0.0006	0.950762	Nucleolar pre-ribosomal-associated protein 1
NLRP2	0.002891	0.0052	0.329450	NACHT, LRR and PYD domains-containing protein 2
ZNF229	0.003865	0.0064	0.002221	Zinc finger protein 229

Table VII. Genes meeting threshold for significance from LRT and mixed effects LRT

Gene	LRT P-val	Perm P-val	Mixed Effects P-val	Details
RNF220	.000086	0.2382	0.002754	Ring finger protein 220
FMN2	.000019	0.4770	0.003756	Formin 2
HHATL	.000097	0.2162	0.000445	Hedgehog acetyltransferase-like
RNASE1	.000039	0.4228	0.000918	Ribonuclease, RNase A family, 1

that are also significantly associated with age, but do not have a detectable *cis*-acting relationship between DNA methylation and alternative splicing. This may be because of incomplete coverage by the methylation microarray, or the relationship may just be pervasive. Figure 29 gives the p-value histogram for the likelihood ratio test from the linear mixed effects model in the left panel. P-values are more conservative than those obtained from an ordinary least squares linear model, but are still highly anti-conservative, which can be a shortcoming of the approach (Schielzeth and Forstmeier 2008). The right panel plots $-\log_{10}$(p-values from the likelihood ratio test from the linear mixed effects model against the likelihood ratio test for general association. Table VII gives a list of genes meeting an FDR = 0.1 threshold for both the likelihood ratio test for association and the linear mixed effects model.

As mentioned in Section 4.2.1, cerebral cortex is composed of several cell types that may vary both across individual and aging. Therefore, many of these apparent changes may be reflecting differences in cell proportions across samples rather than real changes in methylation. To address this issue we perform a reanalysis using mixture-adjusted methylation data. This reanalysis is performed by regressing out the average neuronal and non-neuronal methylation profiles from Kozlenkov et al. 2013 which is equivalent to using the residuals from the linear model given in Equation 2.4. Residuals from this linear model should reflect variation in methylation that cannot be attributed to differences in neuronal proportion.

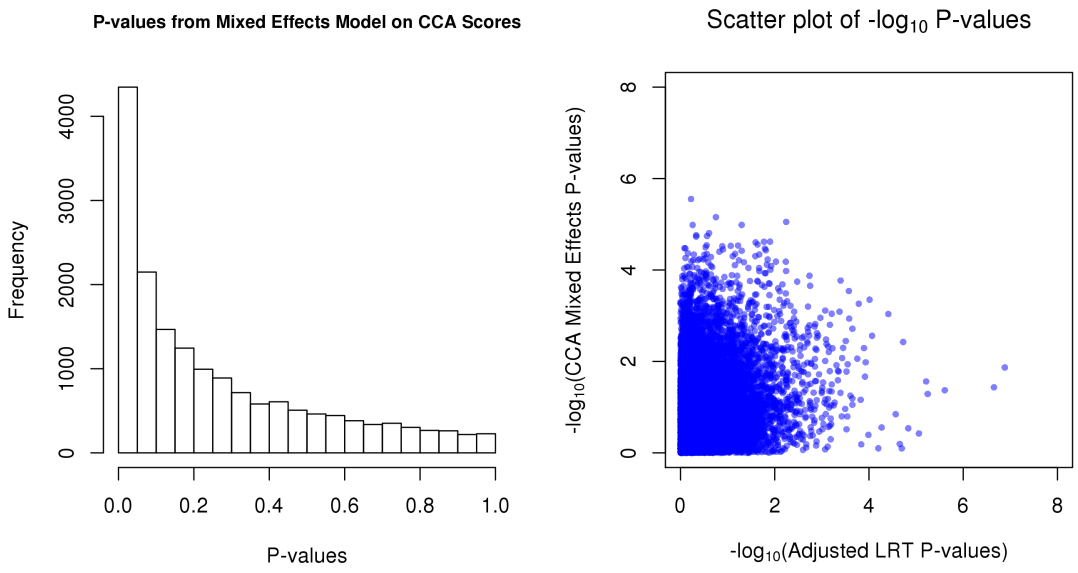

Fig. 29. Results from mixed effect model on canonical covariate scores. P-values from the likelihood ratio test for nested models is given in the left panel. The χ^2 approximation to the likelihood ratio is anti-conservative. In the right panel $-\log_{10}$(p-values from the mixed effects LRT are plotted against $-\log_{10}$(p-values from the LRT for association of methylation and splicing index.

It is more difficult to adjust for differences in gene expression and alternative splicing. While each cell contributes an equal amount of DNA, and therefore an equal amount of methylation signal, neurons are relatively larger cells compared to glial cells and contribute more RNA. Also, many marker genes of neurons lie in synapses which can vary without the actual neural proportion changing. For now, we are limited to adjusting only the methylation data for proportions. Figure 30 gives the results from using the likelihood ratio test with the adjusted methylation residuals. The left panel gives a histogram of p-values which is very similar to the one using the original β-values. The right panel plots $-\log_{10}$(p-values) using mixture adjusted residuals against the $-\log_{10}$(p-values) from the original likelihood ratio test. There does not appear to be a large change in significance, with the exception of one gene that becomes highly significant after adjustment: PRRC1, a gene that is highly expressed in the brain and has been recently implicated as having an effect on liquid intelligence over brain development (Rowe et al. 2013). We will elaborate more on this gene in the following section.

4.5 Detailed analysis of specific genes

Results from the previous section have highlighted specific genes as having spatially co-localized relationships between alternative splicing and DNA methylation that co-vary over time. However, this still does not specify the exact nature of the relationship. In this section we use the results from canonical correlation analysis to interpret loadings of specific genes and try to make claims about how exon inclusion and DNA methylation may be related in these genes over brain development. It appears more likely that observed significant genes are a result of alternative promoter usage rather than alternative splicing. This makes sense since alternative promoter usage should be easier to detect since it can affect multiple exons, making it easier

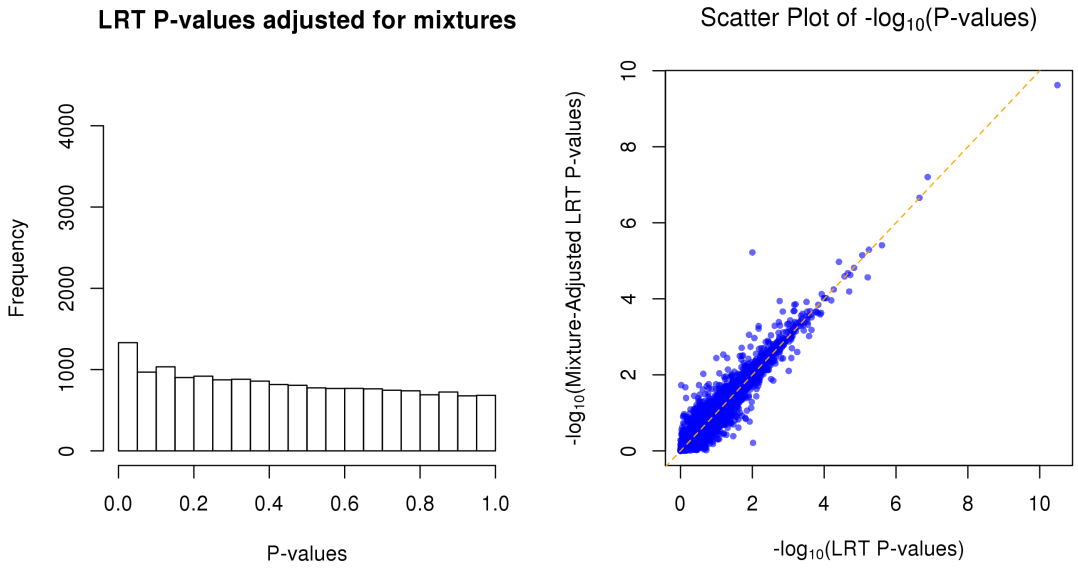

Fig. 30. Results from likelihood ratio test for association of methylation and splicing index after adjusting for neuron proportions in the methylation data. The left panel gives the p-value histogram. The right panel plots $-\log_{10}$(p-values) from the mixture adjusted test against the $-\log_{10}$(p-values) from the original.

to detect. We are likely missing many slight changes in splicing patterns, especially with a sample size of 26.

4.5.1 Kalirin

Kalirin (KLRN), also known as Huntingtin-associated protein-interacting protein (HAPIP), was first identified in 1997 as a protein interacting with huntingtin-associated protein 1 (Colomer et al. 1997). Is also known to play an important role in nerve growth and axonal development (Chakrabarti et al. 2005). It is named after the multiple-handed Hindu goddess Kali for its ability to interact with numerous other proteins. The predominant isoform of Kalirin, Kalirin-7, was found to be necessary for the remodeling and growth of synapses and dendritic spines in mature cortical neurons and is thought to be important in the development of schizophrenia (Xie et al. 2007).

Kalirin is highly expressed in the brain and has mean gene expression greater than roughly 85% of genes in samples assayed. Expression levels dip slightly during early childhood, but begin to rise again after 9 years of age. Figure 31 gives the expression trajectory. Figure 32 gives loadings for the first set of canonical covariates and their trajectory over age. Splicing patterns of Kalirin seem to follow expression closely. Redundancy coefficients (RCs) for methylation β-values and splicing indices given in the right panel show that roughly 31% of total methylation varibility and about 10% of splicing variability in this gene is explained by the first set of canonical covariates. An increase in a specific methylation site in an intron seems to correspond to decreased expression and preferential expression of a shorter isoform of the gene. Closer inspection reveals that this CpG is located in a known enhancer for this gene.

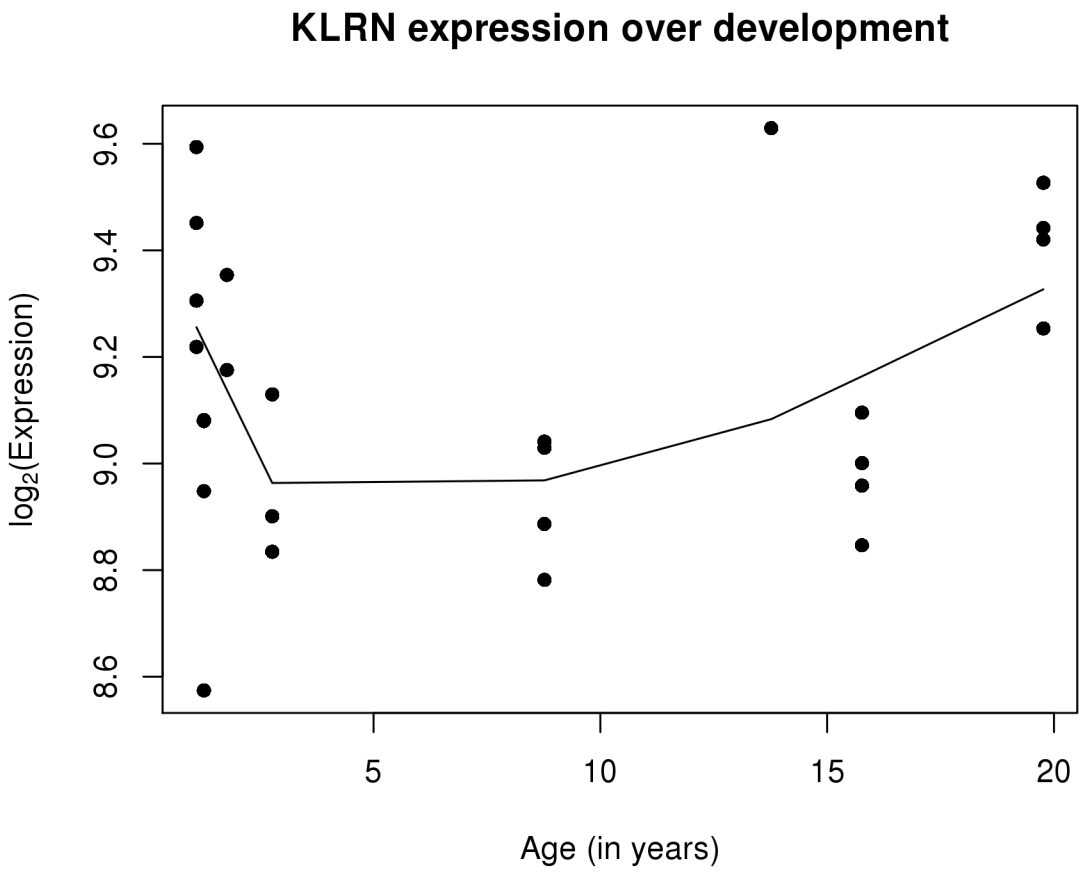

Fig. 31. $\log_2$(Gene Expression) profile of Kalirin over age

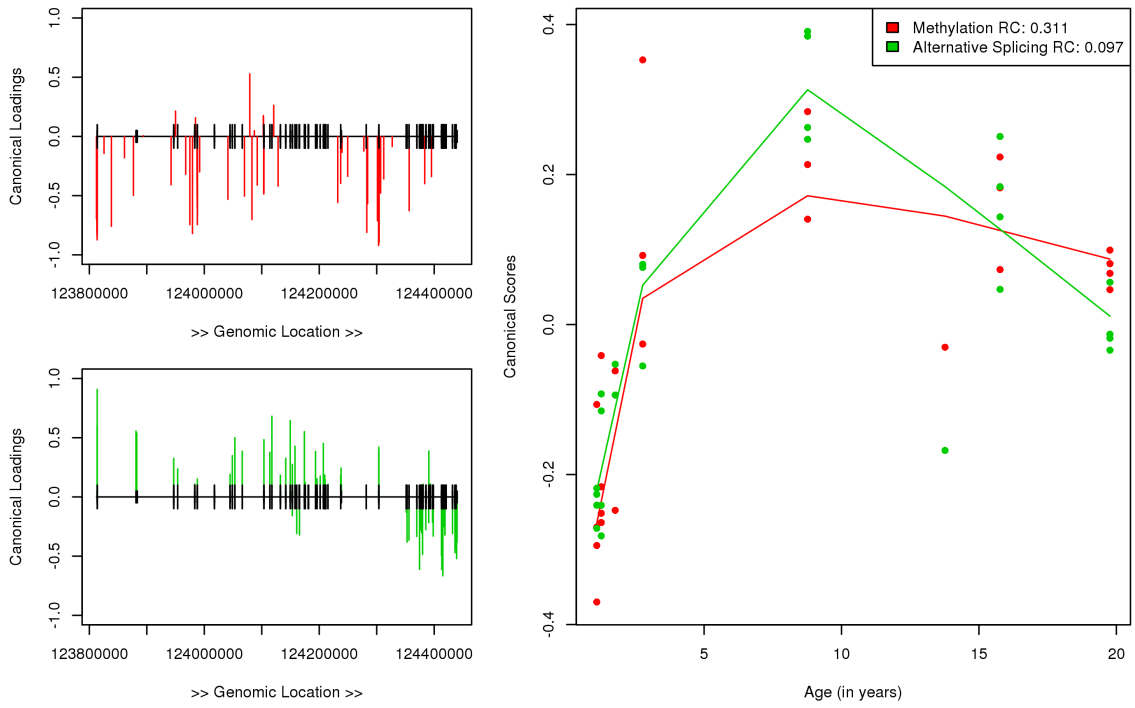

Fig. 32. Splicing pattern in Kalirin over age given by the first set of canonical covariates. A decrease in methylation in an enhancer in an intronic region in the middle of the gene results in increased expression of a shorter version of the gene

4.5.2 Chimerin 2

Chimerin 2 (CHN2) is a nerve tissue protein that has GTPase-activating protein activity that is regulated by phospholipid binding and binding of diacylglycerol induces translocation of the protein from the cytosol to the Golgi apparatus membrane. Many variants arising from alternative splicing have been characterized. A missense mutation of Chimerin 2 has been associated with schizophrenia in men (Hashimoto et al. 2005).

Chimerin 2 is highly expressed in the brain at similar levels to Kalirin. It also follows a similar expression trajectory as given in Figure 33. However, unlike Kalirin, the splicing trajectories obtained from the first set of canonical covariates given in Figure 34 do not follow the expression trajectory. The gene seems to be generally losing methylation over age, but particularly at a specific exon towards the end of the gene which also functions as an alternative start site. Demethylation of this site seems to correspond to increased expression of a shorter version of the gene that starts transcription there and perhaps terminates sooner.

4.5.3 Roundabout homolog 1

Roundabout homolog 1 (ROBO1) encodes an integral membrane protein that is both an axon guidance receptor and a cell adhesion receptor. It is specifically involved in long range axon guidance when axons decide to cross the central nervous system (CNS) midline. A translocation in ROBO1 was implicated in communication disorder based on a Finnish pedigree with severe dyslexia (Bates et al. 2011).

ROBO1 is also highly expressed in the Brain, with a minor dip in expression occuring after 2 years of age that rebounds after 9 years of age (Figure 35). Due to limited microarray coverage, we are only able to assay methylation status from the

