A MODIFIED BUMP HUNTING APPROACH WITH CORRELATION-ADJUSTED KERNEL WEIGHT FOR DETECTING DIFFERENTIALLY METHYLATED REGIONS ON THE 450K ARRAY

By

Jeannie Daniel

Submitted to the Faculty of the Graduate School
of Augusta University in partial fulfillment
of the Requirements for the Degree of
Doctor of Philosophy

July

2017

ACKNOWLEDGEMENTS

I would like to thank my committee members and reader, Drs. Hongyan Xu, Xiaoling Wang, Santu Ghosh, Daniel Linder, and Arni SR Srinivasa Rao for all of their comments and suggestions during this process. Thank you for challenging me and helping me refine my ideas.

I would also like to send a special thank you to Dr. Stephen Looney, Dr. Jennifer Waller, and Ms. Pat Hall for being supportive resources for me over these years. Thank you for your mentorship and for helping me learn how to be a biostatistician.

I also want to thank my classmates, Jaeeun Lee and Taejin Lee. Thank you for our discussions of homework problems, programming issues, and for being my friends over these past 5 years. This is a time in life we will never forget.

However most of all I would like to thank Dr. Jie Chen. Thank you for introducing me to DNA methylation and being my dissertation advisor, but most of all thank you for your guidance, direction, and for helping me learn to think like a researcher.

This dissertation is dedicated to my parents, for their never-ending support of me and my dreams.

ABSTRACT

JEANNIE DANIEL

A modified bump hunting approach with correlation-adjusted kernel weight for detecting differentially methylated regions on the 450K array

(Under the direction of JIE CHEN, PHD, MAJOR ADVISOR)

DNA methylation plays an important role in the regulation of gene expression, as hypermethylation is associated with gene silencing. The general purpose of this dissertation is the development of a statistical method, called DMR Detector, for detecting differentially methylated regions (DMRs) on the 450K array. DMR Detector makes three key modifications to an existing method called Bumphunter. The first is what statistic to collect from the initial fitting for further analysis. The second is to perform kernel smoothing under the assumption of correlated errors using a newly proposed correlation-adjusted kernel weight. The third is how to define regions of interest. In simulation, the method was shown to have high power comparable to Bumphunter, with consistently lower family-wise type I error rate, controlled well below the 0.1 FDR. DMR Detector was applied to real data and was able to detect one DMR that was not detected by Bumphunter.

KEY WORDS: Epigenetics, DNA methylation, differentially methylated regions

Table of Contents

List of Tables

List of Figures

1. INTRODUCTION

1.1 Statement of the Problem

DNA methylation plays an important role in the regulation of gene expression. It is a chemical modification of DNA that can be passed down from generation to generation during cell division, however it is not contained in the DNA sequence itself [Jaffe et al., 2012]. It involves the modification of a cytosine base (C) to form methyl-cytosine, and occurs almost exclusively at C's that are immediately followed by a guanine (G) in the $5'$ to $3'$ direction. These locations are called CpG sites. Both hyper- and hypo-methylation of DNA and their relationship with gene expression have been implicated in many diseases. Hypermethylation is associated with gene silencing [Sofer et al., 2013]. For example, hypermethylation of CpG islands located in the promoter regions of tumor suppressor genes has been established as a major mechanism of gene regulation in cancer [Du et al., 2010, Esteller, 2002, Herman and Baylin, 2003].

1.1.1 Differential methylation analysis

Investigators often want to know if DNA methylation is associated with a certain disease, and where on the genome this occurs. Differential methylation analysis is the

term used for this type of analysis [Du et al., 2010]. However functionally relevant findings in differential methylation analysis have generally been associated with genomic regions as opposed to single CpG sites. Therefore the method proposed in this dissertation is a statistical method for detecting differentially methylated regions (DMRs).

1.1.2 Description of the Data

Infinium HumanMethylation450 (450K) is a preferred technology for studying DNA methylation in large-scale studies [Rakyan et al., 2011, Dedeurwaerder et al., 2011]. The microarray makes it possible to evaluate the methylation status of more than 450,000 CpGs located throughout the genome [Bibikova et al., 2011]. There are two different types of chemical assays used in this technology, Infinium I and Infinium II [Dedeurwaerder et al., 2014]. Both types are based on a quantitative genotyping of the C/T polymorphism generated by bisulfite conversion, however the Infinium I assay resembles a single-channel microarray, whereas the Infinium II assay has a dual-color readout. The Infinium I assay uses two types of probes, one for the methylated allele and one for the unmethylated allele, with the same base extension for both alleles. The Infinium II assay uses a single probe for both alleles, and base extension depends on the methylation state of the hybridized genomic DNA molecule.

1.2 Measuring the methylation level

Two methods have been proposed to measure the methylation level at each CpG site on the 450K array [Du et al., 2010]. The first method calculates what is called a Beta-

value, which is a value ranging from 0 to 1, representing the approximate proportion of methylation at each CpG site. It can be defined as the ratio of the methylated probe intensity to the overall intensity, which is the sum of the methylated and unmethylated probe intensities. Let

$$Beta_i = \frac{max(y_{i,methy}, 0)}{max(y_{i,methy}, 0) + max(y_{i,unmethy}, 0) + \alpha}$$

where $y_{i,methy}$ and $y_{i,unmethy}$ are the intensities measured by the ith methylated and unmethylated probes, respectively [Bibikova et al., 2006]. Illumina recommends adding an offset α to the denominator to regularize Beta-values when both methylated and unmethylated probe intensities are low. By default, this value is set at $\alpha = 100$.

The second method used to measure the methylation level at each CpG site calculates what is called an M-value, which is the log2 of the ratio of the methylated probe intensity to the unmethylated probe intensity. Let

$$M_i = log_2 \left[\frac{max(y_{i,methy}, 0) + \alpha}{max(y_{i,unmethy}, 0) + \alpha} \right]$$

Du et al. (2010) suggest adding a constant α to the intensity values, equal to 1 by default, to prevent unexpected large changes due to small intensity estimation errors [Du et al., 2010]. This is because small changes in the methylated and unmethylated probe intensities can result in large changes in the M-value when the intensities are very low, especially when they are between 0 and 1. Positive M-values indicate that more molecules are methylated than unmethylated, and negative M-values indicate the opposite.

Du et al. (2010) argue that the offsets have little effect on the Beta-value or M-value for

most interrogated CpG sites [Du et al., 2010]. As a result, a logistic relationship between the two can be derived, with the offsets ignored, by

$$Beta_i = \frac{2^{M_i}}{2^{M_i}+1}, \text{where } M_i = log_2\left[\frac{Beta_i}{1-Beta_i}\right] \tag{1.1}$$

Du et al. (2010) recommend using M-values instead of Beta-values in most analyses because Beta-values have significant heteroscedasticity in both the low and high methylation range [Du et al., 2010]. This problem is alleviated by transforming Beta-values to M-values using the relationship in equation (1.1). The resulting M-values are approximately homoscedastic, displaying a distribution with approximately constant variance.

1.3 Preprocessing the data

The design of the 450K array has resulted in important consequences on the generated data, resulting in a need to preprocess the data before it can be used [Dedeurwaerder et al., 2011]. There are several steps that must be taken to preprocess the data [Dedeurwaerder et al., 2014]. This preprocessing includes filtering out problematic probes and normalization.

1.3.1 Filtering out the problematic probes

The first step when performing microarray data preprocessing is to filter out probes that can generate artifactual data [Dedeurwaerder et al., 2014]. This is a problem common to all microarray platforms. Dedeurwaerder et al. (2014) suggest filtering out these probes as the

first step in preprocessing the data. Other scientists would perform this step at the end, after the normalization step, to enable researchers to look at a different probe set than the one initially selected without having to repeat the normalization step. However, Dedeurwaerder et al. (2014) argue that the possibility that these problematic probes influence normalization cannot be excluded, and suggest filtering out values from probes that do not appear reliable as the first step in data preprocessing.

Probes with high detection p-values

One way probes can generate artifactual data is that the scanner may not correctly read the signal for some probes with low intensities or spatial artifacts on the array [Dedeurwaerder et al., 2014]. This problem results in a low quality signal, resulting in what is termed a high detection p-value, for the affected probes. As a result, the authors strongly recommend filtering out probes with a high detection p-value, such as those with $p > 0.05$, before performing downstream analyses.

Cross-reactive probes

Cross-reactive probes are another problem that can generate artifactual data [Dedeurwaerder et al., 2014]. The genome is comprised of sequences of 4 letters, (A,T,G,C). Infinium HumanMethylation450 technology uses bisulfite treatment to convert unmethylated cytosines to uracils, which generates a C/T polymorphism at CpG sites after DNA amplification [Dedeurwaerder et al., 2014, Bibikova et al., 2009]. A consequence of this is the generation of an almost 3-letter genome, (A,T,G), with the only remaining Cs being unmethylated [Dedeurwaerder et al., 2014]. These remaining C's represent about

3.5 percent of the total number of Cs. This considerably increases the probability of probe cross-reactivity, which is the probability that some of the probes will co-hybridize at additional locations on the genome that are different from the regions for which the probes were intended. Between 8.6 and 25 percent of the 450K probes have been identified as being cross-reactive, depending on the criteria used [Price et al., 2013, Zhang et al., 2012].

This is a problem because a DNA methylation measurement from a cross-reactive probe is likely to represent a combination of the methylation levels of multiple genomic sites instead of the methylation level at only the targeted CpG site [Dedeurwaerder et al., 2014]. As a result, artifactual methylation measurements are generated, leading to the detection of artifactual differentially methylated sites. For example, many sex-associated differences in methylation are reported to be technical artifacts created by the cross-reactivity between autosomal probes and genomic regions on the sex chromosomes [Chen et al., 2013]. Dedeurwaerder et al. (2014) therefore recommend either disregarding these probes, and/or using an approach such as bisulfite pyrosequencing (BPS) to confirm the methylation measurements by another method [Dedeurwaerder et al., 2014].

Probes containing common SNPs

Another problem with the bisulfite treatment is that it can also detect C/T polymorphisms that are naturally present at the particular CpG site [Price et al., 2013, Chen et al., 2013]. Therefore, the DNA methylation measurements can be confounded by the actual DNA sequence itself [Dedeurwaerder et al., 2014]. For example, in the case of a fully methylated CpG site, methylation measurements from samples with the C/C genotype are close to 100 percent, while those from samples with the T/T genotype are

close to 0 percent. This is the expected behavior. However, if the sample is heterozygous, the methylation measurement will be close to 50 percent. As a result, in the case of probes containing SNPs at the targeted CpG site, the measurements are more likely to reflect the genotype instead of the methylation value.

Approximately 4.3 percent of the 450K probes contain a known polymorphism at the targeted C or G [Price et al., 2013]. In intra-individual studies, such as longitudinal studies or ones involving monozygotic twins, for example, this issue should not be an important confounder [Dedeurwaerder et al., 2014]. However in inter-individual studies, such as case-control studies, it could be a problem, depending on the frequency of heterozygosity. Although 56.8 percent of these probes display infrequent SNPs, 43.2 percent have a polymorphism that is more frequent in the population. This makes these probes more likely to confound the methylation measurements. SNPs can also be present within the remainder of the probe, although it seems that methylation measurements are not significantly affected by the presence of such SNPs [Price et al., 2013]. Therefore, Dedeurwaerder et al. (2014) recommend filtering out probes containing a frequent SNP at the targeted CpG site and/or performing SNP genotyping along with the methylation experiment in interindividual studies, but suggest that this is probably not necessary in intra-individual studies [Dedeurwaerder et al., 2014].

Other problematic probes

The last problem that can generate artifactual data on the 450K array relates to the relationship between signal intensities and methylation measurements [Dedeurwaerder et al., 2014]. The issue is that probes displaying a high average

intensity are more likely than those with lower average intensities to provide methylation measurements that are inconsistent with measurements obtained from other approaches, such as BPS. A high average intensity is defined as a high average of the methylated and unmethylated signals. These probes have a tendency to provide Beta-values close to 0.5, regardless of their true methylation status. Dedeurwaerder et al. (2014) note that Infinium II probes appear less likely to be prone to this problem than Infinium I probes, but recommend caution with any probe that displays a high average intensity. They suggest filtering out probes with a high average intensity before performing downstream analysis, or checking measurements by another method for confirmation.

1.3.2 Normalization

The next step of microarray preprocessing is to remove any source of variation that is not related to biology, but rather to technical issues [Dedeurwaerder et al., 2014]. This step is called normalization. There are specific normalization methods required for the 450K array.

Within-array normalization

Within-array normalization attempts to correct three main issues: background correction, color bias (or dye bias) adjustment, and Infinium I/II-type bias correction [Dedeurwaerder et al., 2014]. Because the Infinium II assay uses the same bead to measure both methylated and unmethylated signals, the measurement of one of these is affected by the residual emission of the other. This results in a higher background signal for Infinium II probes than for Infinium I probes, and contributes to a reduction in the range of Beta-values

for Infinium II probes.

Color bias is related to the difference in measurement accuracy between the two dyes, red or green [Dedeurwaerder et al., 2014]. Fortunately, color bias has very little impact on the Beta-values from the Infinium I probes, since the methylated and unmethylated states of each CpG are evaluated in the same color channel. However, there is a difference between the Beta-value range for these probes when using the red or green channel that may be due to the different backgrounds of the two color channels. For the Infinium II assay, the color bias is more problematic. This is because the methylated and unmethylated status of each CpG site are evaluated in different channels. This skews the Beta-values from Infinium II probes, and contributes to a reduction in their range as well.

Infinium I/II-type bias is related to the different design of the Infinium I probe versus the Infinium II probe [Dedeurwaerder et al., 2014]. Many methods used to address this bias involve some sort of rescaling in an attempt to match different features of the distribution of the Infinium II probe to the distribution of the Infinium I probe. This type of bias is a combination of the other two types of bias, so a correction of it will also address the color bias and background in a manner similar to a color bias correction combined with a background correction used independently.

Between-array normalization

Between-array normalization methods have been developed to address other sources of non-biological variation. These sources of variation can be due to unequal quantities of starting material, or differences in labeling or detection efficiencies [Dedeurwaerder et al., 2014]. Several methods attempt to reduce these array-to-array

variations by adjusting measurements at a global level. However Dedeurwaerder et al. (2014) argue that there is no between-array normalization method for 450K data that can improve the data enough to offset the degradation of data quality these other sources of variation can sometimes cause [Dedeurwaerder et al., 2014].

A serious problem related to between-array normalization are 'batch' and 'slide' effects [Dedeurwaerder et al., 2014]. Batch effects are non-biological variations that exist between batches of samples that were processed, for example, on different days, on different scanners, or by different experimenters. Slide effects are non-biological variations related to the position of the array on the slide, and the position of the slides within the same batch of samples. Batch and slide effects can generate artifactual measurements at the global level.

Because batch effects can affect only a subset of probes, they cannot be eliminated by global normalization methods [Dedeurwaerder et al., 2014]. As a result, there are some local methods aimed at removing these types of effects. However, Dedeurwaerder et al. (2014) suggest that the best way to control batch and slide effects is to have a good design of the experiment, including ensuring a good distribution of the samples on the slides, and processing all samples on the same day, by the same experimenter, on the same scanner.

2. LITERATURE REVIEW

After the data has been appropriately preprocessed, the search for differentially methylated regions (DMRs) can begin. There are several DMR finding algorithms available for the 450K array.

2.1 IMA

Wang et al. (2012) proposed a pipeline, or workflow, for region-level methylation analysis of 450K data in their R/Bioconductor package, IMA [Wang et al., 2012]. The IMA package can preprocess the raw data with quantile normalization, and allows users to choose several filtering options. The Beta-values are used for the analysis by default, but an arcsine square root transformation or a logit transformation of the Beta-values are also available.

Rocke (1993) explains that random variables on the interval [0,1], like the Beta-value, can cause problems in analysis because the distribution is skewed and the variance is related to the mean [Rocke, 1993]. As a result, normal approximations can yield unacceptable results. One way to deal with these problems is to apply a variance-stabilizing transformation. The authors suggests that common choices for the transformation are

the arcsin square root, which asymptotically stabilizes the variance of a binomial random variable, the logit, the probit, and the complementary log-log.

IMA uses predefined regions as candidates for DMRs, such as the first exon for example. For each predefined region, IMA will collect the loci within it, and calculate an index of overall region methylation. Choices for the index value include the mean, median, or Tukeys biweight robust average. For each specific region, inference can be made for differential methylation between groups using Wilcoxon rank-sum test (default), Student's t-test (pooled or Satterthwaite) or empirical Bayes approaches. Generalized linear models are also provided as an option to adjust for additional covariates. The arcsine square root transformation on the methylation value could be used in a linear regression model for example [Rocke, 1993, Marsit et al., 2011], or the logit transformation could be used in a logistic regression model [Kuan et al., 2010].

The IMA package also has several options for adjusting for multiple testing [Wang et al., 2012]. The authors note that it is not the purpose of the IMA package to determine which method of analysis or adjusting for multiple testing is best, but rather to provide users a variety of choices in their analysis.

2.2 COHCAP

Warden et al. (2013) proposed a pipeline for region-level methylation analysis on 450K data in their R/Bioconductor package, COHCAP [Warden et al., 2013]. The package also has the ability to identify regions that are most likely to regulate downstream gene expression. The package does not provide any normalization methods, but can accept data

that have already been normalized by other methods.

The CpG site analysis in the COHCAP package is based on the method described by Sproul et al. (2011) [Sproul et al., 2011, Warden et al., 2013]. In this method, sites are defined as methylated if the Beta-value is greater than a certain threshold, which is 0.7 for cell line data and 0.3 for patient data [Sproul et al., 2011]. Sites are unmethylated if they have Beta-values less than 0.3. Ambiguous sites are filtered out. The COHCAP package offers two different workflows for methylation analysis, 'Average by Site' and 'Average by Island' [Warden et al., 2013].

The 'Average by Site' workflow first calculates the average methylation values for each group for each CpG site, then tests for differential methylation [Warden et al., 2013]. If there are more than two groups, p-values are calculated via the ANOVA F-statistic. Otherwise a t-test is used. False discovery rate (FDR) values are calculated using the method of Benjamini and Hochberg (1995) [Benjamini and Hochberg, 1995]. CpG sites are then filtered based on average Beta-values (above cutoff in one group and below cutoff in the other group), p-value, and FDR.

The COHCAP package uses predefined regions as candidates for DMRs just as IMA does. In COHCAP, the predefined regions are CpG islands [Warden et al., 2013]. These predefined CpG islands are retained for statistical analysis if they possess a minimum number of filtered CpG sites. The default value is 4. DNA methylation and gene expression data are then integrated by filtering for islands or genes that meet various criteria. Users can filter based on a methylated or unmethylated threshold, CpG island p-value, CpG island FDR, expression fold-change, expression p-value, and/or expression FDR. The final result is a list of genes or islands with an inverse relationship between methylation and gene

expression, since hypermethylation is associated with gene silencing [Sofer et al., 2013].

The 'Average by Island' workflow begins in the same way as the 'Average by Site' workflow, by first calculating the average methylation values for each group for each CpG site, and then testing for differential methylation using either ANOVA or the t-test [Warden et al., 2013]. CpG sites are then filtered based on average Beta-values (above cutoff in one group and below cutoff in the other group), p-value, and FDR. If a CpG island, as defined by the annotation file, contains at least 4 CpG sites then average Beta-values are calculated for filtered CpG sites in each CpG island for each group. CpG island Beta-values are then treated like CpG site Beta-values for statistical analysis.

2.3 FastDMA

Wu et al. (2014) developed another method for differential methylation analysis called FastDMA [Wu et al., 2013]. FastDMA can also perform region-based analysis to detect DMRs. It is implemented as a stand alone software in C++. Analysis of covariance (ANCOVA) is the method used to perform all analyses in the package. FastDMA can include testing with multiple groups, and allows covariates to be included in the model.

FastDMA can use predefined regions as candidates for DMRs, just as IMA and COHCAP can [Wu et al., 2013]. In FastDMA, these predefined regions could be a promoter region or a CpG island, for example. For each region that contains several probes, if the region is uniformly methylated among all groups of interest, it is assumed that the Beta-values of every probe in the region are distributed with the same mean. Otherwise the region is considered to be differentially methylated.

For each region, ANCOVA is used to compare two linear regression models [Wu et al., 2013]. One model assumes an overall mean across all of the groups, and the other assumes different means. A p-value is then calculated to determine which model is better. Adjustment for multiple testing is performed using the method of Benjamini and Hochberg [Benjamini and Hochberg, 1995].

However FastDMA can also determine regions of interest by scanning the entire genome, instead of simply using predefined regions [Wu et al., 2013]. One way this is done is by using a sliding window, then testing whether each window is a DMR using the method described above. Overlapping DMRs are merged together.

There is a faster option as well [Wu et al., 2013]. It uses the Benjamini-Hochberg false discovery rates that are calculated from the single probe analysis option. It then calculates the geometric mean of the FDRs as the score for that region. Windows with a score less than 0.05 are considered to be DMRs. Overlapping DMRs are merged together as before.

2.4 AClust

Sofer et. al (2013) propose another method to DMRs on the 450K array in their R/Bioconductor package, AClust [Sofer et al., 2013]. They call their method Adjacent Site Clustering, or A-clustering, and explain that it defines regions by clustering together neighboring CpG sites according to the correlation of the methylation values at each site. It can also include possible restrictions on the distances between sites.

AClust first performs clustering of CpG sites, and then tests the effect of an environmental exposure on each cluster [Sofer et al., 2013]. For testing, it is assumed that

the exposure affects all CpG sites equally, while each site has its own baseline methylation level. This analysis is performed using generalized estimating equations (GEEs).

The first step is the clustering of adjacent, correlated CpG sites [Sofer et al., 2013]. Users specify the minimum number of CpGs to include in a cluster, and can also specify a maximum genomic distance for clustering that prevents clustering of CpG sites that are further apart than that distance without any other CpGs in between them.

The effect of the exposure on the clusters is then tested [Sofer et al., 2013]. A GEE model is fit assuming common exposure and covariate effects on all CpGs within a cluster, and an individual location effect for each CpG site. The unadjusted p-value for the exposure is calculated based on the GEE model. Finally there is an adjustment for multiple testing using the method of Benjamini and Hochberg [Benjamini and Hochberg, 1995].

2.5 Comb-p

Pedersen et al. (2012) developed a 'moving averages' method of p-value correction, called comb-p, that can also be used to detect DMRs [Pedersen et al., 2012]. The comb-p method does not depend on the test used to generate the p-values, and can therefore be used for a wide variety of applications, including differential methylation analysis. The comb-p method is a command-line tool and a Python library that manipulates BED files of possibly irregularly spaced p-values, such as p-values that are calculated from tests at each CpG site on the genome, for example.

The method is an extension of a method developed by Kechris et al. (2010), which combines p-values in sliding windows and accounts for spatial correlations across the

genome [Kechris et al., 2010]. The comb-p method additionally allows for uneven data structure across the genome, general auto-correlation calculations, and multiple-testing corrections for genomic regions of interest [Pedersen et al., 2012]. The comb-p method first estimates the auto-correlation, combines adjacent p-values, adjusts for false discovery rate control, defines regions of enrichment by collecting series of adjacent low p-values, and, finally, assigns significance to those regions.

The authors cite Fisher [Fisher, 1948], who developed an approach for combining p-values from independent tests to obtain a single meta-analysis test statistic [Pedersen et al., 2012]. This test statistic has a X^2 distribution, with degrees of freedom based on the number of tests combined. A similar method developed by Stouffer et al. (1949) and Liptak (1958) first converts p-values to Z-scores, and then sums and scales them to create a combined Z-score [Samuel A. Stouffer, 1949, Liptak,].

The authors point out that the StoufferLiptak method lends itself to the incorporation of weights on each p-value [Pedersen et al., 2012], and Zaykin et al. (2002) introduced such a method [Zaykin et al., 2002]. This method uses weights to perform a dependence correction on possibly correlated tests. Kechris et al. (2010) used a sliding window correction where each p-value is adjusted by applying the Stouffer-Liptak method to neighboring p-values that are weighted using the observed auto-correlation at the appropriate lag [Kechris et al., 2010]. The comb-p method provides a generic, efficient, and customizable implementation of these methods, and also includes handling of variably spaced probes, a peak finder for dynamically sized regions, and a method to calculate a p-value for each peak. Adjustment for multiple testing is also included.

The comb-p approach accepts p-values from any software or statistical test

[Pedersen et al., 2012]. It first estimates correlations at varying distance lags, and obtains the autocorrelation function (ACF). Once the ACF has been estimated, it is used to perform the Stouffer-Liptak-Kechris (slk) correction, where each p-value is adjusted according to adjacent p-values that are weighted according to the ACF [Pedersen et al., 2012]. A particular p-value will be reduced if its neighbors have low p-values and little autocorrelation, and will likely remain high if its neighbors have high p-values.

A q-value score is calculated based on either the FDR control approach of Benjamini and Hochberg [Benjamini and Hochberg, 1995], or a null model based on shuffled data [Pedersen et al., 2012]. Regions of interest are then defined as series of adjacent low p-values. This search can be based on the FDR q-value, the slk-corrected p-value, or on the original p-value, where the boundaries of the regions of interest are defined using one of these values.

Once the regions of interest are identified, a p-value is assigned to each region using the slk correction [Pedersen et al., 2012]. First, the ACF is calculated. Then, the corrected p-value for each region is calculated using the slk correction of the original, uncorrected p-values within the region.

2.6 Probe Lasso

Butcher and Beck (2015) developed another approach for detecting differentially methylated regions called Probe Lasso [Butcher and Beck, 2015]. The method can be implemented using the R/Bioconductor package ChAMP [Morris et al., 2014]. The Probe Lasso is an approach that defines regions as candidates for DMRs by gathering neighboring

significant signals through a flexible window called a ”probe-lasso”.

The main purpose for developing this algorithm was to redirect subsequent analysis away from exclusively using probes located in probe-dense regions like promoters or CpG islands (CGIs), which the microarray is skewed toward [Butcher and Beck, 2015]. Probe Lasso additionally uses information from potentially important intergenic regions that have largely been ignored. The authors point out that probe spacing is not uniform on the array. Probes within 200 bp of a transcription start site are densely spaced. Probes in intergenic regions, however, are less densely spaced. Probe density also decreases the further a probe is from a CGI. To account for this uneven spacing, Probe Lasso generates dynamic, flexible windows, called ”lassos”. These lassos have a center and a radius, and are ”thrown” around a probe and its radius; the lasso extends both upstream and downstream, centering on the probe itself. These probe-lassos are derived using the data as well as several user-specified options.