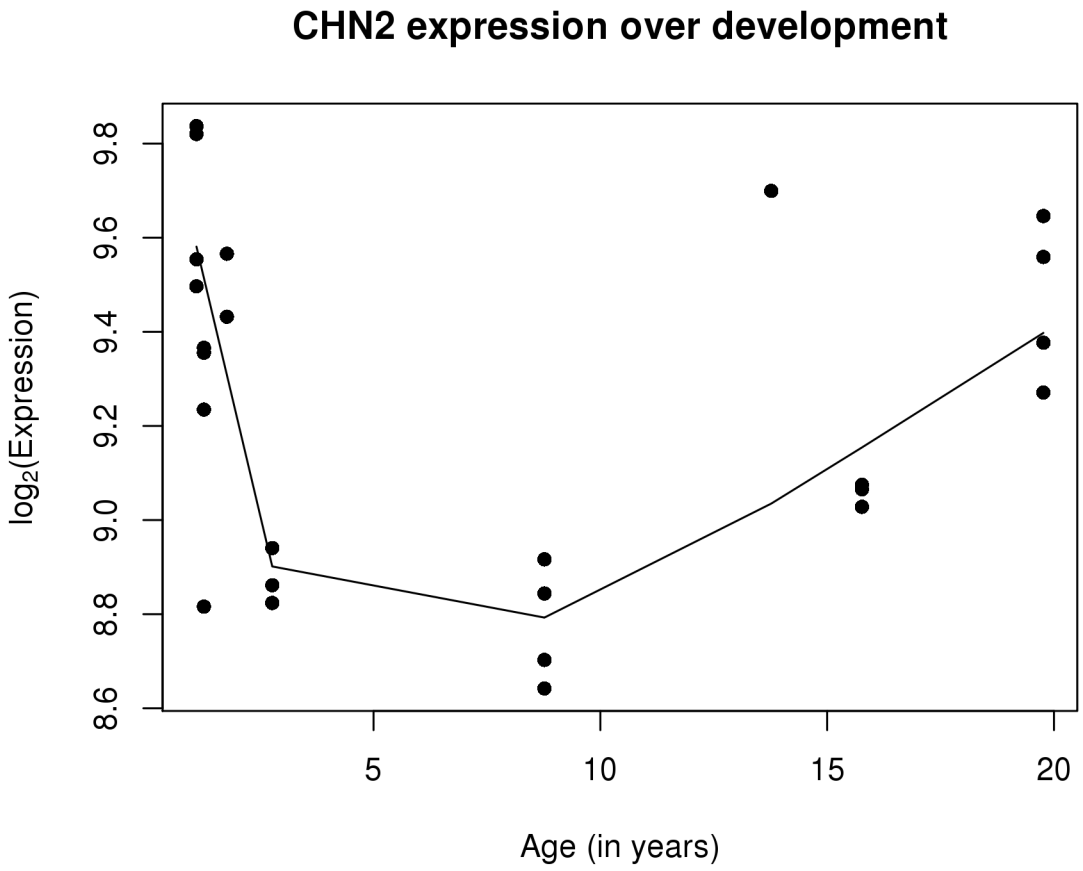

Fig. 33. $\log_2$(Gene Expression) profile of Chimerin 2 over age

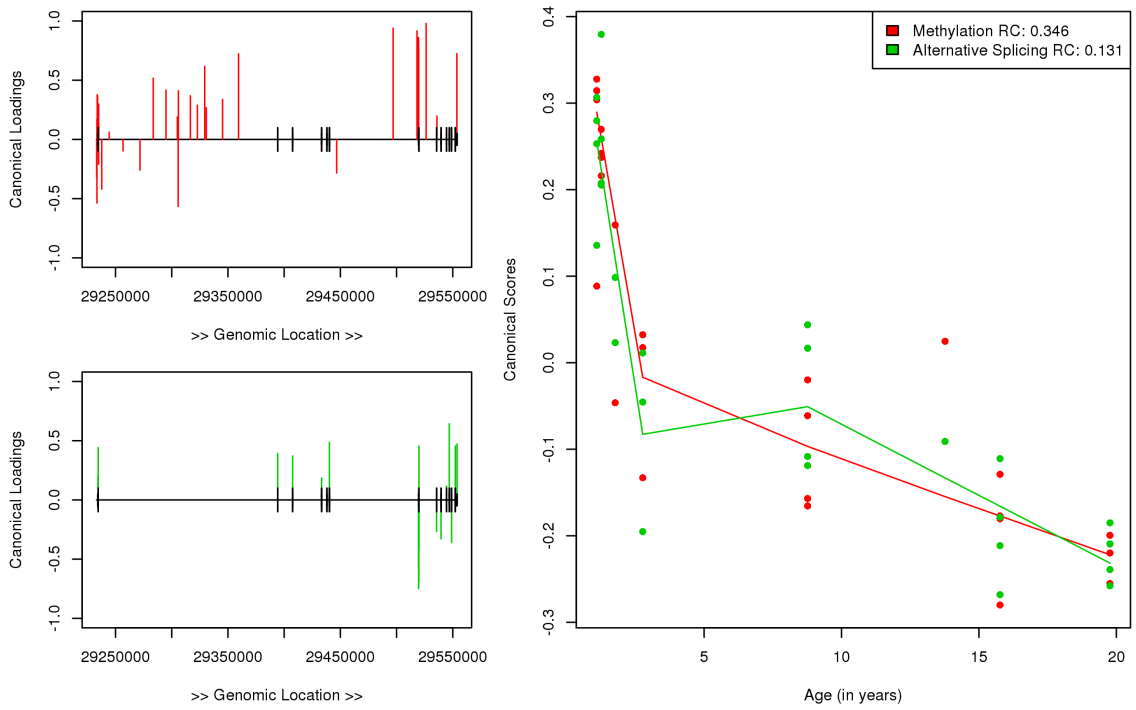

Fig. 34. Splicing pattern in Chimerin 2 over age given by the first set of canonical covariates. General loss of methylation, particularly at an alternative start site near the end of the gene results in increased expression of a shorter isoform.

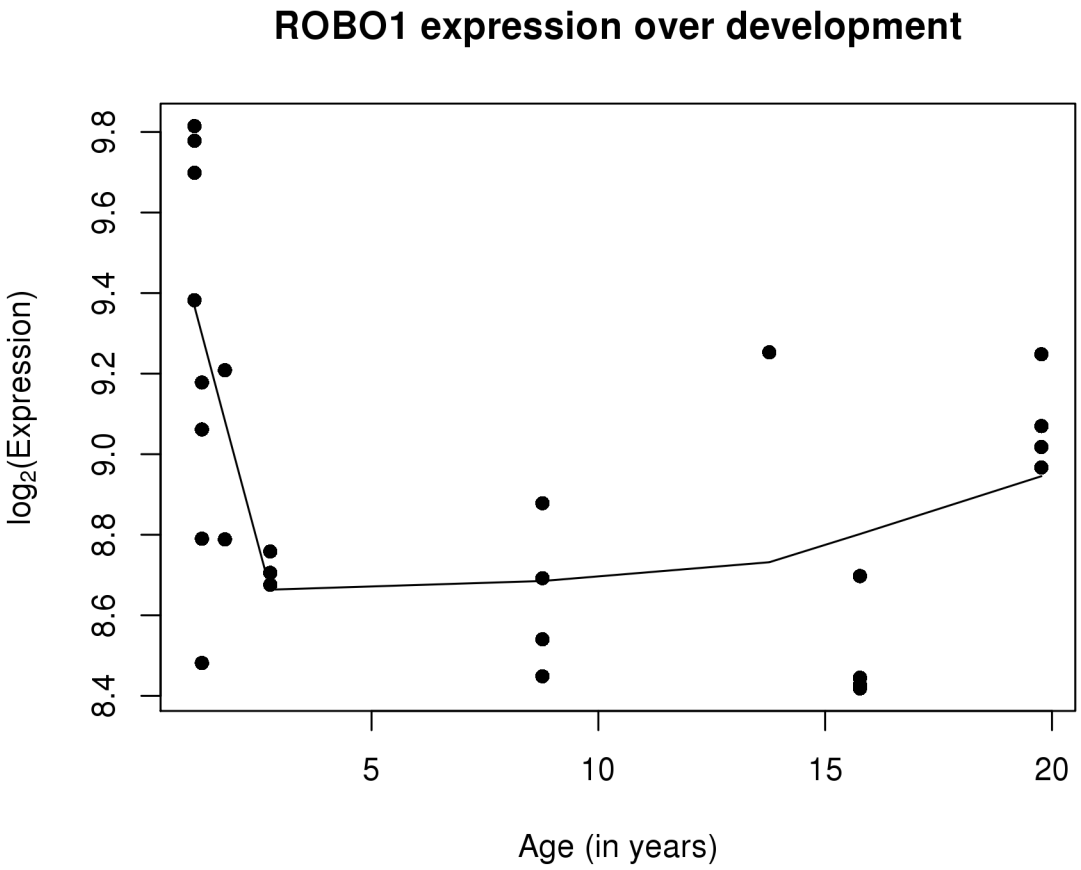

Fig. 35. $\log_2$(Gene Expression) profile of Roundabout homolog 1 over age

first upstream half of the gene. However, from Figure 36, we can see the highest loadings for methylation correspond to the promoter region at the beginning of the gene. Moderately high loadings also appear at the middle exon with the highest loading for exon inclusion. This exon is a known alternative start site for this gene. Therefore, it appears that over brain development, a shorter isoform of this gene starting from this alternative start site is being preferentially transcribed. This phenomenon does not correspond similarly to changes in the expression profile.

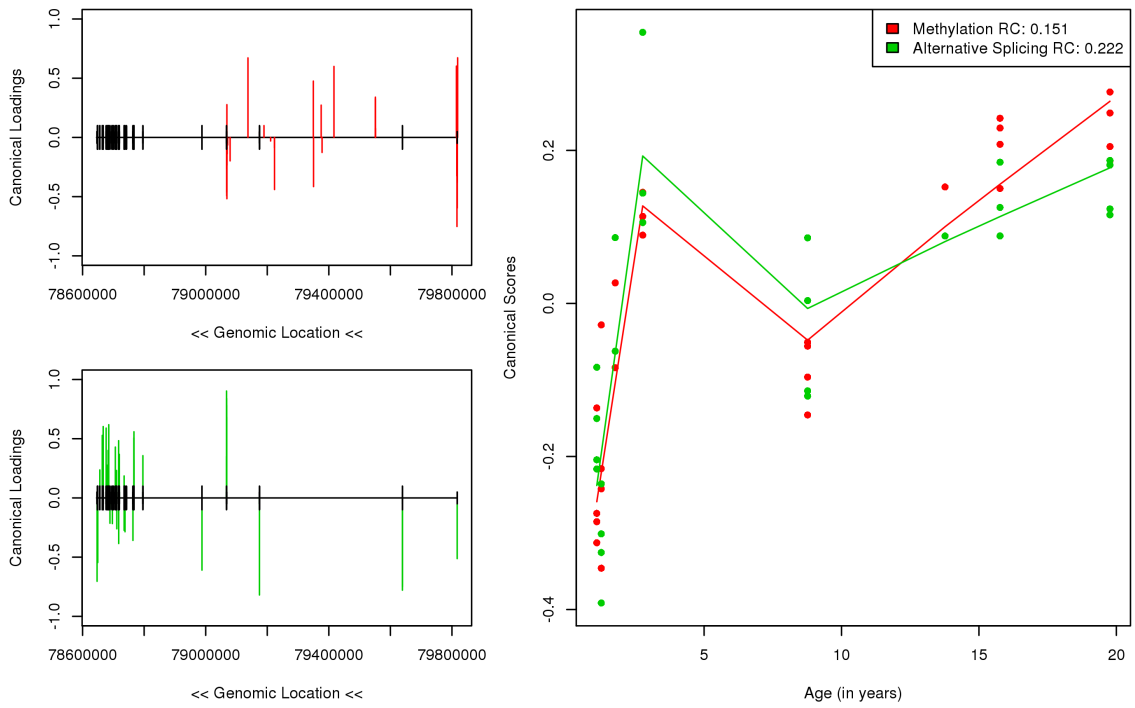

Fig. 36. Splicing pattern in Roundabout homolog 1 over age given by the first set of canonical covariates. Differential methylation at two different transcription start sites results in an increased expression of a shorter gene isoform.

4.5.4 Proline-rich coiled-coil 1

Proline-rich coiled-coil 1 (PRRC1) is a golgi-associated protein with unknown function in the brain. A recent study has implicated PRRC1 as potentially affecting fluid intelligence in humans (Rowe et al. 2013). PRRC1 is currently not otherwise well-studied or understood.

PRRC1 is highly expressed in samples observed, but at lower levels than previous genes mentioned. Unlike the other genes, PRRC1 seems to be consistently increasing in expression over age (Figure 37). The Illumina 450k array provides limited coverage of the gene body of PRRC1. However, a decrease in observed promoter methylation correlates with the increase in expression.

4.6 Summary

In Chapter 4, we have given a brief overview of necessary neurobiology and issues particular to the analysis of neurogenomic data. We have used isolated methylation profiles from Kozlenkov et al. 2013 to estimate the relative abundance of neurons over brain development. We have performed exploratory analysis and found that DNA methylation, gene expression, and splicing index all tend to cluster most strongly according to individual in a way that is not directly related to aging. We have adapted the methods introduced in Chapter 3 that assume independence of samples to account for this clustering. While somewhat underpowered, our analysis was able to detect a subset of genes that had statistically significant, co-localizing relationships between DNA methylation and splicing index. These genes tended to be involved in axon guidance. Due to the small sample size, most significant findings appear to be generally the result of alternative promoter usage rather than differential usage of single cassette exons.

Fig. 37. $\log_2$(Gene Expression) profile of Proline-rich coiled-coil 1 over age

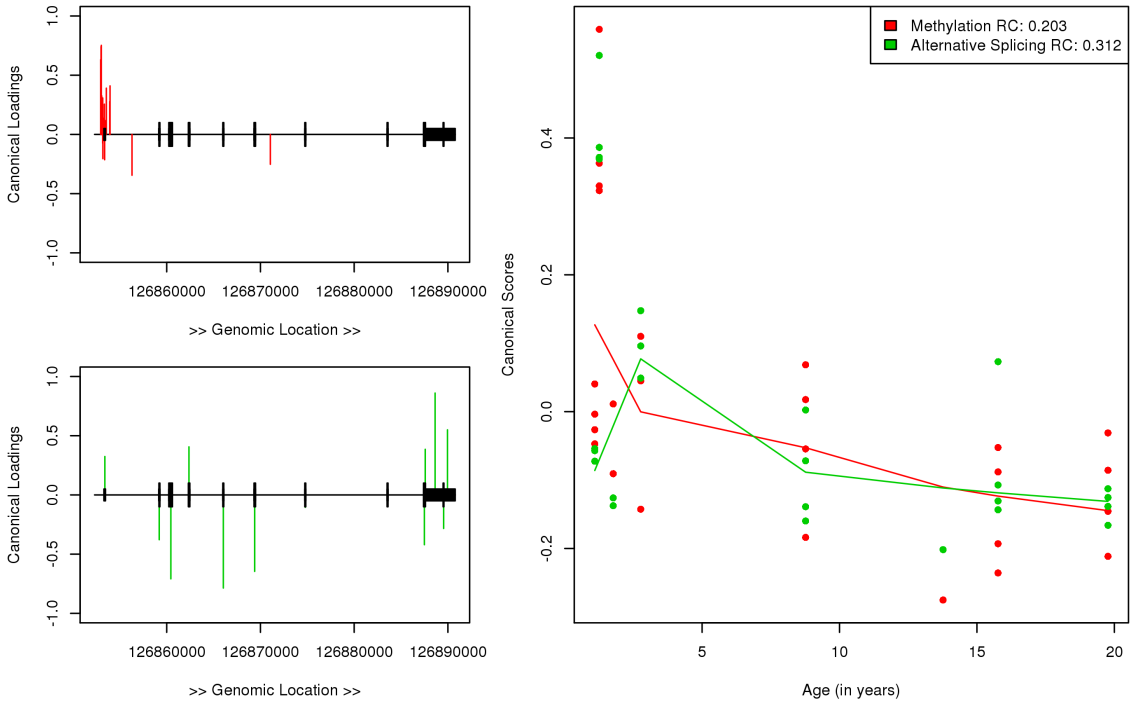

Fig. 38. Splicing pattern in Proline-rich coiled-coil 1 over age given by the first set of canonical covariates. A decrease in promoter methylation corresponds to increase gene expression and perhaps preferential usage of a longer 3′ UTR

CHAPTER 5

INTEGRATIVE ANALYSIS OF STANLEY BRAIN SAMPLES

Schizophrenia and bipolar disorder are widely believed to be heritable complex traits with potentially thousands of SNPs contributing to the phenotype (Lee et al. 2012; McGuffin et al. 2003; Sullivan, Kendler, and Neale 2003). However, genome-wide association studies using common SNPs have been able to account for only a fraction of total heritability (Dongen and Boomsma 2013). There are several plausible explanations for this discrepancy: underpowered studies, heterogenous phenotypes, rare unobserved SNPs, epigenetic factors such as DNA methylation, or probably some combination of several of these things. In this chapter our goal is to detect differences between schizophrenics, bipolars, and neurotypical controls that co-occur across three data modalities taken from brain samples from the Stanley Medical Research Institute brain tissue repository:

- RNA-Seq measurements of genome-wide gene expression
- MBD-Seq measurements of genome-wide DNA methylation
- Imputed genotypes from roughly 16 million SNPs

The Stanley Medical Research Institute (SMRI) is a nonprofit organization supporting research on the causes of, and treatments for, schizophrenia and bipolar disorder. SMRI houses a repository of post-mortem brains taken from schizophrenic and bipolar patients as well as neurotypical controls that has been widely used in hundreds of publications in psychiatric research (**http://www2.stanleyresearch.org**).

In this chapter, we are again interested performing an integrative analysis to find genes with changes in expression that have corresponding changes in DNA methylation. There are a couple of points worth highlighting in this analysis that differ from Chapter 4. In this scenario, the primary focus is comparing *groups* of patients rather than looking at differences across age. Second, sequencing data provides richer coverage of sites in the genome, but presents different challenges for summarization and normalization than microarrays. Both statistical and bioinformatic methods will be detailed in the following sections.

5.1 Overview of data

5.1.1 DNA methylation

MBD-Seq was performed on 100 samples obtained from dorsolateral prefrontal cortex. Two different sequencing protocols were used: most batches were sequenced using 50 bp reads, while batches five and six used longer 75 bp reads. After sequencing, samples were aligned to the hg19 human reference genome. Since the MBD protein can bind upstream or downstream of the actual methylated locus, aligned reads were extended to 250 bases to allow for the imprecision of MBD protein binding. Reads were binned in 300 base intervals. An equal fraction of each non-uniquely mapped read was counted toward each of its possible map sites. Since only a subset of the human genome contains CpG sites, many intervals had counts close to or equal to zero. All intervals with mean counts ≤ 10 across all samples were filtered out. Remaining intervals were scaled by total sample read depth and multiplied by one million to obtain a final measure of methylation for each interval given in "reads per million" (RPM).

There was an issue with mismatched samples during initial sample processing,

so there are fewer than one hundred unique samples and several technical replicates. After correctly identifying samples, there were 76 unique samples with 16 having a technical replicate and one sample being done in triplicate. Four samples were not identifiable and were omitted, and one sample has been identified by SMRI as having an unknown phenotype. This resulted in a total of 95 samples in total.

After data summarization and filtering, samples were clustered using multidimensional scaling using intervals from annotated genic regions in Figure 39. Points are given as batch numbers and are colored by sequencing protocol in the left panel. Samples cluster strongly by batch. The earlier four batches seem to be more variable and cluster separately from the rest of the data. Samples from batches 1 through 4 in the top left corner of the MDS plot have been identified as samples with lower DNA concentrations. Due to the relatively poorer quality of the first 4 batches, a secondary analysis is also performed omitting batches 1 through 4. The right panel of Figure 39 colors samples by disease phenotype. We can see that there is some degree of confounding of disease phenotype with batches. A MDS plot showing only the later batches is given in Figure 40. Samples in later batches don't seem to clearly cluster by batch or disease phenotype.

Before performing an integrative analysis, we first perform an initial exploratory analysis using the MBD-Seq data. For each interval that met the filtering criteria, a one-way ANOVA for disease phenotype is fit. After p-values are obtained, the false-discovery rate is controlled at FDR= 0.1 using the Benjamini-Hochberg method (Benjamini and Hochberg 1995). Figure 41 gives the resulting p-value histograms when using all samples (left panel) and the subset of samples from later batches that were deemed to be of higher quality (right panel). In both scenarios, no intervals were significant at FDR = 0.1. Non-uniform p-value distributions are likely the result of a combination of confounding batch effects and correlated tests of nearby genomic

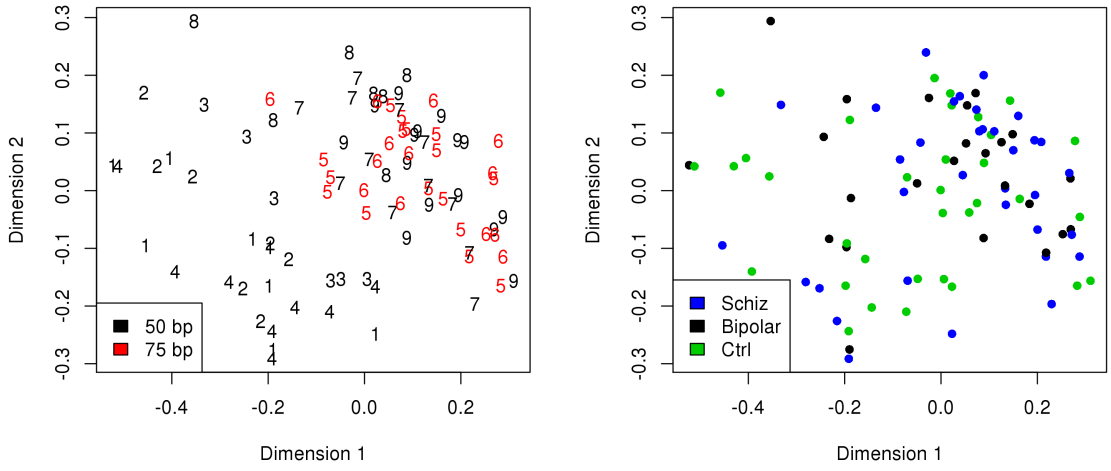

Fig. 39. Multidimensional scaling plots of genic regions of methylation samples in the Stanley data. Samples points are given as batch number in the left figure and are colored by read protocol. Samples in the right panel are colored by phenotype.

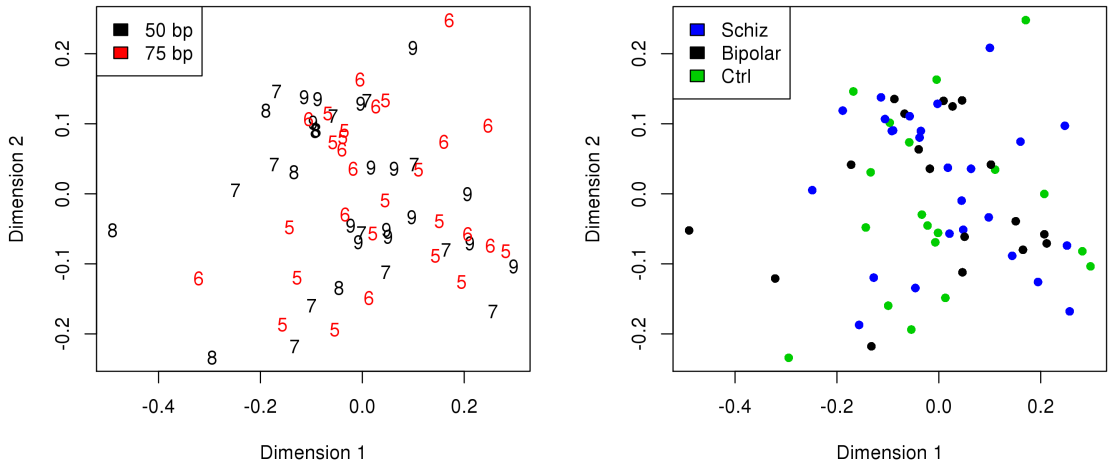

Fig. 40. Multidimensional scaling plots of genic regions of methylation samples from batches five through nine in the Stanley data. Samples points are given as batch number in the left figure and are colored by read protocol. Samples in the right panel are colored by phenotype.

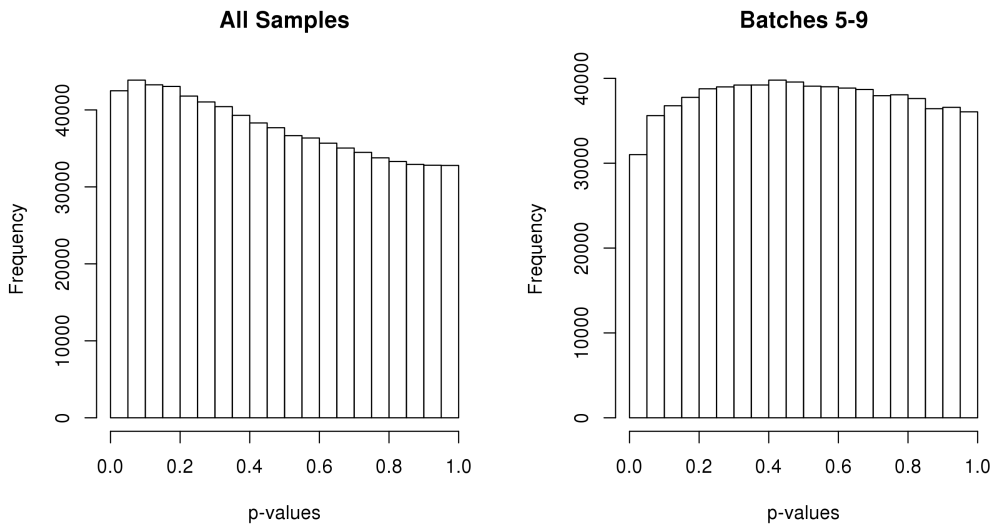

Fig. 41. Distributions of p-values for one-way ANOVA testing for significance of disease phenotype in MBD-Seq data. MBD-Seq was binned in 300 bp intervals. Intervals with mean counts ¡ 10 across all samples were excluded.

intervals.

5.1.2 Gene expression

RNA-Seq was performed on 82 samples taken from a similar, but different region of cerebral cortex: cingulate cortex. Data was processed in 6 batches of varying sizes. Reads were aligned to the hg19 reference genome and counts were aggregated by gene. RPKM was computed for each gene using the formula from Equation 1.9. Samples were sequenced at varying read depths, but samples taken from bipolar patients were systematically sequenced at lower depths as seen in Figure 42. Despite scaling for read depth using RPKM, differences due to read depth can still persist (Robinson and Oshlack 2010), so additional normalization steps are taken. Figure 43 shows an MDS plot of samples colored by disease phenotype and numbered by batch.

For the RNA-Seq data, we have processing information for each sample including

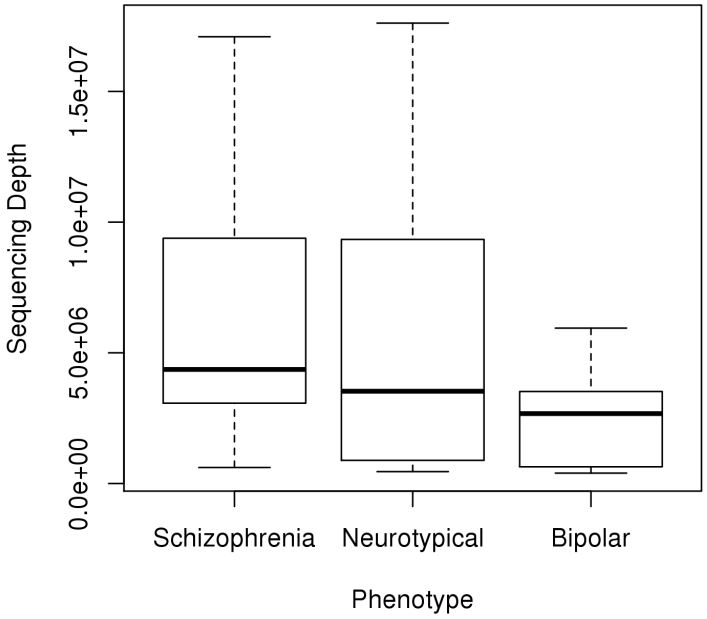

Fig. 42. Boxplots of sample read depths by disease phenotype. Samples from bipolar patients were sequenced at lower read depths

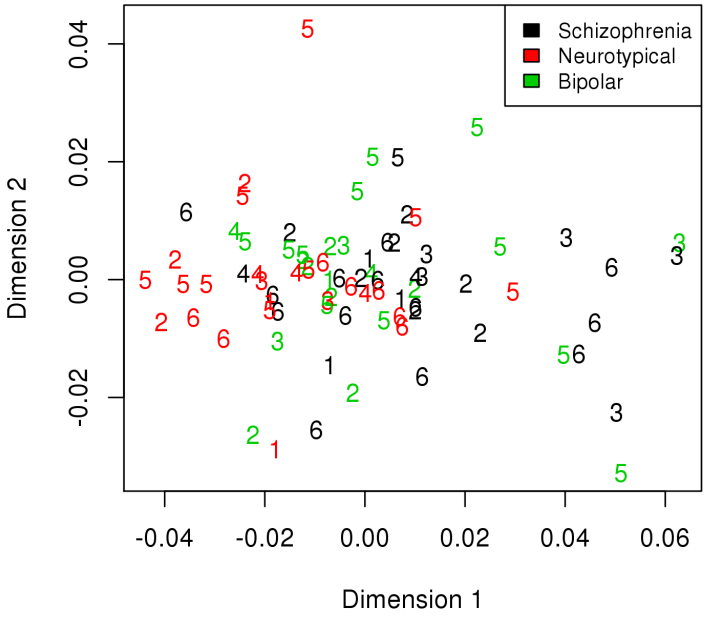

Fig. 43. Multidimensional scaling plot of RNA-Seq samples in the Stanley data. Samples are numbered by batch and colored by disease phenotype.

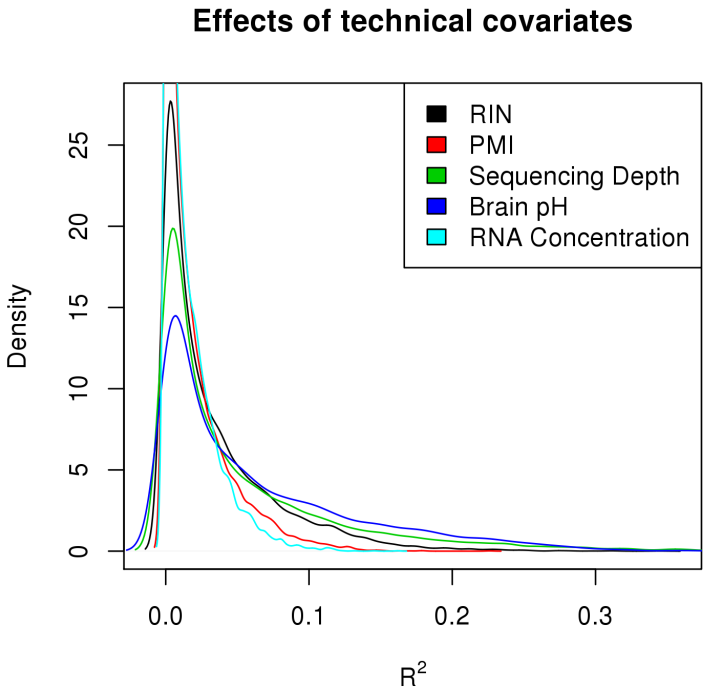

Fig. 44. Density plots of R^2 between technical covariates and $\sqrt{\text{RPKM}}$ for each gene.

RNA integrity number (RIN: Schroede et al. 2006), post-mortem interval (PMI), brain pH, RNA concentration, and sequencing depth. Figure 44 gives densities of R^2 values for each covariate with $\sqrt{\text{RPKM}}$ from each gene. We can see that several of these covariates are able to explain $\sim 10\%$ of the variability or more in $\sqrt{\text{RPKM}}$ for some genes. These technical covariates are also minimally correlated with each other, so an additive linear model was used to regress out the effects of technical covariates for each gene from $\sqrt{\text{RPKM}}$. Figure 45 gives an MDS plot for samples after technical covariates have been regressed out.

Similar to MBD-Seq, we first perform an initial exploratory analysis using only the RNA-Seq data. For each gene, a one-way ANOVA for disease phenotype is fit to $\sqrt{\text{RPKM}}$ after regressing out technical covariates. After p-values are obtained,

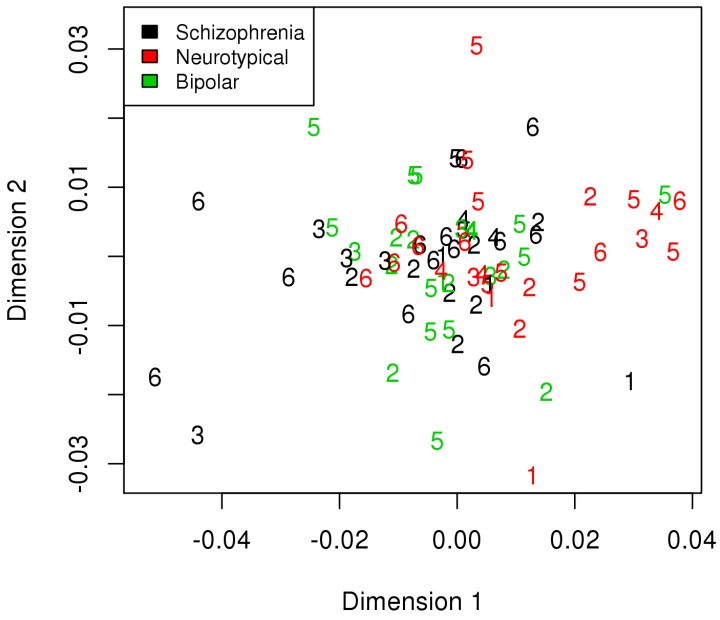

Fig. 45. Multidimensional scaling plot of RNA-Seq samples after regressing out technical covariates. Samples are numbered by batch and colored by disease phenotype.

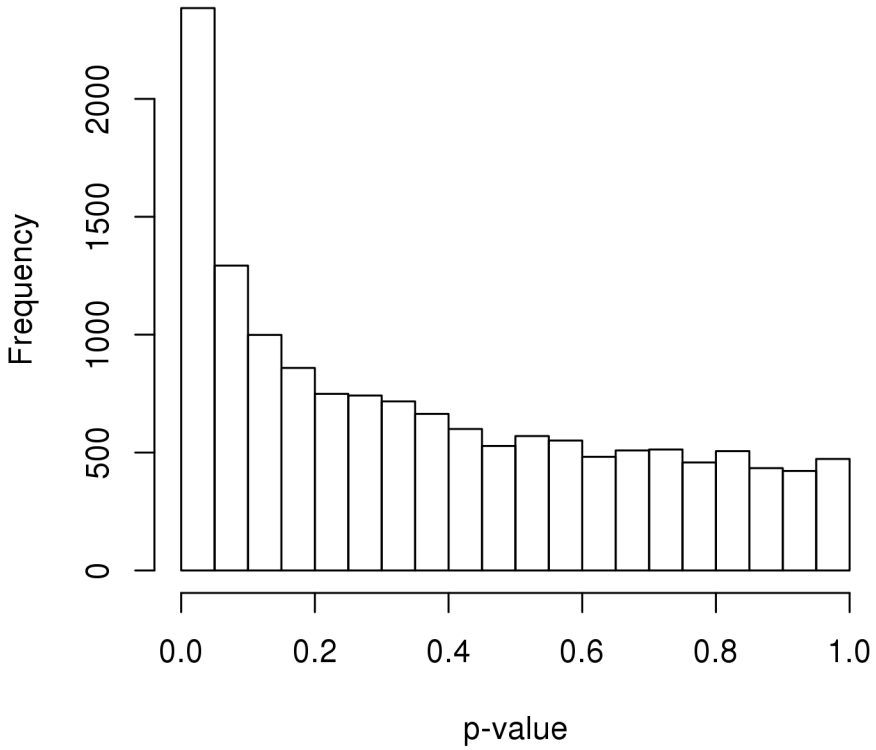

Fig. 46. Distribution of p-values from one-way ANOVAs for each gene testing for significance of disease phenotype in RNA-Seq data.

the false-discovery rate is controlled at (FDR= 0.1) using the Benjamini-Hochberg method. Figure 46 gives the resulting p-value histogram. 189 genes were significant at FDR = 0.1. Table VIII gives the top results from a gene ontology analysis using Fisher's Exact Test and the weight01 algorithm from the topGO package in R (Alexa, Rahnenfhrer, and Lengauer 2006). Differences are enriched for neurotransmitter, cell vesicle, and synaptic categories.

Table VIII. Top enriched GO categories using q-values from a one-way ANOVA for disease phenotype

GO.ID	Term	Annotated	Significant	Expected	P-value
GO:0072659	protein localization to plasma membrane	90	6	2.09	0.00019
GO:0019285	glycine betaine biosynthetic process fro...	2	2	0.05	0.00054
GO:0051932	synaptic transmission, GABAergic	23	5	0.53	0.00156
GO:0016082	synaptic vesicle priming	6	4	0.14	0.00156
GO:0032252	secretory granule localization	3	2	0.07	0.00159
GO:0010807	regulation of synaptic vesicle priming	3	2	0.07	0.00159
GO:0014047	glutamate secretion	24	4	0.56	0.0021
GO:0007214	gamma-aminobutyric acid signaling pathwa...	12	3	0.28	0.00234
GO:0007268	synaptic transmission	520	34	12.09	0.00242
GO:0016188	synaptic vesicle maturation	4	2	0.09	0.00313

5.1.3 Genotypes

Genotypes were obtained for 70 samples and imputed to 16,174,402 total SNPs. Sample files did not include rs IDs (accession numbers used to refer to specific SNPs standing for Reference SNP cluster ID). Because of this, SNPs were then mapped to imputed genotypes from the 1000 Genomes Project by genomic coordinates. Since we are specifically interested in eQTL analysis, we use a list of identified eQTLs from Gibbs et al. 2010 to subset the whole set of SNPs to perform a focused analysis and reduce computational burden and number of statistical tests.

Gibbs et al. 2010 identified roughly 20 thousand eQTLs specific to the brain in a study using 120 brains spanning ages 20 to 101 years old sampled at four distinct brain regions: prefrontal cortex, temporal cortex, cerebellum, and pons. If we subset their list of eQTLs by those specific to the two cortical regions, 6.5k unique eQTLs affecting the expression of 597 genes remain. We then select the subset of Stanley SNPs that have been identified as eQTLs by the Gibbs study. 3.7k SNPs map over from the Stanley samples that are present in the Gibbs eQTL list. However, many of

the SNPs in the Stanley samples contain a large number of missing values and SNPs with more than 10 missing values were omitted. This leaves a final set of 2.1k SNPs corresponding to 83 genes. In the next section we detail the analysis used to validate the putative eQTLs extracted using information from Gibbs et al. 2010.

5.2 Detecting quantitative trait loci

55 samples from the Stanley data had both RNA-Seq and genotype data. In order to test whether eQTLs from Gibbs et al. 2010 had an effect on gene expression in the Stanley samples, a simple linear model was fit for each gene i in sample j with disease phenotype k using normalized $y_i = \sqrt{\text{RPKM}_{ij}}$ as the dependent variable, and genotype and disease phenotype as independent variables. Genotype was coded as $x_{ij} = \{0, 1, \text{or } 2\}$ corresponding to the number of minor alleles and was treated as ordinal. Separate models were fit for each eQTL since some models may become over-parameterized due to some genes having a large number of eQTLs. Since disease phenotype will also likely affect gene expression, it was included in the model as a categorical variable α_{ik}. Equation 5.1 gives the linear model used for each SNP.

$$y_{ijk} = \alpha_{ik} + x_{ij}\beta_i + \epsilon_{ij} \tag{5.1}$$

Figure 47 gives the resulting p-value histograms from Wald tests for the significance of β and α from Equation 5.1. While completely redundant SNPs were filtered out, P-value distributions may be non-uniform due to SNPs being in linkage disequilibrium with each other, and therefore correlated. No eQTLs from Gibbs et al. 2010 were significant at FDR = 0.1 when using the Benjamini-Hochberg correction (Benjamini and Hochberg 1995).

While some of the lack of significance may be attributed to a smaller sample

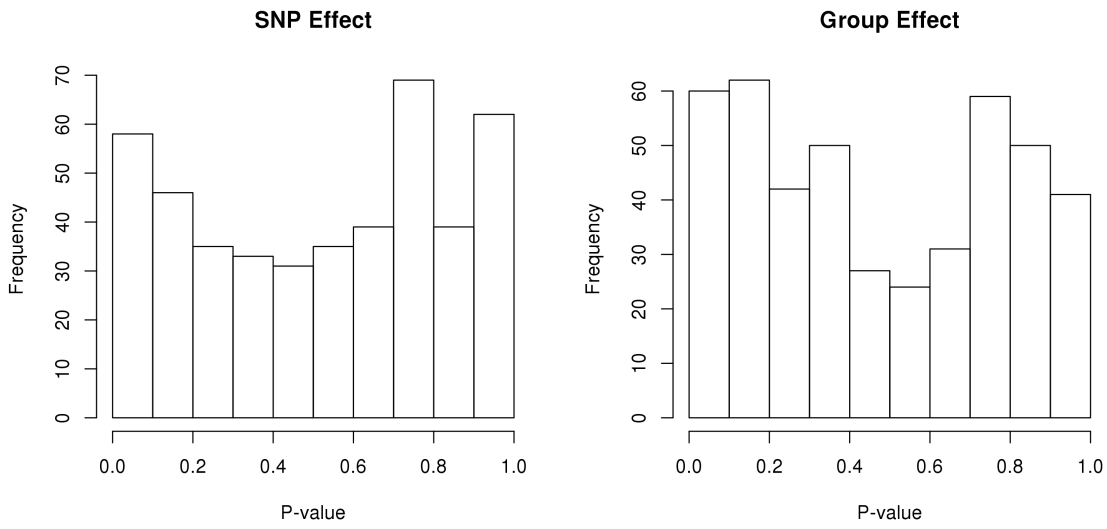

Fig. 47. P-value histograms from Wald tests for eQTL effect and disease phenotype from Equation 5.1.

size and linkage disequilibrium of non-significant SNPs, the majority of SNPs *should* be significant since they come from a preselected set of eQTLs. Lack of significance cannot be attributed to the larger effect of disease phenotype since it was included in the model and has a similar level of significance.