Probe Lasso uses an approach for p-value correction similar to that in Comb-p [Butcher and Beck, 2015]. Comb-p, described previously, is a 'moving averages' method of p-value correction that does not depend on the test used to generate the p-values [Pedersen et al., 2012]. Comb-p uses the auto-correlation information to first correct individual probe p-values, then defines DMRs based on peaks of corrected p-values [Pedersen et al., 2012]. However, Probe Lasso first gathers neighboring significant signals, and then uses auto-correlation information to combine the p-values of probes within a DMR.

Probe Lasso first calculates probe spacing for each probe in the dataset by binning the data into one of the 28 genetic/epigenetic categories (7 gene features x 4 CpG

island relationships), such as the first exon of a gene and a CpG island shore [Butcher and Beck, 2015]. Next, users select the lassoStyle and lassoRadius. The lassoStyle option specifies whether the probe-lasso sizes will be at most 2 times the lassoRadius bp, or at least this amount. Probe Lasso then identifies the genetic/epigenetic category that conforms to the user specified lassoRadius and creates probe-lassos that vary according to each specific genetic/epigenetic feature. This results in 28 dynamic window sizes, the ”probe-lassos”.

Depending on which genetic/epigenetic feature a probe maps to, an appropriately sized probe-lasso is then ”thrown” around each probe [Butcher and Beck, 2015]. Determining whether a probe is significant is determined by a user-specified threshold. Then Probe Lasso counts the number of significant probes that were ”caught”. Overlapping and neighboring lassos are merged if they are separated by less than another user-specified threshold. A DMR is called when there is no more merging of probe-lasso boundaries. DMR coordinates are defined by the minimum and maximum genomic coordinates of the probe-lasso boundaries for probes in the DMR. DMRs that are smaller than the user-specified minimum number of CpG sites are excluded from further analysis.

A p-value is then estimated for the DMR itself [Butcher and Beck, 2015]. The authors point out that because DNA methylation levels at neighboring probes can be substantially correlated [Eckhardt et al., 2006], Fisher’s method [Fisher, 1948] for combining p-values is inappropriate. Instead, Probe Lasso uses Stouffer’s method [Samuel A. Stouffer, 1949] to assign weights to individual p-values before combining them. These weights are based on the underlying correlation structure of the measured Beta-values. Probe Lasso recovers all normalized Beta-values and p-values of probes in a DMR, and the correlation matrix of the

Beta-values within each DMR is estimated. Each probe's p-value is then weighted by "the inverse sum of its squared correlation coefficient". This has the effect of down-weighting p-values of highly correlated probes and up-weighting p-values of uncorrelated probes. The package finally adjusts DMR p-values for multiple testing using the false discovery rate (FDR) method of Hochberg and Benjamini [Hochberg and Benjamini, 1990].

2.7 Bumphunter

Jaffe et al. (2012) implemented a statistical model in their R/Bioconductor package called Bumphunter as part of a method for finding DMRs [Jaffe et al., 2012]. The model can also include covariates or other potential confounders such as batch effects. While this method was specifically developed for CHARM microarrays, the authors explicitly approve of its use with 450K data as well. This method also has the ability to determine regions of interest, instead of using predefined ones based on genomic annotations [Robinson et al., 2014].

The authors fit a generalized linear model to methylation data as follows:

$$Y_{ij} = \mu(t_j) + \beta(t_j)X_i + \sum_{k=1}^{p} \gamma_k(t_j)Z_{i,k} + \sum_{l=1}^{q} a_{l,j}W_{i,l} + \varepsilon_{i,j}, \tag{2.1}$$

where $Y_{i,j}$ is the epigenomic measurement at the j-th genomic locus for individual i, which has been appropriately normalized and transformed [Jaffe et al., 2012]. This could be the logit transformation of the Beta-value, for example. The variable t_j denotes the location on the genome of the j-th locus (such as 'chromosome 2, position 42233500'). The population baseline level of the epigenomic measurement is $\mu(t_j)$, which could be the baseline methylation level of the controls in a case-control study, for example. The variable

X_i is the phenotype of interest, such as disease status, but it could also be continuous. The slope parameters $\beta(t_j)$ describe the association between the phenotype of interest and the epigenomic measurement at location t_j. Genomic locations of interest are those t_j for which $\beta(t_j) \neq 0$.

The model in (2.1) can include covariates, both measured and unmeasured during the data collection procedure [Jaffe et al., 2012]. Note that these covariates are allowed to be correlated with the variable of interest, making them potential confounders in the relationship between the response and the variable of interest. The model can include measured confounders like sex, age, race, or other confounders of interest. These are denoted by Zs in the model, with each column of the Z matrix representing a different measured confounder. The $\gamma_k(t_j)$ represent the effect of measured confounder k at locus t_j. The model can also account for unmeasured confounders such as batch effects. These are estimated using Surrogate Variable Analysis (SVA) [Leek and Storey, 2007], and are denoted by the W's in the model, with each column of the W matrix representing a different confounder estimated using SVA. The $a_{l,j}$ represent the effect of unmeasured confounder l at locus t_j.

SVA is a statistical method used for modeling unexplained variability in genomic measurements, such as batch effects [Leek and Storey, 2007]. A commonly used statistical technique that uncovers structures like this is principal component analysis (PCA) [Jaffe et al., 2012]. In high-throughput experiments, the first few principal components are often associated with unwanted sources of variability [Leek et al., 2010]. However, the authors of Bumphunter stated that they chose SVA over PCA because of a concern that removing these components may result in the removal of an important biological signal

[Jaffe et al., 2012, Leek and Storey, 2007]. SVA uses an iterative procedure to estimate both the number of unmeasured confounders and the confounders themselves. While SVA was originally designed to handle batch effects in gene expression data, it can also be used with appropriately transformed DNA methylation data [Jaffe et al., 2012].

The remaining unexplained variability in the model is represented by an error term, $\varepsilon_{i,j}$, which includes both the variability associated with measurement error and natural biological variability [Jaffe et al., 2012]. The measurement error is assumed to be from a stationary random process with symmetrical marginal distribution centered at 0. The authors allowed the error variance $var(\varepsilon_{i,j}) = \sigma^2(t_j)$ to depend on genomic location t_j, and assumed an unstructured correlation structure.

The authors of Bumphunter define regions of interest to be contiguous CpG sites for which there is evidence that $\beta(t) \neq 0$ for all genomic locations within the region [Jaffe et al., 2012]. These are the genomic regions where methylation levels at consecutive measured locations are associated with the phenotype of interest, such as disease status. The authors state that, based on previous work and biological insight, $\beta(t)$ can be modeled as a smooth function of genomic position since methylation levels for CpGs within 100 bases have been shown to be significantly correlated [Eckhardt et al., 2006]. They go on to explain that since $\beta(t) = 0$ for most of the genome, $\beta(t)$ can be thought of as a horizontal line with N bumps [Jaffe et al., 2012]. The goal of the Bumphunter algorithm is to find these bumps; in other words, to determine the regions of interest by detecting these bumps. They do this with a four step approach: (i) estimate the $\beta(t_j)$ for each t_j; (ii) use these to estimate the smooth function $\beta(t)$; (iii) use the smooth function to estimate the regions of interest; and (iv) use permutation tests to assign statistical uncertainty to each estimated

region.

In order to estimate the $\beta(t_j)$ for each t_j, the unmeasured confounders are first estimated using SVA [Jaffe et al., 2012, Leek and Storey, 2007]. With the SVA estimates in place in the design matrix, least squares is used to fit the model at each location t_j, producing locus-specific estimates $\hat{\beta}(t_j)$. Although $\hat{\beta}(t_j)$ is an unbiased estimate of $\beta(t_j)$ for each t_j, the authors point out that the assumption that $\beta(t)$ is smooth implies that precision can be improved with smoothing techniques. The $\hat{\beta}(t_j)s$ are then smoothed using a robust loess [Cleveland, 1979] smoother, a smoother that is robust to outliers, resulting in the smoothed estimates $\tilde{\beta}(t)$.

The loess smoother used is a robust locally weighted polynomial regression model that is fit locally to the data [Cleveland, 1979]. The weighting helps to reduce the effect of outliers on the smoothed estimates, and is based on the distances that points in a specified neighborhood are from the target loci. For Bumphunter, Jaffe et al. (2012) selected a smoothing window ranging from 300 to 900 bp, and weighted each point using the standard error obtained in the linear model fit [Jaffe et al., 2012]. The smoothing window size was selected based on a review of the literature, as well as their own simulation results.

After smoothing, candidate regions for DMRs are generated for contiguous CpG sites for which $\tilde{\beta}(t) > K$ or $\tilde{\beta}(t) < K$, for some predetermined threshold K, such as the 99th percentile of the distribution of the $\tilde{\beta}(t)s$ [Jaffe et al., 2012]. Permutation or bootstrap techniques are used to assess statistical uncertainty for each candidate region by estimating the probability that an observed region occurred by chance, given $\beta(t_j) = 0$ across the genome. The authors stated that these techniques accommodate the correlated measurement errors, batch effects, and the high-throughput nature of the data.

The authors propose two approaches for generating data with $\beta(t_j) = 0$ for all j, to be used as described above, that preserve all other statistical characteristics of the original data including batch effects and correlated errors [Jaffe et al., 2012]. The first approach permutes the variable of interest X_i and re-runs the entire bump hunting procedure. This is done $B = 1000$ times, and for each permutation, $b = 1, ..., B$, a set of null areas $A^{\star}_{n,b}$ for $n = 1, ..., \tilde{N^{\star}_b}$, the total number of regions for each permutation, is produced. Since permutation can be rather slow, the authors also developed a faster approach based on the application of the bootstrap to linear models discussed by Efron and Tibshirani [Efron and Tibshirani, 1994]. This method yielded results that were in practically equivalent to the first method, but arrived at them more efficiently.

The authors pointed out that any regions identified in these permuted or bootstrap datasets are actually 'null' candidate regions occurring by chance [Jaffe et al., 2012]. These procedures are repeated hundreds of times to generate a distribution of null candidate regions. Each region is summarized by what they refer to as its area, computed as $A_n = \sum_{j \in \hat{R}_n} |\tilde{\beta}(t_j)|$ for candidate regions $\hat{R}_n$. This area can then be used to rank regions of interest for further investigation. Empirical p-values are defined as the fraction of null areas greater than each observed area. For example, an observed area greater than 95 percent of the bootstrap areas will be assigned an empirical p-value of 0.05. Two methods are proposed to account for the multiple testing problem when determining which candidate regions are significant. These are Storey's optimal discovery procedure [Storey, 2007], and a method based on the family-wise error rate (FWER) [Shaffer, 1995].

2.8 DMRcate

DMRcate is another method, proposed by Peters et al. (2015), that has the ability to determine regions of interest instead of using predefined ones based on genomic annotations [Peters et al., 2015]. The authors state that about 1/4 of the CpG sites assayed by the 450K array are in intergenic regions. As a result, there is no associated gene annotation. They go on to explain that DMRs in these regions may contain trans-acting enhancers or other regulatory mechanisms, and should be considered along with those that have an explicit annotation, such as belonging to a promoter region of a gene.

First, a linear model is fit with the use of empirical Bayes methods to impose variance shrinkage [Peters et al., 2015]. This initial fitting is performed using *limma* [Smyth, 2004, Smyth et al., 2005], one of the most widely used R packages for analyzing microarray data. This model can also include covariates. The statistic collected from this step is the squared t-test statistic for use in the next step.

DMRcate collects the squared t-test statistic, whereas Bumphunter collects a signed estimate of the slope at each location [Peters et al., 2015]. This is because there is a concern that using a signed statistic could result in a loss of sensitivity to detect differential methylation due to signal canceling, when the direction of the effect changes abruptly [Day et al., 2013]. The authors explain that promoter methylation is associated with gene silencing, and genic methylation is associated with upregulation, and that this can result in a region with short-range methylation sign change. They suggest that Bumphunter appears to be less able to pick up these abrupt changes than other methods.

The squared t-test statistics are then smoothed using kernel smoothing with a Gaussian kernel on each chromosome [Peters et al., 2015]. Significance is assessed by modeling the smoothed values as a scaled chi-square random variable using the method of Satterthwaite [Satterthwaite, 1946]. Approximate p-values are then computed based on this method using the chi-squared distribution. The Benjamini-Hochberg adjustment for multiple testing is then performed, and significant CpG sites are agglomerated together. Sites can be at most 1000 base pairs from each other to be considered as part of the same DMR.

3. METHODOLOGY

It is important for a new method in detecting differentially methylated regions to be able to include covariates in the relationship between methylation and the variable of interest, to be able to determine differentially methylated regions instead of using predefined ones based on genomic annotations, and to assign significance to these regions [Robinson et al., 2014]. Bumphunter is the most sophisticated statistical model that can be used for differential methylation analysis on the 450K array up to the point of this literature review that can accomplish all of these things [Jaffe et al., 2012]. Therefore the origin of the idea for this algorithm was to make some small adjustments to the Bumphunter algorithm, and to try to come up with a new statistical contribution along the way.

Like Bumphunter, the proposed method can include covariates, including latent covariates that are unmeasured during the data collection procedure such as batch effects. All covariates are allowed to be correlated with the variable of interest, making them potential confounders in the relationship between methylation and the variable of interest, resulting in a potentially rank deficient design matrix. The new method also has the ability to determine regions of interest, instead of using predefined ones based on genomic annotations, and will assign significance to these regions as in Bumphunter. This new method will be called DMR Detector.