5.3 Integrating DNA methylation and gene expression

5.3.1 Principal component regression

58 samples had both RNA-Seq and MBD-Seq data available. As in the case of the BrainSpan data, there is an issue in the Stanley MBD-Seq data that for each gene, the number of samples n is generally smaller than the number of 300 bp intervals p. For example, the GABA-A Receptor Subunit Alpha-5 (GABRA5) gene which is roughly 80 kb has 267 intervals when binned in 300 bp intervals. Many of these intervals may not contain methylation sites or have low counts, so the final number of intervals

included for the analysis ends up being 92, but this is still well above the sample size of 58. Many of these nearby intervals should be correlated, so a method like principal component analysis should be an effective tool for dimension reduction. We can then employ a similar approach of using principal component analysis on a gene-by-gene basis as a tool for dimension reduction in the MBD-Seq data. The first $k = 1, ..., 3$ PC scores x_{ijk} for each gene i and sample j are then used as independent variables in the linear model given in Equation 5.2 where y_{ij} is covariate-adjusted $\sqrt{\text{RPKM}}$.

$$y_{ij} = \sum_{k=1}^{3} x_{ijk}\beta_{ik} + \epsilon_{ij} \tag{5.2}$$

5.3.2 Results

5.3.2.1 Analysis on all samples

First, an analysis was performed using all 58 samples with paired MBD-Seq and RNA-Seq data. Figure 48 gives the resulting p-value histograms and adjusted R^2 distributions from two separate models. The left panel gives p-values from the F-statistic constructed from the linear model in Equation 5.2 testing the full model including all three principal component scores from methylation versus the null model. The middle panel gives the p-value distribution from a one-way ANOVA testing for group effect for gene expression for each gene which is identical to Figure 46. The right panel gives the distribution of adjusted R^2 values from the two models since the ANOVA model uses two degrees of freedom and the methylation linear model uses three. We have already seen that disease phenotype is significantly associated with differences in gene expression for many genes, but it appears that methylation is generally not predictive of expression. This may be in part due to poorer data quality in earlier batches which may be obscuring results. In the next section we perform an

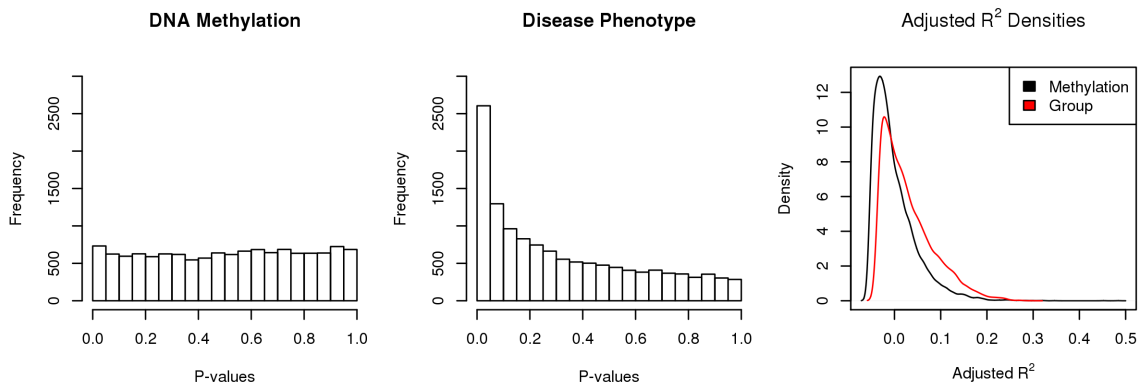

Fig. 48. Results of integrative analysis of DNA methylation and gene expression in the Stanley data. The left panel gives the resulting p-value distribution for regressing gene expression against the first 3 principal components of DNA methylation. The middle panel gives the p-value distribution from a one-way ANOVA for gene expression as a function of disease phenotype. The right panel gives densities of adjusted R^2 from the two models

identical reanalysis after subsetting methylation samples using only the later batches 5 through 9.

5.3.2.2 Reanalysis omitting earlier batches

Since some MBD-Seq samples in earlier batches may be of questionable quality, we perform a focused reanalysis using only the later batches 5 through 9. After subsetting by later batches, 63 MBD-Seq samples remain. After matching these up to corresponding RNA-Seq samples, there are 37 samples left with paired data. An identical analysis to the one performed in the previous section was then performed using this subset of samples. Figure 49 gives the same results figure giving p-value histograms and adjusted R^2 densities. Unfortunately, using only later batches does not seem to remedy the problem, and the decreased power from a smaller sample size seems to remove much of the significance due to differences in disease phenotype.

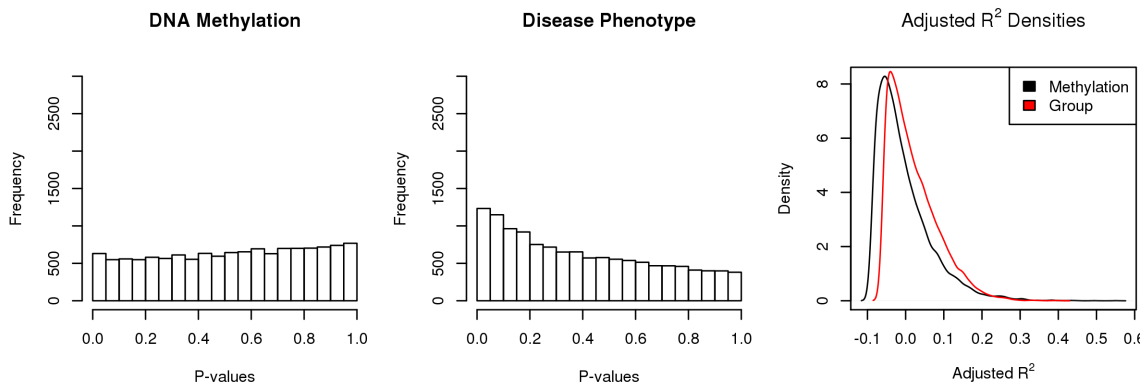

Fig. 49. Results of integrative analysis of DNA methylation and gene expression in the Stanley data using samples from higher quality batches. The left panel gives the resulting p-value distribution for regressing gene expression against the first 3 principal components of DNA methylation. The middle panel gives the p-value distribution from a one-way ANOVA for gene expression as a function of disease phenotype. The right panel gives densities of adjusted R^2 from the two models

5.4 Summary

In this section we performed an integrative analysis of samples obtained from the Stanley Medical Research Institute brain tissue repository. Initial exploratory analysis and quality control of MBD-Seq samples indicated that some samples may be of questionable quality. After quality control and normalization, we observed no significant changes in MBD-Seq across disease phenotype, but observed 189 genes significantly associated with disease phenotype in the RNA-Seq data that were enriched for neurotransmitter, synapse, and synaptic vesicle Gene Ontology categories. When integrating the RNA-Seq data with genotypes, we were unable to obtain similar results to those of Gibbs et al. 2010 who discovered roughly 6.5k eQTLs in cerebral cortex. When integrating gene expression and DNA methylation using principal component regression, we found no statistically significant relationships.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

In this dissertation we have introduced methods for normalization and integrative analysis of multiple genomic data sets. In the process, we have introduced a novel normalization method "Flexible local regression on empirically selected controls" (fresco) in Chapter 2 that uses a local regression surface to model and adjust for technical covariates in microarray signal intensities. By empirical control probes, fresco is robust to global shifts in DNA methylation profiles that can occur due to aberrant methylation or shifting abundances in cell type admixtures. We were able to demonstrate this robustness on several data sets using composite F-statistics to characterize causes for increase in apparent significance after normalization. Using several other performance metrics we showed that our method performed favorably when compared with other current methods.

In Chapter 3, we proposed a gene-centric suite of methods for the integrative analysis of genomic and epigenetic data with a specific focus on DNA methylation and alternative splicing. We introduced a likelihood ratio test based on the covariance matrices of principal component scores of DNA methylation and alternative splicing. Through simulation studies we showed that for modest sample sizes our method is not particularly sensitive to detecting alternative splicing of single cassette exons, but can effectively detect alternative promoter usage affecting multiple exons. After performing the likelihood ratio test, we proposed regressing canonical scores against covariates of interest using linear mixed-effects linear model and plotting canonical

communalities on a gene model to interpret results. Lastly, we proposed a permutation testing method to systematically test for co-localization of associations between DNA methylation β-values and splicing index in the gene.

In Chapter 4, we apply the methods introduced in Chapter 3 to a set of developmental brain samples obtained from the BrainSpan consortium. We estimated relative proportions of neurons and showed that relative neuron abundance decreases over age. We performed exploratory analysis of DNA methylation, alternative splicing, and gene expression and found samples to cluster most strongly by individual, with the exception of cerebellum which was distinct. We developed a method to adapt the likelihood ratio test in Chapter 3 to the situation of clustered data. Despite having little power with a small sample size, genes that had a significant association between alternative splicing and DNA methylation over brain development were primarily involved in axon guidance. We investigated the mechanisms of these relationships in several example genes.

In Chapter 5, we performed a second integrative analysis on a set of brain samples from the Stanley Medical Research Institute containing schizophrenic, bipolar, and neurotypical brain samples. We performed an exploratory analysis of MBD-Seq methylation data and found batch effects to be the main factor influencing clustering. Integrating DNA methylation and RNA-Seq measures of gene expression using principal component regression yielded no signficant genes when controlling the false discovery rate at FDR = 0.1 using the Benjamini-Hochberg method. A second integrative analysis was performed integrating genotypes and gene expression using an ANCOVA model on a subset of SNPs that had been identified as eQTLs in a data set from Gibbs et al. 2010.

6.2 Future work

In Chapter 2 we considered 3 different performance metrics for assessing the effectiveness of normalization methods: reduction in batch effects, increase in apparent significance, and change in composite F statistics post normalization. While these methods provide some insight into the performance of these methods, none of them *directly* assess the issue of statistical power. Due to the complexity of the microarray and variety of current normalization methods it is difficult to simulate a realistic scenario where true methylation states are known but technical artifacts and noise are realistically simulated, especially for out-of-band probes used by funnorm and noob. Furthermore, since reproducible artifacts can occur as a result of normalization, reproducibility of findings in independent data sets after normalization is not a sufficient metric. It would be desirable to have some sort of "spike-in" data set of 450k arrays, where the truth is known and more dependable comparisons can be made. Lastly, since our method fits a multivariate local regression surface, it can often be slower than other methods. Implementation of a parallel framework for model fitting, or perhaps a different surface fitting algorithm may ameliorate this issue. We plan to release an implementation of the methods in Chapter 2 as R package fresco.

In Chapter 3 we introduced a suite of methods for the integrative analysis of genomic and epigenetic data. In order to conduct a likelihood ratio test on a large number of covariance parameters in the case of $n < p$, principal component analysis was first used to reduce the number of parameters before performing canonical correlation analysis. However, sparse L_1 penalized methods exist for canonical correlation analysis that may also aid in the interpretation of canonical loadings. The use of sparse L_1 penalized methods generally involves cross-validation in order to select appropriate tuning parameters. In the case of CCA, two tuning parameters must

be selected, one for each data set. Selecting two tuning parameters for each gene becomes a very computationally intensive task and may not scale well to large data sets. After canonical correlation, we proposed a permutation test for co-localization of associations between two data sets to specific locations on the gene. This test takes into account pairwise distances between the two data sets, but does not account for the fact that all loci lie on a continuous line. A method that accounts for this in some way by perhaps using a smoothing method may prove to be more powerful, especially in the case of sequencing data where coverage and resolution may be higher than for the 450k array. We plan to release an implementation of the methods in Chapter 3 as R package gdi.

In Chapter 4, we implemented the methods from Chapter 3 to a set of developmental brain samples obtained from the BrainSpan consortium. We limited the analysis to four brain regions from prefrontal cortex and focused on developmental changes rather than differences in brain regions. The motivation for analyzing this subset was in part due to several samples lacking paired data from both DNA methylation and the exon array. In the future, more samples will become available including prenatal samples that will allow for an analysis with both a larger sample size and a spanning a wider range of ages.

Current results are also restricted by limited means for estimating cell type abundances in brain samples. Currently, we are only able to estimate neuron abundance with any degree of reliability, but perhaps data sets in the future will provide isolated methylation profiles for the different glial types and allow for estimation of proportions of glial sub-populations and perhaps different neuronal subtypes. For many genes, the Illumina 450k array does not provide adequate coverage in gene bodies to detect potential relationships between alternative splicing, alternative promoter usage, and DNA methylation. Future studies using sequencing technologies will add

improved coverage and resolution and hopefully illuminate many relationships that have potentially gone unobserved.

In Chapter 5, we performed an integrative analysis of brain samples obtained from the Stanley Medical Research Institute. Preprocessing of MBD-Seq data was crude and reads we binned in 300 bp intervals that were agnostic to coding DNA sequences and genomic locations of regulatory sites. Reads that mapped to multiple locations were evenly divided among the multiple locations. A more sophisticated preprocessing method may improve downstream data quality. While the set of eQTLs from Gibbs et al. 2010 did not seem to carry over to the Stanley samples, perhaps a more thorough eQTL analysis or another eQTL list might provide more interesting results. We also only used gene-level summaries of RNA-Seq data. Perhaps aggregating reads by at the exon level would allow for the discovery of changes in alternative splicing across disease phenotypes.

REFERENCES

Abatangelo, Luca et al. (2009). "Comparative study of gene set enrichment methods". In: *BMC Bioinformatics* 10.275.

Adler, Daniel et al. (2014). *ff: memory-efficient storage of large data on disk and fast access functions*. R package version 2.2-13. URL: `http://CRAN.R-project.org/package=ff`.

Alexa, Adrian, Jrg Rahnenfhrer, and Thomas Lengauer (2006). "Improved scoring of functional groups from gene expression data by decorrelating GO graph structure". In: *Bioinformatics* 22.13, pp. 1600–1607.

Analytics, Revolution and Steve Weston (2014). *foreach: Foreach looping construct for R*. R package version 1.4.2. URL: `http://CRAN.R-project.org/package=foreach`.

Anders, Simon and Wolfgang Huber (2010). "Differential expression analysis for sequence count data". In: *Genome Biology* 11.10.

Anderson, E. et al. (1999). *LAPACK Users' Guide, Third Edition*. Society for Industrial and Applied Mathematics.

Aryee, Martin J. et al. (2014). "Minfi: a flexible and comprehensive Bioconductor package for the analysis of Infinium DNA methylation microarrays". In: *Bioinformatics* 30.10.

Bancroft, Tim, Chuanlong Du, and Dan Nettleton (2013). "Estimation of False Discovery Rate Using Sequential Permutation p-Values". In: *Biometrics* 69.1, pp. 1–7.

Bartlett, M.S. (1939). "A Note on Tests of Significance in Multivariate Analysis". In: *Proceedings of the Cambridge Philosophical Society* 35.2.

Bates, Timothy C. et al. (2011). "Genetic variance in a component of the language acquisition device: ROBO1 polymorphisms associated with phonological buffer deficits". In: *Behavioral Genetics* 41.1, pp. 50–57.

Baylin, Stephen B. et al. (2001). "Aberrant patterns of DNA methylation, chromatin formation and gene expression in cancer". In: *Human Molecular Genetics* 10.7, pp. 687–692.

Benjamini, Yoav and Yosef Hochberg (1995). "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing". In: *Journal of the Royal Statistical Society. Series BB* 57.1, pp. 289–300.

Bibikova, Marina et al. (2011). "High density DNA methylation array with single CpG site resolution". In: *Genomics* 98.4, pp. 288–295.

Bing Fan, Marina Bibikova andJian (2010). "Genome-wide DNA methylation profiling". In: *Wiley Interdisciplinary Reviews: Systems Biology and Medicine* 2.2, pp. 210–223.

Bolstad, Benjamin M. et al. (2003). "A comparison of normalization methods for high density oligonucleotide array data based on variance and bias". In: *Bioinformatics* 19.2, pp. 185–193.

Carvalho, Benilton S and Rafael A. Irizarry (2010a). "A Framework for Oligonucleotide Microarray Preprocessing". In: *Bioinformatics*. ISSN: 1367-4803. DOI: http://dx.doi.org/10.1093/bioinformatics/btq431.

Carvalho, Benilton S. and Rafael A. Irizarry (2010b). "A framework for oligonucleotide microarray preprocessing". In: *Bioinformatics* 26.19, pp. 2363–2367.

Chakrabarti, Kausik et al. (2005). "Critical Role for Kalirin in Nerve Growth Factor Signaling through TrkA". In: *Molecular and Cellular Biology* 25.12, pp. 5106–5118.

Chen, Chao et al. (2011). "Removing Batch Effects in Analysis of Expression Microarray Data: An Evaluation of Six Batch Adjustment Methods". In: *PLoS ONE* 6.2.

Chen, Wenan et al. (2013a). "MethylPCA: a toolkit to control for confounders in methylome-wide association studies". In: *BMC Bioinformatics* 14.74.

Chen, Yi an et al. (2013b). "Discovery of cross-reactive probes and polymorphic CpGs in the Illumina Infinium HumanMethylation450 microarray". In: *Epigenetics* 8.2, pp. 203–209.

Chi, Yueh-Yun and Keith E. Muller (2013). "Two-Step Hypothesis Testing When the Number of Variables Exceeds the Sample Size". In: *Communications in Statistics-Simulation and Computation* 42.5, pp. 1113–1125.

Cingolani, Pablo et al. (2013). "Intronic Non-CG DNA hydroxymethylation and alternative mRNA splicing in honey bees". In: *BMC Genomics* 14.666.

Cleveland, W. S., E. Grosse, and W. M. Shyu (1992). "Statistical Models in S". In: ed. by J.M. Chambers and T.J. Hastie. Wadsworth & Brooks/Cole. Chap. Local regression models.

Cline, Melissa S. et al. (2005). "ANOSVA: a statistical method for detecting splice variation from expression data". In: *Bioinformatics* 21.suppl 1.

Collins, Christine E. et al. (2010). "Neuron densities vary across and within cortical areas in primates". In: *PNAS* 107.36.

Colomer, V. et al. (1997). "Huntingtin-Associated Protein 1 (HAP1) Binds to a Trio-Like Polypeptide, with a rac1 Guanine Nucleotide Exchange Factor Domain". In: *Human Molecular Genetics* 6.9, pp. 1519–1525.

Dedeurwaerder, Sarah et al. (2011). "Evaluation of the Infinium Methylation 450K technology". In: *Epigenomics* 3.6, pp. 771–784.

Dongen, Jenny van and Dorret I. Boomsma (2013). "The Evolutionary Paradox and the Missing Heritability of Schizophrenia". In: *Am J Med Genet Part B* 162.2, pp. 122–136.

Du, Pan et al. (2010). "Comparison of Beta-value and M-value methods for quantifying methylation levels by microarray analysis". In: *BMC Bioinformatics* 11.587.

Eisenberg, Eli and Erez Y. Levanon (2013). "Human housekeeping genes, revisited". In: *Trends in Genetics* 29.10, pp. 569–574.

Fields, R. Douglas (2009). *The Other Brain*. 1st ed. Simon & Schuster.

Fisher, R.A. (1973). *Statistical Methods for Research Workers*. Hafner.

Fortin, Jean-Philippe et al. (2014). "Functional normalization of 450k methylation array data improves replication in large cancer studies". In: *bioArχiv*.