DMR Detector originally attempted to make five key modifications to the Bumphunter algorithm, however two were shown to be problematic in simulation and were removed. The first idea was how to perform the initial fitting to accommodate a potentially rank deficient design matrix. Bumphunter utilizes least squares via QR decomposition in their .getEstimate function, DMR Detector attempted to utilize a penalized least squares via ridge regression, however this idea was shown to be problematic in simulation and was removed. As a result, the .getEstimate function from Bumphunter was used instead. Another modification was related to the smoothing window size. Bumphunter smooths in clusters using a variable window size that can be quite small, DMR Detector attempted to smooth in clusters that were somewhat larger using a fixed sliding window, however this idea was also shown to be problematic in simulation and was removed. The clusters using a variable window size as defined in Bumphunter were used instead.

For the final version of the method, DMR Detector makes three key modifications to the Bumphunter algorithm. The first modification is what statistic to collect from the initial fitting for further analysis. Bumphunter collects a signed statistic for the effect of the variable of interest at this step, DMR Detector collects the absolute value of that statistic. The second modification is the major focus of this paper, and is a suggestion for new statistical methodology to perform the smoothing step using kernel smoothing. Bumphunter utilizes the loess [Cleveland, 1979] smoother, which assumes independent and identically distributed errors in a nonparametric regression model. DMR Detector smooths under the assumption of correlated errors, using a newly proposed correlation-adjusted kernel weight. The third modification is related to how to define regions of interest. Bumphunter defines regions to be where there are at least 2 adjacent CpG sites that

are above a certain threshold. DMR Detector defines regions as not necessarily adjacent CpG sites, but rather as a region where at least 75 percent of the CpG sites are above the threshold.

3.1 The Model

The proposed generalized linear model for differential methylation analysis is very similar to the one proposed in Bumphunter that is defined in model (2.1) [Jaffe et al., 2012]. We let

$$Y_{ij} = \mu(t_j) + \beta(t_j)V_i + \sum_{d=1}^{D} \phi_d(t_j)Z_{id} + \sum_{l=1}^{L} \gamma_{lj}U_{il} + \varepsilon_{ij}, \tag{3.1}$$

for $i = 1,...,n$ individuals, and $j = 1,...,J$ genomic loci, where the response variable Y_{ij} is the methylation measurement at the j-th genomic locus for individual i, t_j is the actual location on the genome of the j-th genomic locus where each methylation measurement was collected, $\mu(t_j)$ is the mean baseline level of the methylation measurement at each loci, V_i is the measurement of the variable of interest for each individual, $\beta(t_j)$ is for the association between the variable of interest V_i and the methylation response Y_{ij} at genomic location t_j, Z_{id} is the measurement for the ith individual and dth measured confounder, $d = 1,...,D$, with parameter $\phi_d(t_j)$ for the effect of measured confounder d at locus t_j, U_{il} is the measurement for the ith individual and lth unmeasured confounder, $l = 1,...,L$, with parameter γ_{lj} for the effect of unmeasured confounder l at locus t_j, and error term ε_{ij}.

The response variable Y_{ij} is the methylation measurement at the j-th genomic locus for individual i, which has been appropriately normalized and transformed [Jaffe et al., 2012].

This could be the logit transformation of the methylation proportion, or the commonly used M-value on the 450K array [Jaffe et al., 2012, Du et al., 2010]. The nx1 response vector for the methylation measurement at each loci can be referred to as $\boldsymbol{y}_j = (Y_{1j}, ..., Y_{nj})'$, with its corresponding random variable Y_j. Collectively the vectors can be referred to as the nxJ matrix $\boldsymbol{Y}$.

The variable t_j is the actual location on the genome of the j-th genomic locus where each methylation measurement was collected, such as 'chromosome 2, position 42233500' [Jaffe et al., 2012]. This t_j could be each CpG site probed on the 450K array for example. The parameter for the mean baseline level of the methylation measurement at each loci is $\mu(t_j)$, which would be the mean baseline methylation level of the controls in a case-control setting with no other covariates for example.

The variable V_i is the measurement of the variable of interest for each individual, which may be discrete such as disease status for example, but is also allowed to be continuous [Jaffe et al., 2012]. These measurements can collectively be referred to as the nx1 vector $\boldsymbol{v} = (V_1, ..., V_n)'$. The parameter $\beta(t_j)$ is for the association between the variable of interest V_i and the methylation response Y_{ij} at genomic location t_j. Locations of interest are those t_j for which $\beta(t_j) \neq 0$, such as the locations where the variable of interest is associated with DNA methylation for example. The $\beta(t_j)$ can collectively be referred to as the Jx1 parameter vector $\boldsymbol{\beta} = (\beta(t_1), ..., \beta(t_J))'$.

Like Bumphunter, the model can also include confounders, whether measured or unmeasured during the data collection procedure [Jaffe et al., 2012]. The measured confounders, such as sex, age, or race, are denoted by Zs in the model, with Z_{id} representing the measurement for the ith individual and dth measured confounder, $d = 1, ..., D$.

Collectively they can be referred to as the nxD matrix $\boldsymbol{Z}$, with each column representing a different measured confounder for all individuals. The parameter $\phi_d(t_j)$ is for the effect of measured confounder d at locus t_j.

Following Bumphunter, the unmeasured confounders such as batch effects or any other unmeasured variables, are estimated using Surrogate Variable Analysis (SVA) [Leek and Storey, 2007]. These are denoted by Us in the model, with U_{il} representing the measurement for the ith individual and lth unmeasured confounder, $l = 1, ..., L$. These unmeasured confounders can collectively be referred to as the nxL matrix $\boldsymbol{U}$, with each column representing a different unmeasured confounder. The parameter γ_{lj} is for the effect of unmeasured confounder l at locus t_j. These parameters are not notated as $\gamma_l(t_j)$ in order to remind the reader that they are parameters for the variables that were not measured at location (t_j), they are for the effect of the unmeasured confounders that are estimated using SVA.

The error term ε_{ij} is for the unexplained variability in the model, and is assumed to follow a symmetric distribution with mean zero [Jaffe et al., 2012]. The error variance is allowed to be different at each location t_j due to biological variation, and an unstructured correlation structure between the errors from different genomic locations is assumed. However the error process across the genome is considered a stationary random process, as in Bumphunter [Jaffe et al., 2012]. This point will be mentioned again in more detail later. The nx1 error vectors at each loci can be referred to as $\boldsymbol{\varepsilon}_j = (\varepsilon_{1j}, ..., \varepsilon_{nj})'$.

Real Data The real data used for the proposal is a published Autism dataset that was analyzed using Bumphunter involving 13 postmortem brain tissue samples

[Ladd-Acosta et al., 2014]. The data were downloaded from the Gene Expression Omnibus (GEO), and involve 7 Autism cases and 6 controls from the National Institute of Child Health and Human Development Brain and Tissue Bank for Developmental Disorders at the University of Maryland, Baltimore, MD, USA. The Beta-values were downloaded from the GEO system and transformed to M-values according to the relationship defined in equation (1.1). All computing was performed in R Version 3.3.1 [R Core Team, 2016]. The method was demonstrated for the proposal using 200 CpG sites around a DMR identified in the paper for the Cerebellum.

3.2 Surrogate Variable Analysis (SVA)

The first step is to estimate any unmeasured confounders using Surrogate Variable Analysis, or SVA [Leek and Storey, 2007]. The SVA algorithm identifies latent, unmeasured variables that are allowed to be correlated with the variable of interest, making them potential confounders in the relationship between the response and the variable of interest. However SVA cannot accept a rank deficient design matrix. Therefore, if some measured covariates are included in the design matrix and they are highly correlated with each other or the variable of interest, SVA will not attempt to identify any other confounders. The method will proceed to the next step to fit the model using the rank deficient design matrix. If the measured covariates are independent, then SVA will estimate any unmeasured confounders that need to be adjusted for in the regression before proceeding.

The SVA algorithm identifies an orthogonal set of vectors $\boldsymbol{h}_k$, $k = 1, ..., K$ $(K \leq L)$ that

spans the same linear space as the U_{il} [Leek and Storey, 2007]. That is, it identifies the $\boldsymbol{h}_k$ such that $\sum_{l=1}^{L} \gamma_{lj} U_{il} = \sum_{l=1}^{K} \xi_{kj} h_{ik}$. Using model (3.1), this implies

$$\begin{aligned} Y_{ij} &= \mu(t_j) + \beta(t_j) V_i + \sum_{d=1}^{D} \phi_d(t_j) Z_{id} + \sum_{l=1}^{L} \gamma_{lj} U_{il} + \varepsilon_{ij} \\ &= \mu(t_j) + \beta(t_j) V_i + \sum_{d=1}^{D} \phi_d(t_j) Z_{id} + \sum_{k=1}^{K} \xi_{kj} h_{ik} + \varepsilon_{ij} \end{aligned} \tag{3.2}$$

for $i = 1, ..., n$ individuals, and $j = 1, ..., J$ genomic loci. The algorithm estimates the linear combination $\gamma_{lj} U_{il}$, not the specific values of U_{il}. The $\boldsymbol{h}_k = (h_{1k}, ..., h_{nk})'$, $k = 1, ..., K$, are the K right non-zero singular vectors obtained by the singular value decomposition of the matrix with entries $\sum_{l=1}^{L} \gamma_{lj} U_{il}$. These $\boldsymbol{h}_k$ are the "surrogate variables".

This matrix with entries $\sum_{l=1}^{L} \gamma_{lj} U_{il}$ can be referred to as a reduced matrix $\boldsymbol{X}_r$ of size mxn. The singular value decomposition of $\boldsymbol{X}_r$ is, $\boldsymbol{X}_r = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T$. If the rank of $\boldsymbol{X}_r$ is k, $\boldsymbol{U}$ is Jxk, $\boldsymbol{D}$ is kxk, $\boldsymbol{V}$ is kxk, $\boldsymbol{D} = diag(\lambda_1, ..., \lambda_k)$, and $\lambda_i = \sqrt{\lambda_i^2}$, where the λ_i^2 are the nonzero eigenvalues of $\boldsymbol{X}_r'\boldsymbol{X}_r$ or $\boldsymbol{X}_r\boldsymbol{X}_r'$, then the k columns of $\boldsymbol{V}$ are the normalized eigenvectors of $\boldsymbol{X}_r'\boldsymbol{X}_r$, and the k columns of $\boldsymbol{U}$ are the normalized eigenvectors of $\boldsymbol{X}_r\boldsymbol{X}_r'$ corresponding to eigenvalues λ_i^2. These k columns of $\boldsymbol{U}$ can be used to estimate the $\boldsymbol{h}_k$, denoted as $\hat{\boldsymbol{h}}_k$, $k = 1, ..., K$, and are the "surrogate variables".

Then any further analyses involve the model

$$Y_{ij} = \mu(t_j) + \beta(t_j) V_i + \sum_{d=1}^{D} \phi_d(t_j) Z_{id} + \sum_{k=1}^{K} \xi_{kj} \hat{h}_{ik} + \varepsilon_{ij}, \tag{3.3}$$

which is an approximation of model (3.2). Collectively the $\boldsymbol{h}_k$ can be referred to as the nxK matrix $\boldsymbol{H}$, with estimated values $\hat{\boldsymbol{h}}_k$ and $\hat{\boldsymbol{H}}$, respectively.

3.3 Initial Fitting

Following Bumphunter, there is an initial fitting at each genomic location using the model below [Jaffe et al., 2012]. At each genomic location, model (3.3) can be written as follows. Let

$$\boldsymbol{y}_j = \boldsymbol{X}\boldsymbol{\theta}_j + \boldsymbol{\varepsilon}_j \tag{3.4}$$

for $j = 1, ..., J$ genomic loci, where

$\boldsymbol{y}_j$ is the nx1 response vector for the n individuals at loci j, $\boldsymbol{y}_j = (Y_{1j}, ..., Y_{nj})'$,

$\boldsymbol{X} = [\mathbf{1}\ \boldsymbol{v}\ \boldsymbol{Z}\ \hat{\boldsymbol{H}}]$ is the nxp design matrix including intercept, $p = 2 + D + K$,

$\hat{\boldsymbol{H}} = (\hat{\boldsymbol{h}}_1, ..., \hat{\boldsymbol{h}}_K)$ are the estimated surrogate variables,

$\boldsymbol{\theta}_j = [\mu(t_j)\ \beta(t_j)\ \phi_1(t_j) ... \phi_D(t_j)\ \xi_{1j} ... \xi_{Kj}]'$ is the px1 parameter vector, and

$\boldsymbol{\varepsilon}_j$ the nx1 error vector, with

$$E(\boldsymbol{\varepsilon}_j) = \mathbf{0}$$

and

$$Cov(\boldsymbol{\varepsilon}_j) = \sigma^2(t_j)\boldsymbol{I_n}$$

which implies that the responses from each individual are independent at each particular loci [Jaffe et al., 2012].

The first new idea for the method proposed in this dissertation was how to perform the initial fitting for model (3.4). There are two scenarios here. The first scenario assumes that

$\boldsymbol{X}$ is nxp of rank p. This implies that no unmeasured covariates identified by SVA in $\hat{\boldsymbol{H}}$ are correlated with the variable of interest $\boldsymbol{v}$, no measured covariates in $\boldsymbol{Z}$ are correlated with the variable of interest $\boldsymbol{v}$, and that $\boldsymbol{Z}$ is nxD of rank D because all measured covariates in the model are independent. This scenario also accounts for the case where there are no covariates involved other than the variable of interest $\boldsymbol{v}$ and the intercept. In this case, the ordinary least squares (OLS) estimator is the best linear unbiased (BLU) estimator of $\boldsymbol{\theta}_j$ [Graybill, 1976].

The second scenario assumes that $\boldsymbol{X}$ is nxp of rank less than p. This could be because SVA identified unmeasured confounders in $\hat{\boldsymbol{H}}$ that are correlated with the variable of interest $\boldsymbol{v}$, or because some measured confounders in $\boldsymbol{Z}$ are correlated with the variable of interest $\boldsymbol{v}$, or because $\boldsymbol{Z}$ is nxD of rank less than D because some measured covariates included in the model are correlated with each other. In any case, there is a multicollinearity problem. With a multicollinearity problem, the coefficients can not be estimated well, suffering from high variance [Hastie et al., 2009]. As a result, a biased estimator from a procedure like ridge regression could be used to reduce the variance due to the multicollinearity.

3.3.1 Ridge Regression

Recall that DMR Detector attempted to utilize ridge regression for the initial fitting, however this idea was shown to be problematic in simulation and was removed. It will be explained here to provide justification for its attempted use, and will be revisited in the simulation chapter to provide justification for its removal.

Ridge regression utilizes the idea of shrinkage by minimizing a penalized residual sum of squares. This imposes a size constraint on the coefficients. As a result, the variance due to the multicollinearity is reduced. Ridge regression was selected for use over similar methods such as lasso or elastic net regression because those methods essentially perform a variable selection, setting some coefficients to zero during the shrinkage [Hastie et al., 2009]. This is not the desired behavior because DMRs are investigated with respect to a certain variable of interest. Since these other methods could drop the variable of interest during the fitting simply because another variable was slightly more important, ridge regression was selected for use instead.

Ridge regression minimizes a penalized residual sum of squares, which is defined using model (3.4) as

$$RSS(\lambda_j) = (\boldsymbol{y}_j - \boldsymbol{X}\boldsymbol{\theta}_j)^T(\boldsymbol{y}_j - \boldsymbol{X}\boldsymbol{\theta}_j) + \lambda_j \boldsymbol{\theta}_j^T \boldsymbol{\theta}_j$$

where $\lambda_j \boldsymbol{\theta}_j^T \boldsymbol{\theta}_j$ is a quadratic penalty term with tuning parameter λ_j, $\boldsymbol{y}_j$ is the response from model (3.4), and $\boldsymbol{X}$ is the design matrix from model (3.4) without the intercept term, as in $\boldsymbol{X} = [\boldsymbol{v}\, \boldsymbol{Z}\, \hat{\boldsymbol{H}}]$, where all remaining variables have been centered [Hastie et al., 2009]. Minimizing $RSS(\lambda_j)$ with respect to $\boldsymbol{\theta}_j$ results in the ridge estimator defined as

$$\hat{\boldsymbol{\theta}}_j^{Ridge} = (\boldsymbol{X}'\boldsymbol{X} + \lambda_j \boldsymbol{I_p})^{-1} \boldsymbol{X}'\boldsymbol{y}_j \tag{3.5}$$

where the optimal tuning parameter λ_j can be identified using a cross validation procedure such as generalized cross validation that will be discussed shortly. Since the intercept $\mu(t_j)$ from model 3.4 is left out of the design matrix during the ridge regression, it is estimated

with $\mu(t_j) = \bar{y}_j = \frac{1}{n}\sum_{i=1}^{n} Y_{ij}$ [Hastie et al., 2009]. Note that the ridge estimator becomes the OLS estimator when $\lambda_j = 0$. Although in this case, the original, uncentered design matrix with intercept is used as usual.

3.3.2 Cross Validation

Techniques used to identify the optimal tuning parameter λ_j often involve estimators for the expected squared prediction error (ESPE) on a new observation [Altman, 1990, Hastie et al., 2009]. The ESPE is defined as

$$ESPE = E[Y_j^{new} - \hat{Y}_j]^2$$

for the random variable Y_j representing the vector $\boldsymbol{y}_j = (Y_{1j}, ..., Y_{nj})'$ from model (3.4), and $\hat{Y}_j$ its predicted value. An estimate of ESPE can be used to determine the optimal tuning parameter λ_j by selecting several values of λ_j to check, then finding which value minimizes an estimate of ESPE averaged over all of the design points. In other words, we find the one that minimizes the expected mean squared prediction error (EMSPE). Cross validation is a common technique used for this purpose. For example, the well-known leave-one-out cross validation estimate of expected mean squared prediction error is

$$LCV(\hat{Y}_j)_{EMSPE} = \frac{1}{n}\sum_{i=1}^{n}[Y_{ij} - \hat{Y}_{-i,j}]^2$$

where the fitted value $\hat{Y}_{-i,j}$ is computed using all data except the i-th, at each genomic loci, $j = 1, ..., J$ [Hastie et al., 2009].

Generalized Cross Validation

Generalized cross validation (GCV) was developed as an alternative to leave-one-out cross validation [Craven and Wahba, 1977]. It can alleviate the tendency of usual cross validation procedures to undersmooth, and can be computationally more efficient, which is always a concern with genomic data [Hastie et al., 2009]. So that is why it was selected for use here.

GCV is appropriate for linear fitting methods [Hastie et al., 2009, Craven and Wahba, 1977]. Linear fitting methods are ones that can be written using model (3.4) as $\hat{\boldsymbol{y}}_j = \boldsymbol{S}_j \boldsymbol{y}_j$, for predicted values $\hat{\boldsymbol{y}}_j$, and nxn matrix $\boldsymbol{S}_j$ that does not include the $\boldsymbol{y}_j$ [Hastie et al., 2009]. Ridge regression is a linear fitting method, where the fitted values can be written in that way using model (3.4) as follows. Let

$$\begin{aligned}\hat{\boldsymbol{y}}_j &= \boldsymbol{X}\hat{\boldsymbol{\theta}}_j^{Ridge} \\ &= \boldsymbol{X}(\boldsymbol{X}'\boldsymbol{X} + \lambda_j \boldsymbol{I_p})^{-1}\boldsymbol{X}'\boldsymbol{y}_j \\ &= \boldsymbol{S}_j \boldsymbol{y}_j\end{aligned}$$

where $\boldsymbol{S}_j = \boldsymbol{X}(\boldsymbol{X}'\boldsymbol{X} + \lambda_j \boldsymbol{I_p})^{-1}\boldsymbol{X}'$. The GCV estimate of expected mean squared prediction error can then be defined as

$$GCV(\hat{Y}_j)_{EMSPE} = \frac{1}{n}\sum_{i=1}^{n}\left[\frac{Y_{ij} - \hat{Y}_{ij}}{1 - tr(\boldsymbol{S}_j)/n}\right]^2$$

where $tr(\boldsymbol{S}_j)$ is the trace of $\boldsymbol{S}_j$, calculated at each genomic loci, $j = 1, ..., J$.

3.3.3 Statistic for Variable of Interest

Following Bumphunter, only the estimate for the effect of the variable of interest is collected and considered further [Jaffe et al., 2012]. Since ridge regression was found to be problematic in simulation, the statistic for the variable of interest will be calculated using the .getEstimate function from Bumphunter in the final version of the method. However in the initial version of the method, it was calculated using ridge regression as follows. Let

$$\hat{\beta}(t_j) = \boldsymbol{l'}\hat{\boldsymbol{\theta}}_j^{Ridge} \tag{3.6}$$

for $\hat{\boldsymbol{\theta}}_j^{Ridge}$ defined in equation (3.5), and 1xp vector $\boldsymbol{l'}$=(0 1 0...0), with a 1 in the position matching the column number of the variable of interest in the design matrix, $\boldsymbol{X}$. For model (3.4), the variable of interest is in column 2.

Statistic collected from initial fitting

The next new idea for the method proposed in this paper is about what to collect as the statistic from this step. Bumphunter collects the signed $\hat{\beta}(t_j)$ defined in equation (3.6) at this stage [Jaffe et al., 2012]. The new idea is to collect the absolute value of $\hat{\beta}(t_j)$, called $\hat{f}(t_j)$. Let

$$\hat{f}(t_j) = |\hat{\beta}(t_j)| \tag{3.7}$$

with corresponding parameter

$$f(t_j) = |\beta(t_j)| \tag{3.8}$$

The statistic $\hat{f}(t_j)$ is an unsigned statistic, collected without regard to the direction of the effect in order to allow both hypo- and hyper- methylated sites to be considered as part of the same DMR. The idea for this came from DMRcate where the squared t-test statistic was collected [Peters et al., 2015]. The authors of DMRcate explain that promoter methylation is associated with gene silencing, and genic methylation is associated with upregulation, and that this can result in a region with short-range methylation sign change [Day et al., 2013]. This raises concerns about a reduced ability to detect differential methylation due to signal canceling, when the sign of the effect changes abruptly. Collecting an unsigned statistic at this step can alleviate this problem. A similar idea is also used in comb-p [Pedersen et al., 2012] and Probe Lasso [Butcher and Beck, 2015] [Peters et al., 2015].

3.4 Smoothing

Next is the smoothing step. Background knowledge, ideas, and notation used for smoothing came from Hastie et al. (2009) [Hastie et al., 2009]. The major part of the proposed new method is about how to smooth the two-dimensional scatterplot comprised of points $[t_j, \hat{f}(t_j)]$, to get smoothed points $[t_j, \tilde{f}(t_j)]$, $j = 1, ..., N_c$, for all N loci in each cluster c, $c = 1, ..., C$.

Recall that DMR Detector attempted to smooth in clusters using a fixed window size that was somewhat larger than Bumphunter. Initially, intuition suggested that DMR Detector may need the ability to reach out and find some less correlated points, and that a larger, fixed window might be preferable. However this idea was shown to be problematic

in simulation and was removed. In the final version of the method, these clusters are the same clusters as defined in Bumphunter, and represent a variable smoothing window size. They can be defined as clusters of CpG sites that are found together, with a user defined threshold termed the maxGap parameter, that determines how far apart CpG sites can be in order to be defined as part of the same cluster [Jaffe et al., 2012].

Simulations revealed that the optimal value for maxGap was 1000, meaning the loci must be within 1000 base pairs from each other to be considered as part of the same region of interest. This is congruent with what was suggested in DMRcate [Peters et al., 2015]. Bumphunter recommends a smaller size of 300 base pairs for the CHARM array, because this array has a median distance between probes of 36 to 70 base pairs, depending on the array version [Jaffe et al., 2012]. Since the 450K array has a median distance between probes of 300 base pairs [Jaffe et al., 2012], 1000 seems reasonable, and was further supported by simulation results that will be shown later.

So the smoothing will be performed on the scatterplot involving only the loci that are found in the same cluster. Then the window will slide to the next cluster and this process continues. The N_cx1 vector of loci can collectively be referred to as $\boldsymbol{t} = (t_1, ..., t_{N_c})'$, and similarly for the N_cx1 vectors for the initial fit statistics $\hat{\boldsymbol{f}} = (\hat{f}(t_1), ..., \hat{f}(t_{N_c}))'$, and the smoothed values $\tilde{\boldsymbol{f}} = (\tilde{f}(t_1), ..., \tilde{f}(t_{N_c}))'$.

3.4.1 Kernel Smoothing

Following Bumphunter, the proposed method also utilizes kernel smoothing [Jaffe et al., 2012, Cleveland, 1979]. Kernel smoothing is a method for estimating the mean

function in the nonparametric regression model

$$f = m(t) + \delta \tag{3.9}$$

where f is the random variable with observed values in the N_cx1 vector of initial fit statistics $\hat{\boldsymbol{f}} = (\hat{f}(t_1), ..., \hat{f}(t_{N_c}))'$ with elements that are defined in equation (3.7), $m(t)$ is a smooth deterministic mean function, and δ is an error process with mean zero [Altman, 1990]. DMR Detector uses the same idea of bump hunting as in Bumphunter, that because $\beta(t_j) = 0$ for most of the genome [Jaffe et al., 2012], then $f(t_j) = |\beta(t_j)| = 0$ for most of the genome, and the shape of the true function across the genome can be thought of as a horizontal line with an unknown number of bumps. The goal is to find these bumps.

The model in equation (3.9) fits separately at each target loci t_0, with each loci $t_1, ..., t_{N_c}$ having the opportunity to be the target loci once [Hastie et al., 2009]. In other words, we are smoothing the $\hat{f}(t_0)$ values of the target points $[t_0, \hat{f}(t_0)]$. Then the estimate of $m(t_0)$ is $\hat{m}(t_0)$, which is the estimated smoothed value, $\tilde{f}(t_0)$, at target loci t_0 [Cleveland, 1979]. Estimation of the smoothed value is done nonparametrically, and is based on using a kernel to assign weights to the $\hat{f}(t_j)$ values of points $[t_j, \hat{f}(t_j)]$, $j = 1, ..., N_c$, to determine the smoothed $\tilde{f}(t_0)$ value of the target point $[t_0, \tilde{f}(t_0)]$, where t_0 can be any of the locations $t_1, ..., t_{N_c}$ for the N loci in cluster c, $c = 1, ..., C$.

This t_0 subscript notation will be used to represent the target loci instead of i which may seem more intuitive, because i is the notation used for the $i = 1, ..., n$ individuals, which will also be needed to explain the new method proposed in this section. So all of the notation in the smoothing section will be with respect to a fit at a general target loci t_0, with each loci

having the opportunity to be the target loci once [Hastie et al., 2009].