Gagnon-Bartsch, Johann A. and Terence P. Speed (2011). "Using control genes to correct for unwanted variation in microarray data". In: *Biostatistics* 13.3, pp. 539–552.

Gaidatzis, Dimos et al. (2009). "Overestimation of alternative splicing caused by variable probe characteristics in exon arrays". In: *Nucleic Acides Res* 37.16, e107.

GeneChip Exon Array Design (2005). URL: `http://media.affymetrix.com/support/technical/technotes/exon_array_design_technote.pdf`.

Gibbs, J. Raphael et al. (2010). "Abundant Quantitative Trait Loci Exist for DNA Methylation and Gene Expression in Human Brain". In: *PLoS Genetics* 6.5.

Guintivano, Jerry, Martin J Aryee, and Zachary A Kaminsky (2013). "A cell epigenotype specific model for the correction of brain cellular heterogeneity bias and its application to age, brain region and major depression". In: *Epigenetics* 8.3, pp. 290–302.

Hashimoto, Ryota et al. (2005). "A missense polymorphism (H204R) of a Rho GTPase-activating protein, the chimerin 2 gene, is associated with schizophrenia in men". In: *Schizophrenia Research* 73.2-3, pp. 383–385.

Hotelling, Harold (1933). "Analysis of a complex of statistical variables into principal components". In: *Journal of Educational Psychology* 24.6.

— (1936). "Relations Between Two Sets of Variates". In: *Biometrika* 28.3/4.

Houseman, Eugene Andres et al. (2012). "DNA methylation arrays as surrogate measures of cell mixture distribution". In: *BMC Bioinformatics* 13.86.

Irizarry, Rafael A. et al. (2003). "Exploration, normalization, and summaries of high density oligonucleotide array probe level data". In: *Biostatistics* 4.2, pp. 249–264.

Jaffe, Andrew E. and Rafael A. Irizarry (2014). "Accounting for cellular heterogeneity is critical in epigenome-wide association studies". In: *Genome Biology* 15.2.

Jaffe, Andrew E et al. (2012). "Bump hunting to identify differentially methylated regions in epigenetic epidemiology studies". In: *International Journal of Epidemiology* 41.1, pp. 200–209.

Jirtle, Randy (2012). *geneimprint*. URL: `http://www.geneimprint.com/` (visited on 09/24/2014).

Johnson, Richard A. and Dean W. Wichern (2007). *Applied Multivariate Statistical Analysis: Sixth*. Upper Saddle River.

Johnson, W. Evan and Cheng Li (2006). "Adjusting batch effects in microarray expression data using empirical Bayes methods". In: *Biostatistics* 8.1, pp. 118–127.

Kang, Hyo Jung et al. (2011). "Spatiotemporal transcriptome of the human brain". In: *Nature* 478.7370, pp. 483–489.

Kapur, Karen et al. (2007). "Exon arrays provide accurate assessments of gene expression". In: *Genome Biology* 8.5.

Kapur, Karen et al. (2008). "Cross-hybridization modeling on Affymetrix exon arrays". In: *Bioinformatics* 24.24.

Kozlenkov, Alexey et al. (2013). "Differences in DNA methylation between human neuronal and glial cells are concentrated in enhancers and non-CpG sites". In: *Nucleic Acids Res.* 42.1.

Krebs, Jocelyn E., Elliot S. Goldstein, and Stephen T. Kilpatrick (2013). *Lewin's Essential Genes*. 3rd ed. Jones & Bartlett Learning.

Kshirsagar, A.M. (1972). *Multivariate Analysis*. Marcel Dekker, Inc.

Langmead, Ben et al. (2009). "Ultrafast and memory-efficient alignment of short DNA sequences to the human genome". In: *Genome Biology* 10.3.

Lawley, D.N. (1959). "Tests of Significance in Canonical Analysis". In: *Biometrika* 46.1/2, pp. 59–66.

Lee, S Hong et al. (2012). "Estimating the proportion of variation in susceptibility to schizophrenia captured by common SNPs". In: *Nature Genetics* 44.3, pp. 247–250.

Leek, Jeffrey T and John D Storey (2007). "Capturing Heterogeneity in Gene Expression Studies by Surrogate Variable Analysis". In: *PLoS Genetics* 3.9.

Lister, Ryan et al. (2013). "Global Epigenomic Reconfiguration During Mammalian Brain Development". In: *Science* 341.6146.

Lu, Henry Horng-Shing, Bernhard Schölkopf, and Hongyu Zao, eds. (2011). *Handbook of Statistical Bioinformatics*. Springer.

Maksimovic, Jovana, Lavinia Gordon, and Alicia Oshlack (2012). "SWAN: Subset-quantile Within Array Normalization for Illumina Infinium HumanMethylation450 BeadChips". In: *Genome Biology* 13.6.

Maunakea, Alika K et al. (2013). "Intragenic DNA methylation modulates alternative splicing by recruiting MeCP2 to promote exon recognition". In: *Cell Research* 23, pp. 1256–1269.

McGuffin, Peter et al. (2003). "The Heritability of Bipolar Affective Disorder and the Genetic Relationship to Unipolar Depression". In: *Archives of General Psychiatry* 60.5, pp. 497–502.

Mortazavi, Ali et al. (2008). "Mapping and quantifying mammalian transcriptomes by RNA-Seq". In: *Nature Methods* 5, pp. 621–628.

Nadon, Robert and Jennifer Shoemaker (2002). "Statistical issues with microarrays: processing and analysis". In: *TRENDS in Genetics* 18.5.

Plomin, Robert et al. (1994). "DNA Markers Associated with High Versus Low IQ: The IQ Quantitative Trait Loci (QTL) Project". In: *Behavior Genetics* 24.2.

Purdom, E. et al. (2008). "FIRMA: a method for detection of alternative splicing from exon array data". In: *Bioinformatics* 24.15.

Reinius, Lovisa E. et al. (2012). "Differential DNA Methylation in Purified Human Blood Cells: Implications for Cell Lineage and Studies on Disease Susceptibility". In: *PLoS ONE* 7.7.

Robinson, Mark D., Davis J. McCarthy, and Gordon K. Smyth (2010). "edgeR: a Bioconductor package for differential expression analysis of digital gene expression data". In: *Bioinformatics* 26.1.

Robinson, Mark D and Alicia Oshlack (2010). "A scaling normalization method for differential expression analysis of RNA-seq data". In: *Genome Biology* 11.3.

Rowe, Suzanne J. et al. (2013). "Complex Variation in Measures of General Intelligence and Cognitive Change". In: *PLoS ONE* 9.3.

Schielzeth, Holger and Wolfgang Forstmeier (2008). "Conclusions beyond support: overconfident estimates in mixed models". In: *Behavioral Ecology* 20.2, pp. 416–420.

Schroede, Andreas et al. (2006). "The RIN: an RNA integrity number for assigning integrity values to RNA measurements". In: *BMC Molecular Biology* 7.3.

Serre, David, Byron H. Lee, and Angela H. Ting (2010). "MBD-isolated Genome Sequencing provides a high-throughput and comprehensive survey of DNA methylation in the human genome". In: *Nucleic Acids Research* 38.2, pp. 391–399.

Shukla, Sanjeev et al. (2011). "CTCF-promoted RNA polymerase II pausing links DNA methylation to splicing". In: *Nature* 479, pp. 74–79.

Storey, John D. (2003). "The Positive False Discovery Rate: A Bayesian Interpretation and the q-value". In: *The Annals of Statistics* 31.6.

Subramanian, Aravind et al. (2005). "Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles". In: *PNAS* 102.43.

Sullivan, Patrick F., Kenneth S. Kendler, and Michael C. Neale (2003). "Schizophrenia as a Complex Trait: Evidence From a Meta-analysis of Twin Studies". In: *Archives of General Psychiatry* 60.12, pp. 1187–1192.

Teschendorff, Andrew E. et al. (2012). "A beta-mixture quantile normalization method for correcting probe design bias in Illumina Infinium 450 k DNA methylation data". In: *Bioinformatics* 29.2, pp. 189–196.

Timothy J. Triche, Jr et al. (2013). "Low-level processing of Illumina Infinium DNA Methylation BeadArrays". In: *Nucleic Acids Research* 41.7.

Touleimat, Nizar and Jörg Tost (2012). "Complete pipeline for Infinium Human Methylation 450K BeadChip data processing using subset quantile normalization for accurate DNA methylation estimation". In: *Epigenomics* 4.3.

Warden, Charles D. et al. (2013). "COHCAP: an integrative genomic pipeline for single-nucleotide resolution DNA methylation analysis". In: *Nucleic Acids Research* 41.11.

Xie, Zhong et al. (2007). "Kalirin-7 controls activity-dependent structural and functional plasticity of dendritic spines". In: *Neuron* 56.4, pp. 640–656.

Xing, Yi, Karen Kapur, and Wing Hung Wong (2006). "Probe Selection and Expression Index Computation of Affymetrix Exon Arrays". In: *PLoS ONE* 1.1.

Xing, Yi et al. (2008). "MADS: A new and improved method for analysis of differential alternative splicing by exon-tiling microarrays". In: *RNA* 14.8, pp. 1470–1479.

Zhang, Weiwei et al. (2013). "Predicting genome-wide DNA methylation using methylation marks, genomic position, and DNA regulatory elements". In: *arχiv*.

Appendix A

ABBREVIATIONS

AMY	Amygdala
ANOSVA	Analysis of splice variation
ANOVA	Analysis of variance
BMIQ	Beta mixture quantile normalization
bp	base pairs
CBL/CBC	Cerebellum
CCA	Canonical correlation analysis
CHN2	Chimerin 2
CNS	Central nervous system
COSIE	Corrected Splicing Indices for Exon Arrays
DFC	Dorsolateral prefrontal cortex
DR	Detection rate
ECDF	Empirical cumulative density function
eQTL	Expression quantitative trait loci
FACS	Fluorescence activated cell sorting
FDR	False-discovery rate
FIRMA	Finding isoforms using robust multichip analysis
FRESCO	Flexible local regression on empirically selected control probes
gdi	Genomic data integration
GEO	Gene Expression Omnibus
GO	Gene Ontology

GSEA	Gene Set Enrichment Analysis
GWAS	Genome-wide association study
HCC	Hepatocellular Carcinoma
HIP	Hippocampus
i.i.d.	independently and identically distributed
kb	kilobase
KLRN	Kalirin
LRT	Likelihood ratio test
MAD	Median absolute deviation
MADS	Microarray analysis of differential splicing
MBD	Methyl-CpG-binding domain
MDS	Multidimensional scaling
MFC	Medial prefrontal cortex
mQTL	Methylation quantiative trait loci
NCTX	Neocortex
NGS	Next generation sequencing
Noob	Normal-exponential using out-of-band probes
OFC	Orbitofrontal cortex
PCA	Principal component analysis
PLS	Partial least squares
PMI	Post-mortem interval
QTL	Quantitative trait loci
RC	Redundancy coefficient
RIN	RNA integrity number
RMA	Robust multi-chip average

ROBO1	Roundabout homolog 1
RPKM	Reads per kilobase per million
RPM	Reads per million
rsID	Reference SNP cluster ID
SMRI	Stanley Medical Research Institute
SNP	Single nucleotide polymorphism
SQN	Subset quantile normalization
STR	Striatum
SVD	Singular value decomposition
SWAN	Subset-quantile within array normalization
TCGA	The Cancer Genome Atlas
TF	Transcription factor
THM	Thalamus
TPR	True positive rate
UTR	Untranslated region
VFC	Ventral prefrontal cortex