Gaussian Kernel

The kernel used in the smoothing initially was the commonly used Gaussian kernel [Sheather, 2004]. The Gaussian kernel is defined as,

$$K_\lambda(t_0,t) = D\left(\frac{|t-t_0|}{\lambda_c}\right) \tag{3.10}$$

where

$$D(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}}$$

The kernel assigns weights that are based on the genomic distance $|t_0 - t_j|$ between the target loci t_0 and the other loci t_j, $j = 1,...,J$, with weights that die off smoothly as the genomic distance from the target loci increases. The tuning parameter λ_c controls the size of the kernel, playing the role of the standard deviation in the case of the Gaussian kernel. The c subscript on λ_c indicates that the smoothing is performed on each cluster separately, $c = 1,...,C$, as mentioned previously. The optimal value of the tuning parameter can be determined using GCV, results provided later.

The choice of which kernel to use is generally not of critical importance, as there are many kernels that could be appropriate for use in kernel smoothing. Gaussian kernels have nice theoretical properties, and is the kernel used in DMRcate [Peters et al., 2015]. The Epanechnikov kernel [Hastie et al., 2009], defined as

$$D(t) = \begin{cases} \frac{3}{4}(1-t^2), & if\ |t| \leq 1 \\ \\ 0, & otherwise \end{cases} \tag{3.11}$$

is considered the optimal unimodal kernel, but the difference with the Gaussian kernel is considered to be negligible [De Brabanter et al., 2011]. The Epanechnikov kernel is very similar to the Bisquare and Tricube kernels suggested for use with the loess [Cleveland, 1979] smoother used in Bumphunter [Jaffe et al., 2012]. Kernels like these where

$$D(t) = 0,\ if\ |t| > 1$$

can afford some computational efficiency over the Gaussian, because the weight becomes zero for all points after a certain distance [Cleveland, 1979]. The Gaussian kernel, on the other hand, has unbounded support and assigns weights to all points.

An initial investigation using GCV, results provided later, suggested the Gaussian kernel as being slightly better than the Epanechnikov when using the new method proposed in this paper. Also, some plots that will be shown later suggested that the Gaussian kernel may be more able to detect DMRs with small effect sizes than the Epanechnikov. This is an issue raised by Dr. Xiaoling Wang at Georgia Prevention Institute (GPI), and is also supported in the literature [Breton et al., 2017]. Dr. Wang explained that for some complex diseases, the effect sizes can be rather small, and are often too small for current DMR finding algorithms to detect. However simulation results showed that the Epanechnikov was actually slightly better than the Gaussian at detecting small effect sizes, so therefore the Gaussian kernel was used in the initial version of the method, but the Epanechnikov was used in the final

version of the method. Two commonly used kernel-weighted fit statistics used in kernel smoothing will now be discussed.

3.4.2 Commonly Used Kernel-Weighted Fit Statistics

The following kernel-weighted fit statistics can be used for estimating the mean function in the nonparametric regression model from equation (3.9) at each target loci t_0. This model at each loci can be written as

$$f(t_0) = m(t_0) + \delta_0 \tag{3.12}$$

for target loci t_0, with each loci $t_1, ..., t_{N_c}$ having the opportunity to be the target loci once, where the errors are assumed to be independent and identically distributed with mean zero [Hollander et al., 2015]. This implies that the errors are assumed to come from a stationary random process along the genome [Fuller, 1996].

Nadaraya-Watson Estimator

The usual Nadaraya-Watson kernel-weighted fit statistic for the smoothed y-value $\tilde{f}(t_0)$ can be calculated for the model in equation (3.12) at each target loci t_0, with each loci $t_1, ..., t_{N_c}$ having the opportunity to be the target loci once, as follows [Nadaraya, 1964, Watson, 1964, Hastie et al., 2009]. Let

$$\tilde{f}(t_0)_{NW} = \frac{\sum_{j=1}^{N_c} K(t_0, t_j)\hat{f}(t_j)}{\sum_{j=1}^{N_c} K(t_0, t_j)} \tag{3.13}$$

where the kernel weights $K(t_0,t_j)$ defined in equation (3.10) are assigned to the $\hat{f}(t_j)$ values defined in equation (3.7) that are based on the genomic distance $|t_0 - t_j|$ between the target loci t_0 and the other loci t_j, $j = 1,...,J$, with weights that die off smoothly as the genomic distance from the target loci increases.

Note that the Nadaraya-Watson estimator is a linear fitting method [Liu, 2001]. A linear fitting method is a method that can be factored into two parts, the $\hat{f}(t_j)$ and weights defined as $l_j(t_0)$ that do not involve the $\hat{f}(t_j)$ [Hastie et al., 2009]. Let

$$\begin{aligned}\tilde{f}(t_0)_{NW} &= \sum_{j=1}^{N_c} \left[\frac{K_\lambda(t_0,t_j)}{\sum_{j=1}^{N_c} K_\lambda(t_0,t_j)} \right] \hat{f}(t_j) \\ &= \sum_{j=1}^{N_c} l_j(t_0)\hat{f}(t_j)\end{aligned}$$

where $l_j(t_0) = \frac{K_\lambda(t_0,t_j)}{\sum_{j=1}^{N_c} K_\lambda(t_0,t_j)}$. In vector form using all N loci in cluster c that are smoothed together, this implies

$$\tilde{\boldsymbol{f}} = \boldsymbol{S}_c \hat{\boldsymbol{f}} \tag{3.14}$$

for the N_cx1 vector of smoothed values $\tilde{\boldsymbol{f}} = (\tilde{f}(t_1),...,\tilde{f}(t_{N_c}))'$, the N_cx1 vector of initial fit statistics $\hat{\boldsymbol{f}} = (\hat{f}(t_1),...,\hat{f}(t_{N_c}))'$, and N_cxN_c smoothing matrix $\boldsymbol{S}_c$ that does not include the $\hat{\boldsymbol{f}}$, where each element of each row in $\boldsymbol{S}_c$ is $l_j(t_0)$. The subscript on $\boldsymbol{S}_c$ implies that the smoothing matrix is different for each cluster, $c = 1,...,C$. Also note that the weights $l_j(t_0)$ for the Nadaraya-Watson sum to 1 [Liu, 2001].

It is known that locally weighted averages like the Nadaraya-Watson estimator can

be biased on the boundaries of the domain [Hastie et al., 2009]. This is because of the asymmetry of the kernel in that region. This bias can also be present in the interior of the domain as well, especially if the x-values are not equally spaced. One remedy for this is to fit a straight line locally instead of a constant like the Nadaraya-Watson. The local linear fit removes this bias.

Local Linear Estimator

Locally weighted linear regression fits a straight line locally for the model in equation (3.12) by solving a separate weighted least squares problem at each target loci t_0, with each loci $t_1, ..., t_{N_c}$ having the opportunity to be the target loci once [Hastie et al., 2009]. Let

$$\min_{\alpha_1(t_0),\alpha_2(t_0)} \sum_{j=1}^{N_c} K_\lambda(t_0,t_j)[f(t_j) - \alpha_1(t_0) - \alpha_2(t_0)t_j]^2$$

While the linear model is fit using all of the data in the region, it is only used to estimate the fit at the target loci. The local linear estimator is defined as follows. Let

$$\tilde{f}(t_0)_{LL} = \hat{\alpha_1}(t_0) + \hat{\alpha_2}(t_0)t_0 = \boldsymbol{b}(t_0)^T(\boldsymbol{B}^T\boldsymbol{W}(t_0)\boldsymbol{B})^{-1}\boldsymbol{B}^T\boldsymbol{W}(t_0)\hat{\boldsymbol{f}} \tag{3.15}$$

$$\boldsymbol{W}(t_0) = diag[K_\lambda(t_0,t_j)]$$

for $\boldsymbol{b}(t_0)^T = (1,t_0)$ the input vector for the target loci, which is a particular row of the N_cx2 design matrix with intercept $\boldsymbol{B} = (\boldsymbol{1},\boldsymbol{t})$, $\boldsymbol{W}(t_0)$ the N_cxN_c diagonal matrix with jth diagonal element $K_\lambda(t_0,t_j)$ defined in equation (3.10), and $\hat{\boldsymbol{f}} = (\hat{f}(t_1), ..., \hat{f}(t_{N_c}))'$, the initial fit statistics as defined in equation (3.7). The matrix $\boldsymbol{W}(t_0)$ is comprised of kernel weights

$K_\lambda(t_0, t_j)$ that are assigned based on the genomic distance $|t_0 - t_j|$ between the target loci t_0 and the other loci t_j, $j = 1, ..., J$, with weights that die off smoothly as the genomic distance from the target loci increases.

Note that the local linear estimator is also a linear fitting method [Hastie et al., 2009] like the Nadaraya-Watson, as in

$$\begin{aligned}\tilde{f}(t_0)_{LL} &= \boldsymbol{b}(t_0)^T(\boldsymbol{B}^T\boldsymbol{W}(t_0)\boldsymbol{B})^{-1}\boldsymbol{B}^T\boldsymbol{W}(t_0)\hat{\boldsymbol{f}} \\ &= \sum_{j=1}^{N_c} l_j(t_0)\hat{f}(t_j)\end{aligned}$$

Note that this implies

$$\tilde{\boldsymbol{f}} = \boldsymbol{S}_c\hat{\boldsymbol{f}} \tag{3.16}$$

in exactly the same way that was explained in equation (3.14). Also note that the weights for the local linear $l_j(t_0)$ sum to 1 as well [Cai, 2001].

The loess smoother used in Bumphunter is a robust local linear estimator [Cleveland, 1979, Hollander et al., 2015]. After performing the above regression, additional regressions are performed involving weighting points by their residuals from the first regression to reduce the impact of outliers.

Testing the independence assumption

To assess the suitability of the above estimators, the error independence assumption from model 3.12 was tested using the Autism data. The errors δ_0 were estimated for model 3.12 as

$$e_0 = \hat{f}(t_0) - \tilde{f}(t_0)_{NW} \tag{3.17}$$

for $\hat{f}(t_0)$ the observed initial fit statistic at target loci t_0 defined in equation 3.7, and $\tilde{f}(t_0)_{NW}$ the Nadaraya-Watson estimator defined in equation 3.13 using the optimal tuning parameter identified by GCV. The runs test for error randomness suggests evidence of dependence in the errors ($p < 0.0001$).

The SVA paper explains that the errors in model 3.2 are independent across loci, because all necessary confounders, measured and unmeasured, have been accounted for [Leek and Storey, 2007]. Leek and Storey (2008) further explain in a subsequent paper that if all true unmeasured confounders are known and included in the model, then all subsequent parameter estimates, test statistics, and p-values are independent across loci as well [Leek and Storey, 2008]. The authors of SVA then present model 3.3 as an approximation to model 3.2, with the same independent errors. This is why Bumphunter [Jaffe et al., 2012] uses a loess [Cleveland, 1979] smoother, which assumes the errors are independent across loci, when smoothing the slope parameter estimates for the effect of the variable of interest $\hat{\beta}(t_j)$ defined in equation (3.6). However, in the later paper, Leek and Storey (2008) mention that the unmeasured confounders must be estimated well in order to account for all of the dependence [Leek and Storey, 2008]. Therefore, in practice, it may be beneficial to allow for some degree of dependence to remain in model 3.3.

An additional concern that could affect the correlation structure is what particular values are being smoothed. As mentioned, Bumphunter is smoothing the slope parameter estimates for the effect of the variable of interest $\hat{\beta}(t_j)$ defined in equation (3.6). However

DMR Detector is smoothing the absolute value of those statistics which are defined in equation (3.7). As a result, there likely will be some unknown changes to the correlation structure due to flipping all points up to the upper quadrants. So it may be beneficial to allow for some dependence to remain due to this reason as well.

On the Autism data used in the proposal, SVA did not identify any unmeasured confounders. This implies we are working with model 3.3 where $\hat{h}_{ik} = 0 \ \forall i,k$, which implies that all necessary confounders for the model are expected to have already been accounted for. According to the SVA paper, this should mean that the errors are expected to be independent across loci, and that the subsequent parameter estimates are expected to be independent as well. However if we smooth the slope parameter estimates for the effect of the variable of interest $\hat{\beta}(t_j)$ defined in equation (3.6) used in Bumphunter, and test the residuals from that model, there is evidence that there may still be some degree of dependence using the runs test ($Z = 4.08, p < 0.0001$, with 87 negative errors and 113 positive which is appropriate). Additionally, if we smooth the absolute value of those statistics as we are suggesting for DMR Detector, there is evidence that whatever correlation was already present could potentially be magnified by flipping the points up to the upper quadrants, as the test statistic from the runs test is slightly more extreme ($Z = 4.52, p < 0.0001$, with 90 negative errors and 110 positive which is appropriate).

Therefore, to allow for an unknown amount of estimation error when estimating the unmeasured confounders, and to account for an unknown change in the correlation structure due to flipping all points up to the upper quadrants, the third modification to Bumphunter and major focus of this paper, is a suggestion for new statistical methodology to perform kernel smoothing that accounts for potentially correlated errors across the genome. As

will soon be shown, allowing for the potential for correlated errors in the smoothing step suggests prediction error could be improved.