Appendix B

CODE FROM R PACKAGE FRESCO

```
#'
#' @param object \code{MethylSet} object
#' @param useControls Should empirical controls be used to align and fit loess
    surfaces?
#' @param loessSpan Supply span for fitting loess surface
#' @param fitLoess Should loess curve be fitted after initial alignment and
    scaling?
#' @param sdThreshold Threshold to filter empirical controls by standard deviation
#'
#' @export preprocessFresco


preprocessFresco <-function(object, useControls = TRUE, loessSpan = .15,
                            fitLoess = TRUE, sdThreshold = .15, verbose = TRUE){

  if (!is(object, "MethylSet")) stop("'object' needs to be a 'MethylSet'")
  if (loessSpan > 1 | loessSpan < 0) stop("loessSpan must be between zero and one"
      )

  data(frescoData)

  object <- fixMethOutliers(object)

  # create object for methylated and unmethylated channels
      -------------------------
  signals <- array(dim = c(dim(object), 2))
  signals[, , 1] <- getUnmeth(object)
  signals[, , 2] <- getMeth(object)
  frescoData <- frescoData[match(rownames(object), rownames(frescoData)), ]
  GC <- frescoData$targetGC

  # get set of empirical controls
      ------------------------------------------------
  if (useControls){
    probeSD <- rowSds(getBeta(object))
    controls <- which(!is.na(frescoData$eControls) & probeSD < sdThreshold)
    if (verbose) cat(length(controls), 'empirical control probes selected\n')
  }

  # divide probes and controls up by probe type
      ------------------------------------
  whichSetII <- which(frescoData$probeType == 'II')
  whichSetI <- which(frescoData$probeType == 'I')

  if (useControls){
    whichControlsII <- intersect(whichSetII, controls)
    whichControlsI <- intersect(whichSetI, controls)
  } else {
    whichControlsII <- whichSetII
    whichControlsI <- whichSetI
  }

  # find lower peaks
      ------------------------------------------------------------------
```

```
if (verbose) cat('Aligning signal intensities \n')
typeIpeaks ← apply(signals[whichControlsI, , ], c(2, 3), getLowerPeak)
typeIIpeaks ← apply(signals[whichControlsII, , ], c(2, 3), getLowerPeak)
typeIpeakMeans ← colMeans(typeIpeaks)
typeIIpeakMeans ← colMeans(typeIIpeaks)

# line up samples by their lower peaks
    ------------------------------------------
signals[whichSetI, , 1] ← sweep(signals[whichSetI, , 1], 2, typeIpeaks[, 1], '-'
    )
signals[whichSetI, , 2] ← sweep(signals[whichSetI, , 2], 2, typeIpeaks[, 2], '-'
    )
signals[whichSetII, , 1] ← sweep(signals[whichSetII, , 1], 2, typeIIpeaks[, 1],
    '-')
signals[whichSetII, , 2] ← sweep(signals[whichSetII, , 2], 2, typeIIpeaks[, 2],
    '-')

# scale signals to minimize deviance from control averages
    ----------------------
if (verbose) cat('Applying linear scaling factor \n')
typeIcontrolAvg ← apply(signals[whichControlsI, , ], c(1, 3), mean)
typeIIcontrolAvg ← apply(signals[whichControlsII, , ], c(1, 3), mean)

coefsI1 ← lm(signals[whichControlsI, , 1] ~ typeIcontrolAvg[, 1] + 0)$coef
coefsI2 ← lm(signals[whichControlsI, , 2] ~ typeIcontrolAvg[, 2] + 0)$coef
coefsII1 ← lm(signals[whichControlsII, , 1] ~ typeIIcontrolAvg[, 1] + 0)$coef
coefsII2 ← lm(signals[whichControlsII, , 2] ~ typeIIcontrolAvg[, 2] + 0)$coef

scaledSignals ← array(dim = dim(signals))
scaledSignals[whichSetI, , 1] ← sweep(signals[whichSetI, , 1], 2, coefsI1, '/')
    + typeIpeakMeans[1]
scaledSignals[whichSetI, , 2] ← sweep(signals[whichSetI, , 2], 2, coefsI2, '/')
    + typeIpeakMeans[2]
scaledSignals[whichSetII, , 1] ←sweep(signals[whichSetII, , 1], 2, coefsII1, '/'
    ) + typeIIpeakMeans[1]
scaledSignals[whichSetII, , 2] ←sweep(signals[whichSetII, , 2], 2, coefsII2, '/'
    ) + typeIIpeakMeans[2]
scaledSignals[scaledSignals < 0] ← 0

# stop here if omitting loess fitting ------------------------------------
if (!fitLoess){
  out ← object
  normedUnmeth ← scaledSignals[, , 1]
  normedMeth ← scaledSignals[, , 2]
  rownames(normedUnmeth) ← rownames(normedMeth) ← rownames(object)
  colnames(normedUnmeth) ← colnames(normedMeth) ← colnames(object)

  assayDataElement(out, 'Unmeth') ← normedUnmeth
  assayDataElement(out, 'Meth') ← normedMeth

  out@preprocessMethod ← c(rg.norm = sprintf("fresco alignment and scaling (
      based on a MethylSet preprocessed as '%s'",
                                             preprocessMethod(object)[1]),
                           minfi = as.character(packageVersion('minfi')),
                           manifest = as.character(packageVersion('
                               IlluminaHumanMethylation450kmanifest')))

  return(out)
}

# compute robust experiment average
    ------------------------------------------
if (verbose) cat('Computing robust experiment-wise average\n')
log2Centered ← log2(scaledSignals + 1)
```

```
sexInd ← factor(suppressWarnings(getSex(mapToGenome(object))[, 3]))
XYind ← which(frescoData$chromosome %in% c('X', 'Y'))
log2Standard ← apply(log2Centered, c(1, 3), mean, trim = .1)

if (nlevels(sexInd) == 2){
  mInd ← which(sexInd == 'M')
  fInd ← which(sexInd == 'F')

  log2StandardM ← log2StandardF ← log2Standard
  log2StandardM[XYind, ] ← apply(log2Centered[XYind, mInd, ], c(1, 3), mean,
      trim = .1)
  log2StandardF[XYind, ] ← apply(log2Centered[XYind, fInd, ], c(1, 3), mean,
      trim = .1)
}

# compute deviations from average ----------------------------------------
if (verbose) cat('Computing deviations from average \n')
log2Deviations ← array(dim = dim(log2Centered))

for(kk in 1:2)
  log2Deviations[, , kk] ← log2Centered[, , kk] - log2Standard[, kk]

if (nlevels(sexInd) == 2){
  for (kk in 1:2){
    log2Deviations[XYind, mInd, kk] ← log2Centered[XYind, mInd, kk] -
        log2StandardM[XYind, kk]
    log2Deviations[XYind, fInd, kk] ← log2Centered[XYind, fInd, kk] -
        log2StandardF[XYind, kk]
  }
}

# winsorize by probe type
    ------------------------------------------------------------
if (useControls){
  if (verbose) cat('Winsorizing probes out of prediction range \n')

  GC[whichSetI] ← winsorizeBySubset(GC, whichSetI, whichControlsI)
  GC[whichSetII] ← winsorizeBySubset(GC, whichSetII, whichControlsII)

  for (kk in 1:2){
    log2Standard[whichSetI, kk] ← winsorizeBySubset(log2Standard[, kk],
        whichSetI, whichControlsI)
    log2Standard[whichSetII, kk] ← winsorizeBySubset(log2Standard[, kk],
        whichSetII, whichControlsII)
  }
}

# create independent variable data frame for loess
    ---------------------------------
indepVars ← data.frame(GC = GC, UMavg = log2Standard[, 1], Mavg = log2Standard[,
     2])

if(nlevels(sexInd) == 2){
  indepVarsM ← data.frame(GC = GC, UMavg = log2StandardM[, 1], Mavg =
      log2StandardM[, 2])
  indepVarsF ← data.frame(GC = GC, UMavg = log2StandardF[, 1], Mavg =
      log2StandardF[, 2])
}

# fit loess surfaces
    ------------------------------------------------------------------
if (verbose) cat('Fitting & subtracting out loess\n')
```

```
if (nlevels(sexInd) == 1){
  log2NormedDevs <- array(dim = dim(log2Deviations))

  if (verbose) cat('Normalizing type I probes \n')
  log2NormedDevs[whichSetI, , ] <- apply(log2Deviations, c(2, 3), funLoess,
                                         indepVars = indepVars, whichControls =
                                             whichControlsI,
                                         whichSet = whichSetI,
                                             smoothingParameter = loessSpan)

  if (verbose) cat('Normalizing type II probes \n')
  log2NormedDevs[whichSetII, , ] <- apply(log2Deviations, c(2, 3), funLoess,
                                          indepVars = indepVars, whichControls =
                                              whichControlsII,
                                          whichSet = whichSetII,
                                              smoothingParameter = loessSpan)
}

if (nlevels(sexInd) == 2){
  log2NormedDevs <- array(dim = dim(log2Deviations))

  if (verbose) cat('Normalizing type I probes \n')
  # type I
  log2NormedDevs[whichSetI, mInd, ] <- apply(log2Deviations[, mInd, ], c(2, 3),
      funLoess,
                                             indepVars = indepVarsM,
                                                 whichControls = whichControlsI,
                                             whichSet = whichSetI,
                                                 smoothingParameter = loessSpan)

  log2NormedDevs[whichSetI, fInd, ] <- apply(log2Deviations[, fInd, ], c(2, 3),
      funLoess,
                                             indepVars = indepVarsF,
                                                 whichControls = whichControlsI,
                                             whichSet = whichSetI,
                                                 smoothingParameter = loessSpan)

  # type II
  if (verbose) cat('Normalizing type II probes \n')
  log2NormedDevs[whichSetII, mInd, ] <- apply(log2Deviations[, mInd, ], c(2, 3),
      funLoess,
                                              indepVars = indepVarsM,
                                                  whichControls =
                                                  whichControlsII,
                                              whichSet = whichSetII,
                                                  smoothingParameter = loessSpan
                                                  )

  log2NormedDevs[whichSetII, fInd, ] <- apply(log2Deviations[, fInd, ], c(2, 3),
      funLoess,
                                              indepVars = indepVarsF,
                                                  whichControls =
                                                  whichControlsII,
                                              whichSet = whichSetII,
                                                  smoothingParameter = loessSpan
                                                  )

}

# compute normalized log2 signals
    ------------------------------------------------
log2NormedSignals <- array(dim = dim(log2Centered))
if (nlevels(sexInd) == 1){
  for (kk in 1:2) log2NormedSignals[, , kk] <- log2NormedDevs[, , kk] +
```

```
        log2Standard[, kk]
    rm(log2NormedDevs, log2Standard); gc()
  }
  if (nlevels(sexInd) == 2){
    for(kk in 1:2){
      log2NormedSignals[, mInd, kk] <- log2NormedDevs[, mInd, kk] + log2StandardM[,
          kk]
      log2NormedSignals[, fInd, kk] <- log2NormedDevs[, fInd, kk] + log2StandardF[,
          kk]
    }
    rm(log2NormedDevs, log2StandardM, log2StandardF); gc()
  }

  # create new MethylSet for output
    ------------------------------------------------
  out <- object
  normedUnmeth <- 2^log2NormedSignals[, , 1]
  normedMeth <- 2^log2NormedSignals[, , 2]
  rownames(normedUnmeth) <- rownames(normedMeth) <- rownames(object)
  colnames(normedUnmeth) <- colnames(normedMeth) <- colnames(object)

  assayDataElement(out, 'Unmeth') <- normedUnmeth
  assayDataElement(out, 'Meth') <- normedMeth

  out@preprocessMethod <- c(rg.norm = sprintf("fresco alignment, scaling, and
      surface fitting (based on a MethylSet preprocessed as '%s'",
                                          preprocessMethod(object)[1]),
                            minfi = as.character(packageVersion('minfi')),
                            manifest = as.character(packageVersion('
                                IlluminaHumanMethylation450kmanifest')))

  out
}


# declare peak finding function ------------------------------------------------

getLowerPeak <- function(x){
  intensityDensity <- density(x)
  peaksInd <- which(diff(sign(diff(intensityDensity$y)))==-2)+1
  lowerPeak <- which.min(intensityDensity$x[peaksInd])
  return(intensityDensity$x[peaksInd[lowerPeak]])
}

# declare winsorization function -----------------------------------------

winsorizeBySubset <- function(x, whichSet, whichControls){
  x[whichSet][which(x[whichSet] > max(x[whichControls]))] <- max(x[whichControls])
  x[whichSet][which(x[whichSet] < min(x[whichControls]))] <- min(x[whichControls])
  x[whichSet]
}

# declare loess function
  ------------------------------------------------------------
funLoess <- function(y, indepVars, whichControls, whichSet, smoothingParameter) {
  modelDat <- as.data.frame(cbind(y, indepVars))
  tempFit <- loess(y ~ GC * Mavg * UMavg, trace.hat = 'approx',
                   span = smoothingParameter,
                   modelDat, subset = whichControls)
  resids <- y[whichSet] - predict(tempFit, modelDat[whichSet, ])
  return(resids)
}
```

```
horseRace <- function(object, batchVarName = NULL,
                      covariateNames = NULL, covariateTypes = NULL,
                      compositeF=FALSE){

  if (!is(object, "RGChannelSet"))
    stop("object needs to be a 'RGChannelSet'")

  if (is.null(batchVarName) & is.null(covariateNames))
    stop("Please provide variable name for either batch or covariates of interest"
        )

  if (!is.null(covariateNames) & is.null(covariateTypes))
    stop("Please provide corresponding vector of covariate types: either
         'categorical' or 'continuous'")

  if (!is.null(covariateTypes) & any(!covariateTypes %in% c('categorical', '
      continuous')))
    stop("covariateTypes must be one of 'categorical' or 'continuous'")

  # normalize ------------------------------------------------------------
  normList <- list()
  normList$Raw        <- preprocessIllumina(updateObject(object))
  normList$FRESCO_15 <- preprocessFresco(normList$Raw, loessSpan = .15, sdThreshold
      = .1)
  normList$FRESCO_50 <- preprocessFresco(normList$Raw, loessSpan = .5, sdThreshold
      = .1)
  normList$FRESCO_85 <- preprocessFresco(normList$Raw, loessSpan = .85, sdThreshold
      = .1)
  normList$FRESCO_NL <- preprocessFresco(normList$Raw, fitLoess = FALSE,
      sdThreshold = .1)
  normList$SQN        <- preprocessQuantile(mapToGenome(object))
  normList$Funnorm    <- preprocessFunnorm(object)
  normList$Noob       <- preprocessNoob(object)

  # test for batch effects -----------------------------------------------
  if (!is.null(batchVarName)){
    f.results <- lapply(normList, .catTest, cvn=batchVarName)
    roc.results <- lapply(f.results, function(x) .rocComp(na.omit(x[, 2])))

    # p-value ecdf
    plot(0, 0, xlim = c(0, 1), ylim = c(0, 1), type='n',
         xlab = 'P-value', ylab = 'ECDF',
         main = 'P-value ECDF for Batch Effects')
    abline(0, 1, lty = 3)

    for(ii in 1:length(roc.results))
      lines(seq(0, 1, .01), roc.results[[ii]], col = ii)
    legend('bottomright', legend = names(roc.results), fill = 1:length(roc.results
        ))
  }

  # look at power for covariates of interest -----------------------------

  if (!is.null(covariateNames) & !compositeF){
    for(ii in 1:length(covariateNames)){

      if (covariateTypes[ii] == 'categorical'){
        f.results <- lapply(normList, .catTest, cvn=covariateNames[ii])
        roc.results <- lapply(f.results, function(x) .rocComp(na.omit(x[, 2]), BH.
            adj = TRUE))
      }

      if (covariateTypes[ii] == 'continuous'){
```

```
      f.results <- lapply(normList, .contTest, cvn=covariateNames[ii])
      roc.results <- lapply(f.results, function(X) .rocComp(na.omit(x[, 2]), BH.
          adj = TRUE))
    }

    # plot
    plot(0, 0, xlim = c(0, 1), ylim = c(0, 1), type='n',
         xlab = 'FDR Threshold', ylab = 'Prop sig at given FDR',
         main = paste('Significant Differences for', covariateNames[ii]))
    abline(0, 1, lty = 3)

    for(ii in 1:length(roc.results))
      lines(seq(0, 1, .01), roc.results[[ii]], col = ii)
    legend('bottomright', legend = names(roc.results), fill = 1:length(roc.
        results))
  }
}

# look at composite F-scores for covariates of interest --------------------

if (!is.null(covariateNames) & compositeF){
  for(ii in 1:length(covariateNames)){

    if (covariateTypes[ii] == 'categorical'){

      # compute anova sum of squares and get degrees of freedom
      f.results <- lapply(normList, .compSS, cvn=covariateNames[ii])
      p <- nlevels(factor(pData(normList[[1]])[, covariateNames[ii]]))
      n <- ncol(normList[[1]])

      # re-order f-results because mapToGenome re-orders the CpGs
      for(jj in 2:length(f.results)){
        f.results[[jj]] <- f.results[[jj]][match(rownames(normList[[1]]),
            rownames(normList[[jj]])) ,]
      }

      p.vals <- list()

      for(jj in 2:length(normList)){
        # compute composite F scores and p-values
        comp.pvals <- comp.f.stats <- matrix(nr=nrow(normList[[1]]), nc=3)
        colnames(comp.pvals) <- colnames(comp.f.stats) <- c('orig', 'ssr_raw', '
            sse_raw')
        comp.f.stats[, 1] <- (f.results[[1 ]][, 1] / (p-1) ) / (f.results[[1
            ]][, 2] / (n-p) )
        comp.f.stats[, 2] <- (f.results[[1 ]][, 1] / (p-1) ) / (f.results[[jj
            ]][, 2] / (n-p) )
        comp.f.stats[, 3] <- (f.results[[jj]][, 1] / (p-1) ) / (f.results[[1
            ]][, 2] / (n-p) )
        comp.pvals <- pf(comp.f.stats, df1=p-1, df2=n-p, lower.tail=FALSE)
        p.vals[[(jj-1)]] <- comp.pvals
      }
      names(p.vals) <- names(normList)[-1]

      # plot
      axis.lims <- -log10(unlist(p.vals))
      axis.lims <- max(axis.lims[which(is.finite(axis.lims))])

      par(mfcol = c(2, 5), mar=c(5, 4, 4, 1.5))

      for(jj in c(1, 4:7)){
        plot( -log10(p.vals[[jj]][, 1]), -log10(p.vals[[jj]][, 2]),
              pch=16, cex=.2, col=rgb(0,0,1,alpha=.4),
              xlab = 'Original F Statistic', ylab = expression('F'['Err']),
```

```
            main=names(p.vals)[jj], xlim=c(0, axis.lims), ylim=c(0, axis.lims),
            cex.axis = 1.7, cex.lab = 1.7)
      abline(0,1,col='red')

      plot( -log10(p.vals[[jj]][, 1]), -log10(p.vals[[jj]][, 3]),
            pch=16, cex=.2, col=rgb(0,0,1,alpha=.4),
            xlab = 'Original F Statistic', ylab = expression('F'['ES']),
            xlim=c(0, axis.lims), ylim=c(0, axis.lims),
            cex.axis = 1.7, cex.lab = 1.7)
      abline(0,1,col='red')

    }
  }

  if (covariateTypes[ii] == 'continuous'){
    cat('Not yet supported')
  }

 }
}


}


.rocComp <- function(x, eval = seq(0, 1, .01), BH.adj = FALSE){
  if (BH.adj) x <- p.adjust(x, method = 'BH')
  sapply(eval, function(z) sum(x < z) / length(x))
}

.catTest <- function(x, cvn) rowFtests(getBeta(x), factor(pData(x)[, cvn]))
.contTest <- function(x, cvn){
  cont.cov <- as.numeric(pData(x)[, cvn])
  apply(getBeta(x), 1, function(z) biglm(z ~ cont.cov))
}

.compSS <- function(x, cvn){
  cat.cov <- factor(pData(x)[, cvn])
  betas <- getBeta(x)
  fac.means <- matrix(nr=nrow(betas), nc=nlevels(cat.cov))
  res.mat <- matrix(nr=nrow(betas), nc=ncol(betas))
  mean.vec <- rowMeans(betas)

  for(ii in 1:nlevels(cat.cov)){
    fac.level <- which(cat.cov == levels(cat.cov)[ii])
    fac.means[, ii] <- rowMeans(betas[, fac.level])
    res.mat[, fac.level] <- betas[, fac.level] - fac.means[, ii]
  }

  sse <- rowSums(res.mat^2)
  ssr <- rowSums(sweep((fac.means - mean.vec)^2, 2, table(cat.cov), '*'))

  out <- cbind(ssr, sse)
  out
}
```

```
#'
#' @param object \code{MethylSet} object
#' @param useControls Should empirical controls be used to align and fit loess
    surfaces?
#' @param loessSpan Supply span for fitting loess surface
#' @param sdThreshold Threshold to filter empirical controls by standard deviation

returnFitStats <-function(object, useControls = TRUE, loessSpan = .15,
                          sdThreshold = .15, verbose = FALSE){

  if (loessSpan > 1 | loessSpan < 0) stop("loessSpan must be between zero and one"
      )

  data(frescoData)
  object <- fixMethOutliers(object)

  # create object for methylated and unmethylated channels
      -------------------------
  signals <- array(dim = c(dim(object), 2))
  signals[, , 1] <- getUnmeth(object)
  signals[, , 2] <- getMeth(object)
  frescoData <- frescoData[match(rownames(object), rownames(frescoData)), ]
  GC <- frescoData$targetGC

  # get set of empirical controls
      ------------------------------------------------
  if (useControls){
    probeSD <- rowSds(getBeta(object))
    controls <- which(!is.na(frescoData$eControls) & probeSD < sdThreshold)
    if (verbose) cat(length(controls), 'empirical control probes selected\n')
  }

  # divide probes and controls up by probe type
      ------------------------------------
  whichSetII <- which(frescoData$probeType == 'II')
  whichSetI <- which(frescoData$probeType == 'I')

  if (useControls){
    whichControlsII <- intersect(whichSetII, controls)
    whichControlsI <- intersect(whichSetI, controls)
  } else {
    whichControlsII <- whichSetII
    whichControlsI <- whichSetI
  }

  # find lower peaks
      ---------------------------------------------------------------------
  if (verbose) cat('Aligning signal intensities \n')
  typeIpeaks <- apply(signals[whichControlsI, , ], c(2, 3), getLowerPeak)
  typeIIpeaks <- apply(signals[whichControlsII, , ], c(2, 3), getLowerPeak)
  typeIpeakMeans <- colMeans(typeIpeaks)
  typeIIpeakMeans <- colMeans(typeIIpeaks)

  # line up samples by their lower peaks
      ---------------------------------------------
  signals[whichSetI, , 1] <- sweep(signals[whichSetI, , 1], 2, typeIpeaks[, 1], '-'
      )
  signals[whichSetI, , 2] <- sweep(signals[whichSetI, , 2], 2, typeIpeaks[, 2], '-'
      )
  signals[whichSetII, , 1] <- sweep(signals[whichSetII, , 1], 2, typeIIpeaks[, 1],
      '-')
  signals[whichSetII, , 2] <- sweep(signals[whichSetII, , 2], 2, typeIIpeaks[, 2],
      '-')
```

```
# scale signals to minimize deviance from control averages
    ------------------------
if (verbose) cat('Applying linear scaling factor \n')
typeIcontrolAvg ← apply(signals[whichControlsI, , ], c(1, 3), mean)
typeIIcontrolAvg ← apply(signals[whichControlsII, , ], c(1, 3), mean)

coefsI1 ← lm(signals[whichControlsI, , 1] ~ typeIcontrolAvg[, 1] + 0)$coef
coefsI2 ← lm(signals[whichControlsI, , 2] ~ typeIcontrolAvg[, 2] + 0)$coef
coefsII1 ← lm(signals[whichControlsII, , 1] ~ typeIIcontrolAvg[, 1] + 0)$coef
coefsII2 ← lm(signals[whichControlsII, , 2] ~ typeIIcontrolAvg[, 2] + 0)$coef

scaledSignals ← array(dim = dim(signals))
scaledSignals[whichSetI, , 1] ← sweep(signals[whichSetI, , 1], 2, coefsI1, '/')
    + typeIpeakMeans[1]
scaledSignals[whichSetI, , 2] ← sweep(signals[whichSetI, , 2], 2, coefsI2, '/')
    + typeIpeakMeans[2]
scaledSignals[whichSetII, , 1] ←sweep(signals[whichSetII, , 1], 2, coefsII1, '/'
    ) + typeIIpeakMeans[1]
scaledSignals[whichSetII, , 2] ←sweep(signals[whichSetII, , 2], 2, coefsII2, '/'
    ) + typeIIpeakMeans[2]
scaledSignals[scaledSignals < 0] ← 0

# compute robust experiment average
    ---------------------------------------------
if (verbose) cat('Computing robust experiment-wise average\n')
log2Centered ← log2(scaledSignals + 1)

sexInd ← factor(suppressWarnings(getSex(mapToGenome(object))[, 3]))
XYind ← which(frescoData$chromosome %in% c('X', 'Y'))
log2Standard ← apply(log2Centered, c(1, 3), mean, trim = .1)

if (nlevels(sexInd) == 2){
  mInd ← which(sexInd == 'M')
  fInd ← which(sexInd == 'F')

  log2StandardM ← log2StandardF ← log2Standard
  log2StandardM[XYind, ] ← apply(log2Centered[XYind, mInd, ], c(1, 3), mean,
      trim = .1)
  log2StandardF[XYind, ] ← apply(log2Centered[XYind, fInd, ], c(1, 3), mean,
      trim = .1)
}

# compute deviations from average ----------------------------------------
if (verbose) cat('Computing deviations from average \n')
log2Deviations ← array(dim = dim(log2Centered))

for(kk in 1:2)
  log2Deviations[, , kk] ← log2Centered[, , kk] - log2Standard[, kk]

if (nlevels(sexInd) == 2){
  for (kk in 1:2){
    log2Deviations[XYind, mInd, kk] ← log2Centered[XYind, mInd, kk] -
        log2StandardM[XYind, kk]
    log2Deviations[XYind, fInd, kk] ← log2Centered[XYind, fInd, kk] -
        log2StandardF[XYind, kk]
  }
}

# winsorize by probe type
    ------------------------------------------------------------
if (useControls){
  if (verbose) cat('Winsorizing probes out of prediction range \n')

  GC[whichSetI] ← winsorizeBySubset(GC, whichSetI, whichControlsI)
```

```
  GC[whichSetII] ← winsorizeBySubset(GC, whichSetII, whichControlsII)

  for (kk in 1:2){
    log2Standard[whichSetI, kk] ← winsorizeBySubset(log2Standard[, kk],
        whichSetI, whichControlsI)
    log2Standard[whichSetII, kk] ← winsorizeBySubset(log2Standard[, kk],
        whichSetII, whichControlsII)
  }
}

# create independent variable data frame for loess
    ---------------------------------
indepVars ← data.frame(GC = GC, UMavg = log2Standard[, 1], Mavg = log2Standard[,
    2])

if(nlevels(sexInd) == 2){
  indepVarsM ← data.frame(GC = GC, UMavg = log2StandardM[, 1], Mavg =
      log2StandardM[, 2])
  indepVarsF ← data.frame(GC = GC, UMavg = log2StandardF[, 1], Mavg =
      log2StandardF[, 2])
}

# fit loess surfaces
    ---------------------------------------------------------------------
if (verbose) cat('Fitting & subtracting out loess\n')

if (nlevels(sexInd) == 1){
  if (verbose) cat('Normalizing type I probes \n')
  typeInormed ← apply(log2Deviations, c(2, 3), funLoessSS,
                      indepVars = indepVars, whichControls = whichControlsI,
                      whichSet = whichSetI, smoothingParameter = loessSpan)

  if (verbose) cat('Normalizing type II probes \n')
  typeIInormed ← apply(log2Deviations, c(2, 3), funLoessSS,
                       indepVars = indepVars, whichControls = whichControlsII,
                       whichSet = whichSetII, smoothingParameter = loessSpan)

  estimatedErrorVar ← list(typeInormed, typeIInormed)
  return(estimatedErrorVar)
}

if (nlevels(sexInd) == 2){
  if (verbose) cat('Normalizing type I probes \n')
  typeInormed ← typeIInormed ← array(dim = c(3, dim(log2Deviations)[2], 2))
  # type I
  typeInormed[, mInd, ] ← apply(log2Deviations[, mInd, ], c(2, 3), funLoessSS,
                                indepVars = indepVarsM, whichControls =
                                    whichControlsI,
                                whichSet = whichSetI, smoothingParameter =
                                    loessSpan)

  typeInormed[, fInd, ] ← apply(log2Deviations[, fInd, ], c(2, 3), funLoessSS,
                                indepVars = indepVarsF, whichControls =
                                    whichControlsI,
                                whichSet = whichSetI, smoothingParameter =
                                    loessSpan)

  # type II
  if (verbose) cat('Normalizing type II probes \n')
  typeIInormed[, mInd, ] ← apply(log2Deviations[, mInd, ], c(2, 3), funLoessSS,
                                 indepVars = indepVarsM, whichControls =
                                     whichControlsII,
                                 whichSet = whichSetII, smoothingParameter =
                                     loessSpan)
```

```
    typeIInormed[, fInd, ] <- apply(log2Deviations[, fInd, ], c(2, 3), funLoessSS,
                                    indepVars = indepVarsF, whichControls =
                                        whichControlsII,
                                    whichSet = whichSetII, smoothingParameter =
                                        loessSpan)

    estimatedErrorVar <- list(typeInormed, typeIInormed)
    return(estimatedErrorVar)
  }

}

# declare loess function
funLoessSS <- function(y, indepVars, whichControls, whichSet, smoothingParameter) {
  modelDat <- as.data.frame(cbind(y, indepVars))
  tempFit <- loess(y ~ GC * Mavg * UMavg, trace.hat = 'approx', span =
      smoothingParameter,
                   modelDat, subset = whichControls)
  traceL <- tempFit$trace.hat
  sigma2 <- sum(tempFit$residuals^2)/(tempFit$n - 1)
  aicc <- log(sigma2) + 1 + 2 * (2 * (traceL + 1))/(tempFit$n - traceL - 2)
  gcv <- tempFit$n * sigma2/(tempFit$n - traceL)^2
  r2 <- cor(tempFit$y, tempFit$fitted)^2
  return(c(aicc, gcv, r2))
}
```

```
#'
#' @param object \code{MethylSet} object
#' @param sdThreshold Standard deviation cut-off for filtering empirical controls
#'
#' @export empiricalControlCoverage


empiricalControlCoverage <- function(object, sdThreshold = .15){

  if (!is(object, "MethylSet")) stop("'object' needs to be a 'MethylSet'")

  data(frescoData)

  # create object for methylated and unmethylated channels ------------------
  methTmp <- getMeth(object)
  probeIDs <- rownames(methTmp)
  signals <- array(dim = c(dim(methTmp), 2))
  signals[, , 1] <- getUnmeth(object)
  signals[, , 2] <- methTmp
  frescoData <- frescoData[match(probeIDs, rownames(frescoData)), ]
  GC <- frescoData$targetGC

  log2Centered <- apply(log2(signals + 1), c(1, 3), mean)

  # filter controls and create indicator variables ---------------------------
  probeSD <- rowSds(getBeta(object))
  frescoData$eControls[probeSD > sdThreshold] <- NA
  typeI <- which(frescoData$probeType == 'I')
  typeII <- which(frescoData$probeType == 'II')
  hemEC <- which(frescoData$eControls == 'Hemimethylated')
  methEC <- which(frescoData$eControls == 'Methylated')
  umethEC <- which(frescoData$eControls == 'Unmethylated')

  par(mfrow = c(2, 3))
  controlCex <- .7

  # type I probes M & UM
  smoothScatter(log2Centered[typeI, 2:1], xlab = 'log2(Methylated Signal)',
                ylab = 'log2(Unmethylated Signal)')

  points(log2Centered[intersect(methEC, typeI), 2:1],
         pch = 16, cex = controlCex, col = 'red')

  points(log2Centered[intersect(umethEC, typeI), 2:1],
         pch = 16, cex = controlCex, col = 'green')

  points(log2Centered[intersect(hemEC, typeI), 2:1],
         pch = 16, cex = controlCex, col = 'yellow')

  # type I probes M & GC
  smoothScatter(log2Centered[typeI, 2], GC[typeI],
                xlab = 'log2(Methylated Signal)',
                ylab = 'Target GC Content',
                main = 'Type I Probes')

  points(log2Centered[intersect(methEC,typeI), 2],
         GC[intersect(methEC, typeI)],
         pch = 16, cex = controlCex, col = 'red')

  points(log2Centered[intersect(umethEC,typeI), 2],
         GC[intersect(umethEC, typeI)],
         pch = 16, cex = controlCex, col = 'green')

  points(log2Centered[intersect(hemEC,typeI), 2],
```

```
       GC[intersect(hemEC, typeI)],
       pch = 16, cex = controlCex, col = 'yellow')

# type I UM & GC
smoothScatter(log2Centered[typeI, 1], GC[typeI],
              xlab = 'log2(Unmethylated Signal)',
              ylab = 'Target GC Content')

points(log2Centered[intersect(methEC, typeI), 1],
       GC[intersect(methEC, typeI)],
       pch = 16, cex = controlCex, col = 'red')

points(log2Centered[intersect(umethEC, typeI), 1],
       GC[intersect(umethEC, typeI)],
       pch = 16, cex = controlCex, col = 'green')

points(log2Centered[intersect(hemEC, typeI), 1],
       GC[intersect(hemEC, typeI)],
       pch = 16, cex = controlCex, col = 'yellow')

# type II probes M & UM
smoothScatter(log2Centered[typeII, 2:1], xlab = 'log2(Methylated Signal)',
              ylab = 'log2(Unmethylated Signal)')

points(log2Centered[intersect(methEC,typeII), 2:1],
       pch = 16, cex = controlCex, col = 'red')

points(log2Centered[intersect(umethEC, typeII), 2:1],
       pch = 16, cex = controlCex, col = 'green')

points(log2Centered[intersect(hemEC, typeII), 2:1],
       pch = 16, cex = controlCex, col = 'yellow')

# type II M & GC
smoothScatter(log2Centered[typeII, 2], GC[typeII],
              xlab = 'log2(Methylated Signal)',
              ylab = 'Target GC Content',
              main = 'Type II Probes')

points(log2Centered[intersect(methEC, typeII), 2],
       GC[intersect(methEC, typeII)],
       pch = 16, cex = controlCex, col = 'red')

points(log2Centered[intersect(umethEC, typeII), 2],
       GC[intersect(umethEC, typeII)],
       pch = 16, cex = controlCex, col = 'green')

points(log2Centered[intersect(hemEC, typeII), 2],
       GC[intersect(hemEC, typeII)],
       pch = 16, cex = controlCex, col = 'yellow')

# type II UM & GC
smoothScatter(log2Centered[typeII, 1], GC[typeII],
              xlab = 'log2(Unmethylated Signal)',
              ylab = 'Target GC Content')

points(log2Centered[intersect(methEC, typeII), 1],
       GC[intersect(methEC, typeII)],
       pch = 16, cex = controlCex, col = 'red')

points(log2Centered[intersect(umethEC, typeII), 1],
       GC[intersect(umethEC, typeII)],
       pch = 16, cex = controlCex, col = 'green')
```

```
  points(log2Centered[intersect(hemEC, typeII), 1],
         GC[intersect(hemEC, typeII)],
         pch = 16, cex = controlCex, col = 'yellow')

}
```

```
#'
#' @param object \code{MethylSet} object
#' @param sdThreshold Standard deviation cut-off for filtering empirical controls
#'
#' @export empiricalControlQA


empiricalControlQA <- function(object, sdThreshold = .15){

  if (!is(object, "MethylSet")) stop("'object' needs to be a 'MethylSet'")

  data(frescoData)

  # pull out control probes and get average ----------------------------------------
  betaVals <- getBeta(object)
  frescoData <- frescoData[match(rownames(betaVals), rownames(frescoData)), ]
  controlInd <- which(!is.na(frescoData$eControls))
  controlInd <- intersect(controlInd, which(rowSums(is.na(betaVals)) == 0))
  means <- rowMeans(betaVals[controlInd,])

  # plot sorted control probes as heat map -----------------------------------------
  par(mfrow = c(1, 3))
  image(betaVals[controlInd[order(means)], ], axes = FALSE,
        main = 'Empirical Control Probe QC',
        xlab = 'CpGs ordered by avg methylation',
        ylab = 'Samples')
  lines(seq(0, 1, length.out = length(means)),
        means[order(means)], col = 1)

  # plot control probe standard deviations -----------------------------------------
  controlsSD <- rowSds(betaVals[controlInd[order(means)], ])

  plot(density(controlsSD),
       main = 'Empirical Control Probe Standard Deviations',
       xlab='Standard Deviation')

  abline(v = sdThreshold)
  cat(paste(sum(controlsSD < sdThreshold), 'of',
            length(controlInd), 'controls remaining'))

  image(betaVals[controlInd[order(means)], ][which(controlsSD < sdThreshold), ],
        axes = FALSE,
        main = 'Filtered Empirical Control Probes',
        xlab = 'CpGs ordered by avg methylation',
        ylab = 'Samples')

  lines(seq(0, 1, length.out = length(which(controlsSD < sdThreshold))),
        means[order(means)][which(controlsSD < sdThreshold)], col = 1)
}
```

```
#'
#' @param object \code{MethylSet} or \code{GenomicRatioSet} object
#' @param removeChromosomes A character string of chromoses to remove
#' @param filterCrossHyb Filter autosomal probes that cross-hybridize to sex
    chromosomes?
#' @param filterNA Filter probes containing at least one NA?
#' @param filterSNP Filter probes containing SNPs?
#' @param minorAlleleFreq What is the largest minor allele frequency we are
    willing to tolerate?
#' @param population What population should be used to compute minor allele
    frequency?
#' Default is 'All'
#'
#' @export filterCpGs

filterCpGs <- function(object, removeChromosomes = NULL, filterCrossHyb = TRUE,
                       filterNA = TRUE, filterSNP = TRUE,
                       minorAlleleFreq = 0, population = 'All'){

  if (sum(!class(object) %in% c("MethylSet", "GenomicRatioSet")) > 0){
    stop("'object' needs to be a 'MethylSet' or 'GenomicRatioSet'")
  }

  populationAF <- c('All', 'African', 'American', 'Asian', 'European')
  if (!population %in% populationAF){
    stop("population' must be one of: 'All', 'African', 'American', 'Asian', or '
        European'")
  }

  if (sum(!removeChromosomes %in% c('X', 'Y', 1:22)) > 0){
    stop("'removeChromosomes' needs to be a list of
         chromosomes to remove e.g. c('X', '1')")
  }

  data(frescoData)

  removeProbes <- NULL
  if (is(object, 'MethylSet')) probeIDs <- rownames(getMeth(object))
  if (is(object, 'GenomicRatioSet')) probeIDs <- rownames(getM(object))

  frescoData <- frescoData[match(probeIDs, rownames(frescoData)), ]

  if (length(removeChromosomes) > 0){
    removeProbes <- c(removeProbes, probeIDs[which(frescoData$chromosome %in%
        removeChromosomes)])
  }

  if (filterCrossHyb){
    removeProbes <- c(removeProbes, probeIDs[which(frescoData$crossHyb)])
  }

  if (filterNA){
    NAind <- probeIDs[which(rowSums(is.na(getBeta(object))) > 0)]
    removeProbes <- c(removeProbes, probeIDs[NAind])
  }

  if (filterSNP){
    AFtype <- match(population, populationAF) + 4
    SNPind <- which(frescoData[, AFtype] > minorAlleleFreq)
    removeProbes <- c(removeProbes, probeIDs[SNPind])
  }

  removeProbes <- unique(removeProbes)
  keepCpGs <- setdiff(probeIDs, removeProbes)
```

```
  if(is(object, 'MethylSet')){
    out ← object
    assayDataElement(out, 'Unmeth') ← getUnmeth(object)[keepCpGs, ]
    assayDataElement(out, 'Meth') ← getMeth(object)[keepCpGs, ]
    return(out)
  }

  if(is(object, 'GenomicRatioSet')){

    out ← GenomicRatioSet(gr = rowData(object)[keepCpGs],
                          Beta = NULL,
                          M = getM(object)[keepCpGs, ],
                          CN = getCN(object[keepCpGs, ]),
                          pData = pData(object),
                          annotation = annotation(object),
                          preprocessMethod = preprocessMethod(object))

    return(out)
  }

}
```

```
#'
#' @param object a \code{MethylSet} object
#' @param useControls Should empirical controls be used to align and fit loess
    surfaces?
#' @param loessSpan Supply vector of possible spans for fitting loess surface
#' @param sdThreshold Threshold to filter empirical controls by standard deviation
#'
#' @export plotFitStats


plotFitStats <- function(object, useControls = TRUE,
                         loessSpan = seq(.05, .95, .15), sdThreshold = .15){

  if (!is(object, "MethylSet")) stop("'object' needs to be a 'MethylSet'")

  fitstats <- list()

  # generate fit statistics ------------------------------------------------
  for(ii in 1:length(loessSpan)){
    fitstats[[ii]] <- returnFitStats(object, useControls = useControls,
                                     loessSpan = loessSpan[ii], sdThreshold =
                                         sdThreshold)
    cat(ii, 'of', length(loessSpan), '\n\n')
  }

  par(mfrow = c(2,2))

  # generate gcv curves ----------------------------------------------------
  chType <- c('UM', 'M')
  statType <- c('AICC', 'GCV', 'R^2')
  for(thisStat in 1:3){
    for(probeType in 1:2){
      for(channelType in 1:2){

      gcvCurves <- NULL
      for(ii in 1:length(fitstats))
        gcvCurves <- cbind(gcvCurves, fitstats[[ii]][[probeType]][thisStat, ,
            channelType])

      matplot(loessSpan, t(gcvCurves), type='l',
              main = paste('Type ', probeType, ' : ',
                           chType[channelType], ' channel : ',
                           statType[thisStat], sep = '')
              )
      }
    }
  }

}
```