3.4.3 Introduction to New Smoothing Method

Following Bumphunter and the loess smoother, which follows the assumptions from the model in equation (3.12), the errors in the new model are assumed to come from a stationary random process with mean zero [Jaffe et al., 2012, Cleveland, 1979, Hollander et al., 2015, Fuller, 1996]. However the new method does not assume that the errors are independent. Here it is assumed that

$$f(t_0) = m(t_0) + \delta_0 \tag{3.18}$$

$$E(\delta_0) = 0$$

$$Var(\delta_0) = \psi^2$$

$$Corr(\delta_0, \delta_j) = \rho(|t_0 - t_j|)$$

where $\rho(|t_0 - t_j|)$ is a stationary correlation function with $\rho(0) = 1$ that is based on the genomic distance $|t_0 - t_j|$ between the target loci t_0 and the other loci $t_j,\ j = 1, ..., J$ [Opsomer et al., 2001, Altman, 1990, Fuller, 1996].

The correlation function in the model defined in equation 3.18 is written as $\rho|t_0 - t_j|$ to indicate that it depends only on the distance $|t_0 - t_j|$ between the loci and not on their actual

values, a requirement for stationarity [Fuller, 1996]. A stationary correlation function implies that, for any subset of loci containing j^* observations of the total number of loci J, $j* \in J$, the correlation matrix of the errors $(\delta_1, \delta_2, ..., \delta_{j*})$ is the same as the correlation matrix of the errors $(\delta_{1+h}, \delta_{2+h}, ..., \delta_{j*+h})$ for constant lag h. This implies that points that are a certain distance from any target loci are expected to have the same correlation as points that are the same distance from another target loci.

The loess smoother used in Bumphunter is not designed to handle correlated errors [Hollander et al., 2015, Cleveland, 1979]. Also the method suggested in the paper for determining the optimal tuning parameter requires that they be independent as well [Altman, 1990, De Brabanter et al., 2011]. This is because it is essentially the usual leave-one-out cross validation (CV) procedure. It is a well-known problem that the presence of correlated errors causes a breakdown in usual cross validation procedures, like leave-one-out CV, when determining the optimal tuning parameter. Therefore, the loess smoother is not appropriate in this situation. Additionally, the Bumphunter algorithm does not even attempt to determine the optimal tuning parameter, using 0.3 by default. The consequences of this could be severe if the true optimal value is not near 0.3. Methods like cross validation need to be used to determine the optimal value.

However the usual cross validation methods tend to be "fooled" by the correlation [De Brabanter et al., 2011, Opsomer et al., 2001]. This is because they perceive the data structure to be due to the mean function, and attempt to use that information when estimating the trend [Opsomer et al., 2001]. If the errors are positively correlated, these usual methods will select an arbitrarily small tuning parameter, resulting in an estimate that is undersmoothed. If the errors are negatively correlated, they will select an arbitrarily large

tuning parameter, becoming oversmoothed. Many adjustments to usual cross validation procedures have been suggested to account for this problem.

The new idea proposed in this paper came from the idea of using bimodal kernels to account for the correlation, so that the usual methods for cross validation can be used [De Brabanter et al., 2011]. Like the commonly used unimodal kernels, bimodal kernels put more weight on points that are close to the target loci than those that are further away. However the weight is reduced on points that are very close to the target loci, as this is often where correlated points that are causing the correlated errors can be found. This essentially reduces the effect of those correlated points on the fitting of the target point.

If the correlated points have very little effect on the fitting of the target point, it is almost as if they are not there. Therefore, usual cross validation procedures can then be used to determine the optimal tuning parameter [De Brabanter et al., 2011]. The authors of the paper on bimodal kernels demonstrate this by using the usual leave-one-out CV procedure to determine the optimal tuning parameter for the bimodal kernel. Note that this demonstration can be found within a two-stage procedure used for support vector machines in the paper, but that is not important. What is important is an understanding of how the optimal tuning parameter is selected when using a kernel weight that has reduced the effect of the correlated points on the fitting of the target point. That idea will be followed here.

3.4.4 Correlation-Adjusted Kernel Weight

The proposed method in this paper is to modify commonly used kernel-weighted fit statistics by creating a correlation-adjusted kernel weight. This essentially reduces the

effect of the correlated points on the fitting of the target point like the bimodal kernel, although in a different way. The usual cross validation procedures like leave-one-out CV can then be used to determine the optimal tuning parameter. New estimators that are based on modifications to the Nadaraya-Watson estimator and the local linear estimator will be discussed.

The correlation-adjusted kernel weight adjusts the weight assigned by the kernel that is based on the genomic distance from the target loci, by the correlation with the sample at the target loci, r_{0j}. The correlation-adjusted kernel weight can be defined as

$$W(t_0, t_j) = \frac{1}{{r_{0j}}^2} K_\lambda(t_0, t_j) \tag{3.19}$$

where r_{0j} is the correlation with the sample at the target loci, which is raised to the 2nd power, and $K_\lambda(t_0, t_j)$ is the kernel weight defined in Equation 3.10. The origin of this idea began with using $\frac{1}{|r_{0j}|} K_\lambda(t_0, t_j)$ for the correlation adjustment. However a more formal investigation using GCV, results provided later, revealed that increasing the exponent to 2 was better. Note that it is not the sign of the correlation that is important. It is the magnitude.

Next up is how to formally define this correlation r_{0j}. As described, it is essentially the correlation with the sample at the target loci, which is the biological correlation between the sample at the target loci and the sample at each of the other loci. However it will be estimated using Spearman's rank correlation coefficient, which is simply the Pearson correlation coefficient applied to the ranks [Hollander et al., 2015]. The Spearman's rank correlation coefficient is defined as

$$r_{0j}{}^{S} = \frac{12\sum_{i=1}^{n}(R_i - \frac{n+1}{2})(S_i - \frac{n+1}{2})}{n(n^2-1)}$$

where R_i are the ranks of the methylation values at the target loci, and similarly for S_i the ranks at the other loci. If there are ties among the methylation values, the average rank of the tied cases is used. The Pearson correlation coefficient is defined as

$$r_{0j}{}^{P} = \frac{\sum_{i=1}^{n}(Y_{i0} - \bar{y}_0)(Y_{ij} - \bar{y}_j)}{\sqrt{\sum_{i=1}^{n}(Y_{i0} - \bar{y}_0)^2}\sqrt{\sum_{i=1}^{n}(Y_{ij} - \bar{y}_j)^2}}$$

for $\bar{y}_0 = \frac{1}{n}\sum_{i=1}^{n} Y_{i0}$, the mean of the methylation values at the target loci, and similarly for $\bar{y}_j$ at the other loci. In a formal investigation using GCV, results provided later, the Spearman correlation was found to have better performance than the Pearson. So it will be used as the formal definition in the proposed method.

Note that r_{0j} is not an estimate of the error correlation $\rho(|t_0 - t_j|)$, which may seem more intuitive to use for the correlation-adjustment. The correlation r_{0j} takes into account the biological variability at each loci. However $\rho(|t_0 - t_j|)$ is a stationary correlation function, which does not depend on which loci are being estimated, and is the same across the genome [Fuller, 1996]. It does not take into account the biological variability at each loci.

To estimate $\rho(|t_0 - t_j|)$, we can use the sample autocorrelation function (ACF) [Venables and Ripley, 2002, Fuller, 1996], denoted as $r(|t_0 - t_j|)$. The definition used for the sample ACF came from the built-in function "acf" in the "stats" package in R [R Core Team, 2016], which is simply a call to the same built-in function in C, which is the same built-in function in S [Venables and Ripley, 2002]. So using the definition provided for several major statistical software packages, the sample ACF $r(|t_0 - t_j|)$ is defined as

$$r(|t_0 - t_j|) = \frac{\frac{1}{N_c}\sum_{0=max(1,-h)}^{min(N_c-h,N_c)}[e_0 - \bar{e}][e_{0+h} - \bar{e}]}{\frac{1}{N_c}\sum_{0=1}^{N_c}[e_0 - \bar{e}][e_0 - \bar{e}]} \tag{3.20}$$

for lag h and $\bar{e} = \frac{1}{N_c}\sum_{0=1}^{N_c} e_0$, for e_0 the errors defined in equation 3.17.

The comb-p method utilizes the idea of the ACF to combine p-values from the different loci when using the Stouffer-Liptak-Kechris (SLK) correction [Pedersen et al., 2012, Kechris et al., 2010]. Kechris et al. (2010) warn that use of the ACF requires the assumption that the CpG sites be equally spaced [Kechris et al., 2010]. The authors accept this assumption when performing the SLK correction, but explain that it does not hold regularly across the genome. As a result, they have to drop loci that are neighboring large gaps from the calculation. The correlation r_{0j} does not require this assumption, and can therefore utilize all loci regardless of whether the spacing is regular or not.

A formal investigation using GCV, results provided later, found that using r_{0j} for the correlation adjustment was better than $r(|t_0 - t_j|)$ on the Autism data. This is likely because the correlation r_{0j} is penalizing points based their biological correlation with the sample at the target loci. Conversely, $r(|t_0 - t_j|)$ is an estimate of the correlation function $\rho(|t_0 - t_j|)$, which is stationary across the genome, and depends only on the distance between the loci and not on their actual values. When using a stationary correlation function, points that are a certain distance from the target loci have the same correlation as points that are the same distance from another target loci. For example, all points that are 1 CpG site away from any target loci will have the same correlation, and would receive the same weight, regardless of which target point is being estimated or the biological variability in how correlated those samples actually are with the sample at the target loci. This is not the desired behavior, as

will be shown using GCV later.

This idea is very similar to how Bumphunter allows the error variance in model (3.1) to be different at each location t_j due to biological variation [Jaffe et al., 2012], but when smoothing with the loess smoother across the genome in model 3.12, they assume the error process is stationary with homoscedastic variance [Cleveland, 1979]. The same idea is being utilized here.

Obviously this formulation for the correlation-adjusted kernel weight $\frac{1}{|r_{0j}|}K_\lambda(t_0,t_j)$ or $\frac{1}{{r_{0j}}^2}K_\lambda(t_0,t_j)$ cannot allow zero correlations, because the correlation term is in the denominator. However, there is also an adjustment made for very low correlations as well. This is because small correlation values will cause the weight to become extremely inflated. In order to limit the degree of inflation for low correlations, a lower bound is placed on the correlations when determining the weight, which would consider all correlations lower than this boundary the same. This would prevent any weight from becoming extremely inflated. The correlation, including the lower bound, is now formally defined using Spearman's rank correlation coefficient as

$$r_{0j} = \begin{cases} {r_{0j}}^S, & if\ |{r_{0j}}^S| > 0.05 \\ 0.05, & if\ |{r_{0j}}^S| \leq 0.05 \end{cases} \tag{3.21}$$

Figure (1) presents the plot used in the visual investigation of options for the lower bound. The x-axis is for different values of the correlation, and the y-axis is for the weight $\frac{1}{|r_{0j}|}$. The lower bound was initially set to be 0.02 because the weight visually appears to become more seriously inflated with correlations lower than that on this plot. However a more formal investigation using GCV with results provided later, identified the optimal

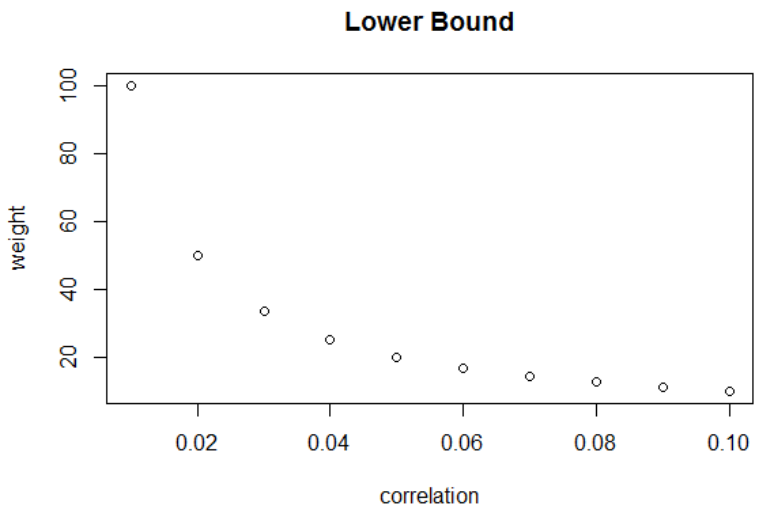

Figure 1: Visual Investigation of Lower Bound

lower bound to be 0.05, so the value from the more formal investigation will be used in the definition here.

The kernel $K_\lambda(t_0, t_j)$ in the correlation-adjusted kernel weight works as usual, putting more weight on points that are close to the target loci than those that are further away. However because the weight assigned by the kernel is then divided by the correlation with the sample at the target loci, the correlation-adjusted weights for samples with low correlations are much higher than the weights for samples with high correlations. As a result, the effect of the correlated points on the fitting of the target point is reduced, similarly to how a bimodal kernel works.

This is a much more targeted approach than using a bimodal kernel however, because it weights the points according to both their genomic distance from the target loci and the correlation with the sample at the target loci, while the bimodal kernel weights points solely according to their genomic distance from the target loci. The weights from a bimodal kernel

are reduced for all of the close points, regardless of whether they are correlated or not. In the new method, since the weights are assigned based on both the distance and correlation, the close points that come from a sample that is not highly correlated can still be heavily utilized.

Also for the bimodal kernel, while the weight is reduced on the short-range dependent points because they are close to the target loci, highly correlated points at moderate to far distances are still weighted the same as points with low correlations at the same distance. So the effect of all potentially correlated points is not being addressed by a bimodal kernel, only the ones that are close to the target loci. In the new method, the effect of all points will be adjusted by their correlation with the sample at the target loci, regardless of the their genomic distance. As a result, the correlation-adjusted kernel weight can address both short-range and long-range dependence problems.