Appendix C

CODE FROM R PACKAGE GDI

```
# Define ------------------------------------------------------------------

#' @exportClass GDset

setOldClass('ffdf')
setOldClass("data.frame")
setClassUnion("data.frameORffdf", c("data.frame", "ffdf"))
setClass("GDset",
         slots = c(dat = 'data.frameORffdf',
                   annot = "GRanges",
                   pheno = 'data.frame',
                   platform = "character"
                   ))

# Validate ----------------------------------------------------------------

.validGDset <- function(object) {

  # Required annotation columns
  annot.req <- c(entrez.id = "character")

  # Check for required annotation column(s)
  md <- mcols(object@annot)
  if (!all(names(annot.req) %in% names(md))) {
    stop("GDset slot 'annot' must contain all of the following columns:\n",
         paste(names(annot.req), collapse = "\n"), call. = FALSE)
  }

  # Check that annotation matches data
  if (!identical(rownames(object@dat), names(object@annot))){
    stop("Names of 'annot' must match rownames of 'experimentData'", call. = FALSE
         )
  }

  # Check that meta data matches data
  if (!identical(colnames(object@dat), rownames(object@pheno))){
    stop("colnames of 'dat' must match rownames of 'pheno'")
  }

  return(TRUE)
}

setValidity("GDset", .validGDset)

# Constructors ------------------------------------------------------------

GDset <- function(dat, annot, pheno, platform){

  new("GDset",
      dat = dat,
      pheno = pheno,
      platform = platform,
      annot = annot)
}
```

```
# Accessors ------------------------------------------------------------

#' @export getPlatform
setMethod("getPlatform", "GDset", function(object) object@platform)

#' @export getAnnot
setMethod("getAnnot", "GDset", function(object) object@annot)

#' @export getPheno
setMethod("getPheno", "GDset", function(object) object@pheno)

#' @export getDat
setMethod("getDat", "GDset", function(object) object@dat)

setMethod("[", c("GDset", "ANY", "ANY"),
          function(x, i, j, ..., drop = FALSE){

            new.dat <- x@dat[i, j, drop=FALSE]
            new("GDset", annot = x@annot[i],
                dat = new.dat,
                pheno = x@pheno[j, , drop=FALSE],
                platform = x@platform)

          })

setMethod("[", c("GDset", "missing", "ANY"),
          function(x, i, j, ..., drop = FALSE){

            new.dat <- new.dat <- x@dat[, j, drop=FALSE]

            new("GDset", annot = x@annot,
                dat = new.dat,
                pheno = x@pheno[j, , drop=FALSE],
                platform = x@platform)
          })

setMethod("[", c("GDset", "ANY", "missing"),
          function(x, i, j, ..., drop = FALSE){

            new.dat <- x@dat[i, , drop=FALSE]


            new("GDset", annot = x@annot[i],
                dat = new.dat,
                pheno = x@pheno[, , drop=FALSE],
                platform = x@platform)
          })

# Summaries ------------------------------------------------------------

setMethod("show", "GDset", function(object) {
  cat("A GDset object \n")
  cat("Platform:", object@platform, "\n")
  cat("Data contains: \n")
  cat("  ", nrow(object@dat), "loci \n")
  cat("  ", ncol(object@dat), "samples \n")
  cat("With", ncol(object@pheno), "Covariates:\n")
  cat(colnames(object@pheno), '\n')
})

setMethod("dim", "GDset", function(x){
  c(loci = nrow(x@dat), samples = ncol(x@dat))
})
```

```
# Define ------------------------------------------------------------------

#' @exportClass GDIset
setClass("GDIset", slots = c(set1 = "GDset", set2 = "GDset"))

# Validate ----------------------------------------------------------------

.validGDIset <- function(object) {
  if (!identical(object@set1@pheno, object@set2@pheno))
    stop("'pheno' must match between GDsets", call. = FALSE)
  return(TRUE)
}

setValidity("GDIset", .validGDIset)

# Constructors ------------------------------------------------------------

GDIset <- function(x, y){
  new("GDIset", set1 = x, set2 = y)
}

# Accessors ---------------------------------------------------------------

#' @exportMethod getPheno
setMethod("getPheno", "GDIset", function(object) object@set1@pheno)

#' @exportMethod getDat
setMethod("getDat", "GDIset", function(object){
  out <- list(object@set1@dat, object@set2@dat)
  names(out) <- c(object@set1@platform, object@set2@platform)
  out
})

#' @exportMethod getPlatform
setMethod("getPlatform", "GDIset", function(object){
  list(set1 = object@set1@platform, set2 = object@set2@platform)})

#' @exportMethod getAnnot
setMethod("getAnnot", "GDIset", function(object){
  out <- list(object@set1@annot, object@set2@annot)
  names(out) <- c(object@set1@platform, object@set2@platform)
  out
})

setMethod("[", c("GDIset", "ANY", "ANY"), function(x, i, j, ..., drop = FALSE){

  if (!is(i, 'character')){
    stop('Row index must be a vector of entrez ids')
  } else {
    print('Please make sure you are subsetting by entrez id')
  }

  GDset1 <- x@set1[which(x@set1@annot$entrez.id %in% i), j]
  GDset2 <- x@set2[which(x@set2@annot$entrez.id %in% i), j]

  new("GDIset", set1 = GDset1, set2 = GDset2)
})

setMethod("[", c("GDIset", "missing", "ANY"), function(x, i, j, ..., drop = FALSE)
    {

  GDset1 <- x@set1[, j]
  GDset2 <- x@set2[, j]
  new("GDIset", set1 = GDset1, set2 = GDset2)
```

```
})

setMethod("[", c("GDIset", "ANY", "missing"), function(x, i, j, ..., drop = FALSE)
    {

  if (!is(i, 'character')){
    stop('Row index must be a vector of entrez ids')
  } else {
    print('Please make sure you are subsetting by entrez id')
  }

  GDset1 <- x@set1[which(x@set1@annot$entrez.id %in% i), ]
  GDset2 <- x@set2[which(x@set2@annot$entrez.id %in% i), ]

  new("GDIset", set1 = GDset1, set2 = GDset2)
})

getSet <- function(object, whichset = 1){
  if (!is(object, "GDIset"))
    stop("object must be a 'GDIset'")

  if (whichset == 1) return(object@set1)
  if (whichset == 2) return(object@set2)
}


# Summaries ------------------------------------------------------------------

setMethod("show", "GDIset", function(object){
  cat("A GDIset object containing \n\n")
  print(object@set1)
  cat("\n\n")
  print(object@set2)

})

setMethod("dim", "GDIset", function(x){
  out <- list(c(loci = nrow(x@set1@dat), samples = ncol(x@set1@dat)),
              c(loci = nrow(x@set2@dat), samples = ncol(x@set2@dat)))
  names(out) <- getPlatform(x)
  out
})

# consolidate GDI set --------------------------------------------------------

consolidate <- function(object){

  if (!is(object, "GDIset")) stop("object needs to be a 'GDIset'")

  genes <- lapply(getAnnot(object), function(x) x$entrez.id)
  has.both <- intersect(genes[[1]], genes[[2]])

  set1.include <- which(genes[[1]] %in% has.both)
  set2.include <- which(genes[[2]] %in% has.both)

  set1.annot <- object@set1@annot[set1.include]
  set2.annot <- object@set2@annot[set2.include]

  subs1 <- 1:nrow(object@set1@dat) %in% set1.include
  subs2 <- 1:nrow(object@set2@dat) %in% set2.include

  set1.dat <- subset(object@set1@dat, subset=subs1)
  set2.dat <- subset(object@set2@dat, subset=subs2)
```

```
  rownames(set1.dat) ← rownames(object@set1@dat)[set1.include]
  rownames(set2.dat) ← rownames(object@set2@dat)[set2.include]

  GDset1 ← GDset(dat = set1.dat,
                 annot = set1.annot,
                 pheno = object@set1@pheno,
                 platform = object@set1@platform)

  GDset2 ← GDset(dat = set2.dat,
                 annot = set2.annot,
                 pheno = object@set2@pheno,
                 platform = object@set2@platform)

  GDIset(GDset1, GDset2)
}
```

```
### cross covariance test

ccaTest <- function(object, npcs = 5, min.set1=5, min.set2=3, cc.pvalue.threshold
    =.1){

  if (!is(object, 'GDIset')) stop("object must be a 'GDIset'")

  # combine data sets --------------------------------
  set1.df <- object@set1@dat
  if (is(object@set1@dat, 'ffdf')){
    set1.df$set <- ff(factor(rep('set1', nrow(set1.df))))
  } else {
    set1.df$set <- factor(rep('set1', nrow(set1.df)))
  }

  set2.df <- object@set2@dat
  if (is(object@set2@dat, 'ffdf')){
    set2.df$set <- ff(factor(rep('set2', nrow(set2.df))))
  } else {
    set2.df$set <- factor(rep('set2', nrow(set2.df)))
  }

  if (is(set1.df, 'ffdf') & is(set2.df, 'ffdf')){
    full.set <- ffdfappend(set1.df, set2.df)
  } else {
    full.set <- rbind(as.data.frame(set1.df), as.data.frame(set2.df))
  }

  # do analysis grouped by gene ----------------------------------------
  entrez.ids <- c(object@set1@annot$entrez.id, object@set2@annot$entrez.id)
  ids.in.both <- intersect(object@set1@annot$entrez.id, object@set2@annot$entrez.id
      )
  unique.ids <- unique(ids.in.both)
  ind <- 1
  out.return <- list()

#   out <- foreach(gene = unique.ids, .packages='gdi') %do% {
  for(gene in unique.ids){

    cat(gene, ' ', ind)

    dat <- full.set[which(entrez.ids == gene), ]
    n.sites <- table(dat$set)
    set1 <- as.matrix(dat[dat$set == 'set1', -ncol(dat)])
    set2 <- as.matrix(dat[dat$set == 'set2', -ncol(dat)])

    set1 <- apply(set1, 1, na2mean)
    set2 <- apply(set2, 1, na2mean)

    if (n.sites[1] < min.set1 | n.sites[2] < min.set2){
      out.return[[gene]] <- NA
      cat(' omitted')
    } else {

      # perform PCA for each set
      pca.set1 <- prcomp(set1)
      pca.set2 <- prcomp(set2)

      # do CCA on PC scores
      cc.res <- cancor(pca.set1$x[, 1:npcs], pca.set2$x[, 1:npcs])

      # do LRT w/ bartlett correction for CCA
      n <- nrow(set1)
      cc.rho2 <- rev(cc.res$cor^2)
```

```
    test.stat ← (-1)*(n - 1 - .5 * (npcs + npcs + 1)) * log(cumprod(1 - cc.rho2)
        )
    df ← (npcs - length(cc.rho2):1 + 1 ) * (npcs - length(cc.rho2):1 + 1 )
    p.value ← (1 - pchisq(test.stat, df))

    test.stat ← rev(test.stat)
    df ← rev(df)
    p.value ← rev(p.value)

    n.ccs ← npcs

    # compute canonical covariate scores and loadings
        ----------------------------------------

    set1.scores ← pca.set1$x[, 1:npcs, drop=FALSE] %*% cc.res$xcoef[, 1:n.ccs,
        drop=FALSE]
    set2.scores ← pca.set2$x[, 1:npcs, drop=FALSE] %*% cc.res$ycoef[, 1:n.ccs,
        drop=FALSE]
    set1.loads ← cor(set1, set1.scores)
    set2.loads ← cor(set2, set2.scores)

    # redundancy index
    dat2cc.set1 ← colSums((colVars(set1) * set1.loads^2)/sum(colVars(set1)))
    dat2cc.set2 ← colSums((colVars(set2) * set2.loads^2)/sum(colVars(set2)))
    set1.redundancy ← dat2cc.set1*(cc.res$cor^2)[1:n.ccs]
    set2.redundancy ← dat2cc.set2*(cc.res$cor^2)[1:n.ccs]

    # consolidate results into a list ----------------------------------------
    output ← list()
    output$test.results ← cbind(chisq_stat=test.stat,
                                df=df,
                                p_value=p.value,
                                set1_r2=set1.redundancy,
                                set2_r2=set2.redundancy)

    output$loadings ← list()
    output$loadings$set1 ← set1.loads
    output$loadings$set2 ← set2.loads

    output$scores ← list()
    output$scores$set1 ← set1.scores
    output$scores$set2 ← set2.scores

    out.return[[gene]] ← output
  }
  ind ← ind + 1
  cat('\n')
}

# consolidate results  ----------------------------------------
out.final ← list()

# testing results
out.final$testing.results ← lapply(out.return, function(x){
  if(!is.na(x)){
    x$test.results
  } else {
    rep(NA, 5)
  }
})

# set 1 cc scores
out.final$set1.scores ← lapply(out.return, function(x, n.c){
  if(!is.na(x)){
```

```
      x$scores$set1
    } else {
      rep(NA, n.c)
    }
  }, n.c=dim(full.set)[2])

  # set 2 cc scores
  out.final$set2.scores <- lapply(out.return, function(x, n.c){
    if(!is.na(x)){
      x$scores$set2
    } else {
      rep(NA, n.c)
    }
  }, n.c=dim(full.set)[2])

  # set1 loadings
  out.final$set1.loadings <- lapply(out.return, function(x){
    if(!is.na(x)){
      x$loadings$set1
    } else {
      NA
    }
  })

  # set 2 loadings
  out.final$set2.loadings <- lapply(out.return, function(x){
    if(!is.na(x)){
      x$loadings$set2
    } else {
      NA
    }
  })
  out.final

}

na2mean <- function(x){
  x[is.na(x)] <- mean(na.omit(x))
  return(x)
}

pca <- function(x) prcomp(t(x[, -ncol(x)]))
```

```
permTest ← function(object, cca.results = NULL, n.perm = 1000, half.life = 400){

  if (!is(object, 'GDIset')) stop("object must be a 'GDIset'")

  if (!is.null(cca.results)){
    cat('Performing permutation test on communalities \n')
    incl.ind ← which(unlist(lapply(cca.res$testing.results, function(x) !is.na(x
        [1]))))
    unique.ids ← names(cca.results$testing.results)[incl.ind]
  }else{
    cat('Performing permutation test on R-squared values \n')
    set1.names ← names(table(getAnnot(object)[[1]]$entrez.id) > 3)
    set2.names ← names(table(getAnnot(object)[[2]]$entrez.id) > 3)
    unique.ids ← intersect(getAnnot(object)[[1]]$entrez.id,
                           getAnnot(object)[[2]]$entrez.id)
  }

  out ← foreach(gene=unique.ids, .packages='gdi', .combine='c') %dopar% {

    # get locations of sites
    set1.ind ← which(object@set1@annot$entrez.id %in% gene)
    set1.loc ← (end(object@set1@annot[set1.ind]) + start(object@set1@annot[set1.
        ind]))/2
    names(set1.loc) ← names(object@set1@annot[set1.ind])

    set2.ind ← which(object@set2@annot$entrez.id %in% gene)
    set2.loc ← (end(object@set2@annot[set2.ind]) + start(object@set2@annot[set2.
        ind]))/2
    names(set2.loc) ← names(object@set2@annot[set2.ind])

    if (!is.null(cca.results)){
      # save communalities with short variable names
      set1.comm ← cca.results$set1.loadings[[gene]][, 1, drop=FALSE]^2
      set2.comm ← cca.results$set2.loadings[[gene]][, 1, drop=FALSE]^2

      # get outer product of communalities
      comm.outer ← set1.comm %*% t(set2.comm)

    }else{
      # compute R^2 matrix from actual data
      comm.outer ← cor(t(object@set1@dat[set1.ind, ]),
                       t(object@set2@dat[set2.ind, ]))^2
    }

    # get distance matrix
    dist.mat ← matrix(set1.loc, nr = length(set1.loc),
                      nc=length(set2.loc))

    dist.mat ← abs(sweep(dist.mat, 2, set2.loc, '-'))

    # apply exponential decay function to get weights
    lambda ← log(2)/half.life
    weight.mat ← exp(-dist.mat*lambda)
    obs.stat ← sum(comm.outer * weight.mat)

    # permute weights and compute stats
    perm.stats ← numeric(n.perm)
    for(jj in 1:n.perm){
      perm.stats[jj] ← sum(comm.outer * weight.mat[sample(nrow(weight.mat)), ][ ,
          sample(ncol(weight.mat))])
    }

    perm.pval ← sum(perm.stats > obs.stat)/n.perm
```

```
    perm.pval
  }

  names(out) <- unique.ids
  out

}
```

Appendix D

CODE FOR CHAPTER 3 SIMULATION STUDIES

```
library(foreach)
library(doMC)

# Simulation to test for type I error ------------------------------------
# set simulation parameters ----------------------------------------------
n      <- c(26, 50, 100, 200, 500)      # number of samples with matched data
loci1  <- 30                            # number of CpG sites
loci2  <- 8                             # number of exons
rho1   <- .25                           # null CpG correlation for compound
    symmetry covariance
rho2   <- -.0678                        # null exon correlation for compound
    symmetry covariance
slope1 <- .00313/(loci1-1)              # slope to span range of CpG variances
intercept1 <- .000437                   # intercept to span range of CpG variances
slope2 <- .158/(loci2-1)                # slope to span range of exon variances
intercept2 <- .086                      # intercept to span range of exon variances
npcs <- c(1, 3, 5, 10, 15)              # number of PCs to keep after PCA for CCA
    step
n.sims <- 1e5                           # number of sims for each param combo

n.cores <- floor(2*detectCores()/3)
registerDoMC(n.cores)

results <- foreach(n.ii = n, .packages='MASS') %dopar% {
  cat('started ', n.ii, '...\n')
  compute.typeI <- function(x, n, n.sim){
    mean1 <- numeric(x[1])
    mean2 <- numeric(x[2])

    sigma1 <- matrix(x[3], nr = x[1], nc = x[1])
    sigma2 <- matrix(x[4], nr = x[2], nc = x[2])
    diag(sigma1) <- (1:nrow(sigma1)-1) * x[5] + x[7]
    diag(sigma2) <- (1:nrow(sigma2)-1) * x[6] + x[8]

    for(ii in 1:nrow(sigma1)){
      for(jj in 1:nrow(sigma1)){
        if(ii != jj) sigma1[ii, jj] <- sigma1[ii, jj]*sqrt(sigma1[ii, ii])*sqrt(
            sigma1[jj, jj])
      }
    }

    for(ii in 1:nrow(sigma2)){
      for(jj in 1:nrow(sigma2)){
        if(ii != jj) sigma2[ii, jj] <- sigma2[ii, jj]*sqrt(sigma2[ii, ii])*sqrt(
            sigma2[jj, jj])
      }
    }

    typeI <- numeric(n.sims)
    for(ii in 1:n.sims){
      set1 <- mvrnorm(n=n, mu=mean1, Sigma=sigma1)
      set2 <- mvrnorm(n=n, mu=mean2, Sigma=sigma2)

      scores1 <- prcomp(set1)$x[, 1:min(x[9], ncol(set1))]
```

```
        scores2 <- prcomp(set2)$x[, 1:min(x[9], ncol(set2))]

        cc.res <- cancor(scores1, scores2)

        min.set1 <- nrow(cc.res$xcoef)
        min.set2 <- nrow(cc.res$ycoef)

        cc.rho2 <- rev(cc.res$cor^2)
        test.stat <- (-1)*(n - 1 - .5 * (min.set1 + min.set2 + 1)) * log(cumprod(1 -
            cc.rho2))
        df <- (min.set1 - length(cc.rho2):1 + 1 ) * (min.set2 - length(cc.rho2):1 + 1
            )
        p.value <- (1 - pchisq(test.stat, df))
        typeI[ii] <- tail(p.value, 1)
      }
      sum(typeI < .05)/n.sims
    }

    result.mat <- as.matrix(expand.grid(loci1, loci2, rho1, rho2, slope1, slope2,
        intercept1, intercept2, npcs))

    colnames(result.mat) <- c('loci1', 'loci2', 'rho1', 'rho2', 'slope1', 'slope2', '
        intercept1', 'intercept2', 'npcs')

    typeI.error <- apply(result.mat, 1, compute.typeI, n = n.ii, n.sim = n.sims)
    result.mat <- cbind(result.mat, typeI.error)
    cat('finished ', n.ii, '\n')
    result.mat
}

names(results) <- n

results.df <- NULL
for(ii in 1:length(n)){

    results.df <- rbind(results.df,
                        cbind(rep(n[ii], nrow(results[[ii]]))
                              , results[[ii]]))
}



setwd('/home/manserpt/gdi_ch3/data')
save(results.df, file='cca-test-typeI-results-realistic.rda')
```

```
library(foreach)
library(doMC)

# Simulation to test for power ------------------------------------------------
# set simulation parameters ---------------------------------------------------
n       <- c(26, 50, 100, 200)          # number of samples with matched data
loci1   <- 20                           # number of CpG sites
loci2   <- 8                            # number of exons
rho1    <- .25                          # null CpG correlation for compound
    symmetry covariance
rho2    <- -.0678                       # null exon correlation for compound
    symmetry covariance
slope1 <- 0                             # slope to span range of CpG variances
intercept1 <- .00123                    # intercept to span range of CpG variances
slope2 <- 0                             # slope to span range of exon variances
intercept2 <- 0.15                      # intercept to span range of exon variances
npcs <- c(1, 3, 5)                      # number of PCs to keep after PCA for CCA
    step
methy.change <- .2                      # mean difference between case and control
n.cpgs <- c(1, 3, 5)                    # how many CpGs change?
splice.change <-  1.2                   # mean difference between case and control
n.exons <- c(1, 4)                      # how many exons change?
n.sims <- 1e5                           # number of sims for each param combo

n.cores <- floor(2*detectCores()/3)
registerDoMC(n.cores)

results <- foreach(n.ii = n, .packages='MASS') %dopar% {
  cat('started ', n.ii, '...\n')
  compute.typeI <- function(x, n, n.sim){
    mean1 <- numeric(x[1])
    mean2 <- numeric(x[2])

    sigma1 <- matrix(x[3], nr = x[1], nc = x[1])
    sigma2 <- matrix(x[4], nr = x[2], nc = x[2])
    diag(sigma1) <- (1:nrow(sigma1)-1) * x[5] + x[7]
    diag(sigma2) <- (1:nrow(sigma2)-1) * x[6] + x[8]

    for(ii in 1:nrow(sigma1)){
      for(jj in 1:nrow(sigma1)){
        if(ii != jj) sigma1[ii, jj] <- sigma1[ii, jj]*sqrt(sigma1[ii, ii])*sqrt(
            sigma1[jj, jj])
      }
    }

    for(ii in 1:nrow(sigma2)){
      for(jj in 1:nrow(sigma2)){
        if(ii != jj) sigma2[ii, jj] <- sigma2[ii, jj]*sqrt(sigma2[ii, ii])*sqrt(
            sigma2[jj, jj])
      }
    }

    typeI <- numeric(n.sims)
    for(ii in 1:n.sims){
      set1 <- mvrnorm(n=n, mu=mean1, Sigma=sigma1)
      set2 <- mvrnorm(n=n, mu=mean2, Sigma=sigma2)

      set1[1:(ncol(set1)/2), 1:x[11]] <- set1[1:(ncol(set1)/2), 1:x[11]] + x[10]
      set2[1:(ncol(set2)/2), 1:x[13]] <- set2[1:(ncol(set2)/2), 1:x[13]] + x[12]

      scores1 <- prcomp(set1)$x[, 1:min(x[9], ncol(set1))]
      scores2 <- prcomp(set2)$x[, 1:min(x[9], ncol(set2))]

      cc.res <- cancor(scores1, scores2)
```

```
        min.set1 <- nrow(cc.res$xcoef)
        min.set2 <- nrow(cc.res$ycoef)

        cc.rho2 <- rev(cc.res$cor^2)
        test.stat <- (-1)*(n - 1 - .5 * (min.set1 + min.set2 + 1)) * log(cumprod(1 -
            cc.rho2))
        df <- (min.set1 - length(cc.rho2):1 + 1 ) * (min.set2 - length(cc.rho2):1 + 1
            )
        p.value <- (1 - pchisq(test.stat, df))
        typeI[ii] <- tail(p.value, 1)
      }
      sum(typeI < .05)/n.sims
    }

    result.mat <- as.matrix(expand.grid(loci1, loci2, rho1, rho2,
                                        slope1, slope2, intercept1, intercept2, npcs
                                            ,
                                        methy.change, n.cpgs, splice.change, n.exons
                                            ))

    colnames(result.mat) <- c('loci1', 'loci2', 'rho1', 'rho2', 'slope1', 'slope2',
                              'intercept1', 'intercept2', 'npcs', 'methy.change',
                              'n.cpgs', 'splice.change', 'n.exons')

    typeI.error <- apply(result.mat, 1, compute.typeI, n = n.ii, n.sim = n.sims)
    result.mat <- cbind(result.mat, typeI.error)
    cat('finished ', n.ii, '\n')
    result.mat
}

names(results) <- n

results.df <- NULL
for(ii in 1:length(n)){

    results.df <- rbind(results.df,
                        cbind(rep(n[ii], nrow(results[[ii]]))
                              , results[[ii]]))
}

results.df[, c(1, 10, 11:14, 15 )]

(results.df[,c(1, 10, 12, 14, 15)])


setwd('/home/manserpt/gdi_ch3/data')
save(results.df, file='cca-test-power-results-realistic-alt-prom.rda')
```