Correlation-Adjusted Nadaraya-Watson Estimator

The correlation-adjusted Nadaraya-Watson estimator for the smoothed y-value $\tilde{f}(t_0)$ can now be calculated for the model in equation (3.18) at each target loci t_0, with each loci $t_1, ..., t_{N_c}$ having the opportunity to be the target loci once, as follows. Let

$$\tilde{f}(t_0)_{CorrNW} = \frac{\sum_{j=1}^{N_c} W(t_0, t_j)\hat{f}(t_j)}{\sum_{j=1}^{N_c} W(t_0, t_j)} \tag{3.22}$$

$$W(t_0, t_j) = \frac{1}{{r_{0j}}^2} K_\lambda(t_0, t_j)$$

where $\frac{1}{{r_{0j}}^2}K_\lambda(t_0,t_j)$ is the correlation-adjusted kernel weight defined in equation (3.19). The weights assigned to the $\hat{f}(t_j)$ values defined in equation (3.7) are now based on both the genomic distance $|t_0 - t_j|$ from the target loci, and the correlation with the sample at the target loci r_{0j}. The weights still die off smoothly as the genomic distance from the target loci increases as usual, but are then adjusted for correlation using r_{0j}.

Note that the correlation-adjusted Nadaraya-Watson estimator is also a linear fitting method. Recall that a linear fitting method can be factored into two parts, the $\hat{f}(t_j)$ and weights defined as $l_j(t_0)$ that do not involve the $\hat{f}(t_j)$ [Hastie et al., 2009]. Let

$$\begin{aligned}\tilde{f}(t_0)_{CorrNW} &= \sum_{j=1}^{N_c}\left[\frac{\frac{1}{{r_{0j}}^2}K_\lambda(t_0,t_j)}{\sum_{j=1}^{N_c}\frac{1}{{r_{0j}}^2}K_\lambda(t_0,t_j)}\right]\hat{f}(t_j)\\ &= \sum_{j=1}^{N_c} l_j(t_0)\hat{f}(t_j)\end{aligned}$$

where $l_j(t_0) = \frac{\frac{1}{{r_{0j}}^2}K_\lambda(t_0,t_j)}{\sum_{j=1}^{N_c}\frac{1}{{r_{0j}}^2}K_\lambda(t_0,t_j)}$. In vector form using all N loci in cluster c that are smoothed together, this implies

$$\tilde{\boldsymbol{f}} = \boldsymbol{S}_c\hat{\boldsymbol{f}} \tag{3.23}$$

in exactly the same way that was explained in equation (3.14). Also note that the weights for the correlation-adjusted Nadaraya-Watson $l_j(t_0)$ sum to 1, as in

$$\sum_{j=1}^{N_c} l_j(t_0) = \sum_{j=1}^{N_c}\left[\frac{\frac{1}{{r_{0j}}^2}K_\lambda(t_0,t_j)}{\sum_{j=1}^{N_c}\frac{1}{{r_{0j}}^2}K_\lambda(t_0,t_j)}\right] = \left[= \frac{\sum_{j=1}^{N_c}\frac{1}{{r_{0j}}^2}K_\lambda(t_0,t_j)}{\sum_{j=1}^{N_c}\frac{1}{{r_{0j}}^2}K_\lambda(t_0,t_j)}\right] = 1$$

Correlation-Adjusted Local Linear Estimator

The correlation-adjusted local linear estimator can also be defined for the model in equation (3.18). On this dataset, however, it needs to be fit without the intercept term. This is because the matrix $(\boldsymbol{B}^T\boldsymbol{W}(t_0)\boldsymbol{B})$ was not invertible. After investigation, it was determined that removing the intercept in $\boldsymbol{B}$ eliminated this problem. This is likely because of a correlation between the weights in $\boldsymbol{W}(t_0)$ and the intercept in $\boldsymbol{B}$ due to the correlation adjustment.

When the matrix is less than full rank, the estimated coefficients can be imprecise with high variance [Hastie et al., 2009]. There are many solutions to this problem, with none being guaranteed to be the best. Using a penalized regression like ridge or lasso regression would reduce the variance, which could result in a better fit, but we would need to remove the intercept in that case anyway. Therefore the proposed solution to this problem here is to simply remove the intercept. This makes sense because $f(t) = |\beta(t)| = 0$ for most of the genome [Jaffe et al., 2012], so the intercept is likely very near zero at each target point. Therefore removing the intercept seems like a reasonable thing to do, but this choice will be revisited more formally later.

Locally weighted linear regression solves the following weighted least squares problem

$$\min_{\alpha_2(t_0)} \sum_{j=1}^{N_c} \frac{1}{{r_{0j}}^2} K_\lambda(t_0, t_j)[f(t_j) - \alpha_2(t_0)t_j]^2$$

where

$$\tilde{f}(t_0)_{CorrLL} = \hat{\alpha_2}(t_0)t_0 = t_0(\boldsymbol{t}^T\boldsymbol{W}(t_0)\boldsymbol{t})^{-1}\boldsymbol{t}^T\boldsymbol{W}(t_0)\hat{\boldsymbol{f}} \tag{3.24}$$

$$\boldsymbol{W}(t_0) = diag\left[\frac{1}{{r_{0j}}^2}K_\lambda(t_0,t_j)\right]$$

for target loci t_0, which is a member of N_cx1 vector of loci $\boldsymbol{t}$, and $\boldsymbol{W}(t_0)$ the N_cxN_c diagonal matrix with jth diagonal element $\frac{1}{{r_{0j}}^2}K_\lambda(t_0,t_j)$, which is the correlation-adjusted kernel weight defined in equation (3.19), and $\hat{\boldsymbol{f}} = (\hat{f}(t_1),...,\hat{f}(t_{N_c}))'$, the initial fit statistics as defined in equation (3.7). The matrix $\boldsymbol{W}(t_0)$ is now comprised of weights $\frac{1}{{r_{0j}}^2}K_\lambda(t_0,t_j)$ that are based on both the genomic distance $|t_0 - t_j|$ from the target loci, and the correlation with the sample at the target loci r_{0j}. The weights still die off smoothly as the genomic distance from the target loci increases as usual, but are then adjusted for correlation using r_{0j}.

Note that the correlation-adjusted local linear estimator is also a linear fitting method, as in

$$\begin{aligned}\tilde{f}(t_0)_{CorrLL} &= t_0(\boldsymbol{t}^T\boldsymbol{W}(t_0)\boldsymbol{t})^{-1}\boldsymbol{t}^T\boldsymbol{W}(t_0)\hat{\boldsymbol{f}} \\ &= \sum_{j=1}^{N_c} l_j(t_0)\hat{f}(t_j)\end{aligned}$$

Note that this implies

$$\tilde{\boldsymbol{f}} = \boldsymbol{S}_c\hat{\boldsymbol{f}} \tag{3.25}$$

in exactly the same way that was explained in equation (3.14). Also note that the weights $l_j(t_0)$ in locally weighted least squares have been shown to sum to 1 regardless of what weight is actually used [Liu, 2001].

In a formal investigation using GCV, the correlation-adjusted Nadaraya-Watson estimator was found to perform better than the correlation-adjusted local linear estimator when using the new method. So that will be used as part of the formal definition here. GCV results will be provided later.

3.4.5 Generalized Cross Validation

As mentioned previously, since the correlation-adjusted kernel weight reduces the effect of the correlated points on the fitting of the target point like the bimodal kernel, the usual cross validation procedures like leave-one-out CV can be used to determine the optimal tuning parameter [De Brabanter et al., 2011]. However instead of following the paper on bimodal kernels and using leave-one-out cross validation, generalized cross validation (GCV) will be used instead.

As mentioned, GCV was developed as an alternative to the leave-one-out cross validation procedure [Craven and Wahba, 1977]. It has been shown to be nearly unbiased for ESPE in the case of unequally spaced design points, if the design points are considered to be fixed [Altman, 1990, Craven and Wahba, 1977]. This is the current situation so this is ideal. The design points are the x-values of the points $[t_0, \hat{f}(t_0)]$ needing to be smoothed, which are the CpG sites in the case of methylation data. The CpG sites are not equally spaced along the genome. However they can be considered fixed, because their position is not random, they are in the same place each time the genome is visited. GCV can also alleviate the tendency of usual cross validation procedures to undersmooth, and can be computationally more efficient, which is always a concern with genomic data

[Hastie et al., 2009]. It is for these reasons that it was selected for use here over leave-one-out CV. The original GCV paper seems to suggest it is only appropriate for data on [0,1], however another paper published with one of the original authors demonstrates its use without this assumption [Golub et al., 1979]. Additionally, other authors have used this interval and explicitly stated it was for simplicity [Opsomer et al., 2001].

As mentioned earlier, GCV is appropriate for linear fitting methods [Hastie et al., 2009, Craven and Wahba, 1977]. Recall that linear fitting methods can be written as $\tilde{\boldsymbol{f}} = \boldsymbol{S}_c\hat{\boldsymbol{f}}$, for N_cxN_c matrix $\boldsymbol{S}_c$ that does not include the $\hat{\boldsymbol{f}}$ [Hastie et al., 2009]. All of the above methods are linear fitting methods as defined in equations (3.14),(3.16),(3.23), and (3.25). The GCV estimate of expected mean squared prediction error can then be defined as

$$GCV(\tilde{f})_{EMSPE} = \frac{1}{N_c}\sum_{j=1}^{N_c}\left[\frac{\hat{f}(t_j)-\tilde{f}(t_j)}{1-tr(\boldsymbol{S}_c)/N_c}\right]^2$$

where $tr(\boldsymbol{S}_c)$ is the trace of $\boldsymbol{S}_c$, and is calculated for each cluster, $c = 1,...,C$.

Introduction to GCV investigations for smoothing step

As an initial look, GCV was used to determine optimal tuning parameters and compare methods for the smoothing step. Recall that the idea for the correlation-adjusted kernel weight began as $\frac{1}{|r_{0j}|}K_\lambda(t_0,t_j)$ mentioned previously, using $\frac{1}{|r_{0j}|}$ for the correlation adjustment. Then it was determined that increasing the exponent to 2 was better, resulting in $\frac{1}{{r_{0j}}^2}K_\lambda(t_0,t_j)$, using $\frac{1}{{r_{0j}}^2}$ for the correlation adjustment. So the investigation presented here began by using $\frac{1}{|r_{0j}|}$ for the adjustment, and will end with how $\frac{1}{{r_{0j}}^2}$ was determined to be better.

GCV investigation into best lower bound

Table I presents GCV estimates of expected mean squared prediction error for different options for the lower bound placed on the definition of the correlation in equation (3.21). This investigation began using the Gaussian kernel defined in equation (3.10), the correlation-adjusted Nadaraya-Watson estimator defined in equation (3.22), and the Pearson definition of r_{0j} in $\frac{1}{|r_{0j}|}$ for the correlation adjustment. From this table we can see that a lower bound of 0.05 resulted in the lowest error, so that it why it is selected for use as the lower bound. However all of the errors are very similar, which implies this is not a critical choice. The lower bound was set to 0.05 for the rest of this investigation.

Table I: GCV Investigation of Lower Bound

Lower Bound	Optimal λ	GCV_{EMSPE}	Best one
0.02	125	0.08746464	
0.03	125	0.08723752	
0.04	125	0.08709065	
0.05	125	0.08698906	X
0.06	125	0.08700038	
0.07	125	0.09033244	

GCV investigation into best estimator, kernel, and correlation definition

Table II presents the GCV estimate of expected mean squared prediction error for all of the above mentioned estimators using the Gaussian kernel, with Pearson and Spearman

definitions of r_{0j} in $\frac{1}{|r_{0j}|}$ for the correlation adjustment. From this table we can see that the estimators using the Spearman $\frac{1}{|r_{0j}|}$ give the lowest errors. Among the estimators using the Spearman $\frac{1}{|r_{0j}|}$, the best estimator is the correlation-adjusted Nadaraya-Watson defined in equation (3.22). However the error for the correlation-adjusted local linear defined in equation (3.24) is only slightly higher, even with the questionable issue of removing the intercept. Note that for the usual estimators defined in equations (3.13) and (3.15), the local linear estimator is slightly better than the Nadaraya-Watson. This is evidence that the intercept is likely very close to 0 at each target point, and that removing it appears to be an acceptable choice as the result is not drastically different from the Nadaraya-Watson.

Table II: GCV Estimates using Gaussian kernel

Correlation	Estimator	Optimal λ	GCV_{EMSPE}	Best one
None	Nadaraya-Watson (NW)	49	0.1242834	
	Local Linear (LL)	49	0.1242795	
Pearson $\frac{1}{\lvert r_{0j}\rvert}$	Correlation-Adjusted NW	125	0.08698906	
	Correlation-Adjusted LL	125	0.08699095	
Spearman $\frac{1}{\lvert r_{0j}\rvert}$	Correlation-Adjusted NW	143	0.08445823	X
	Correlation-Adjusted LL	143	0.08446300	

Table III presents the GCV estimate of expected mean squared prediction error for all of the above mentioned estimators using the Epanechnikov kernel defined in equation (3.11), with Pearson and Spearman definitions of r_{0j} in $\frac{1}{|r_{0j}|}$ for the correlation adjustment. From this table we can see that the estimators using the Pearson $\frac{1}{|r_{0j}|}$ give the lowest

errors. Among the estimators using the Pearson $\frac{1}{|r_{0j}|}$, the best estimator is the correlation-adjusted local linear defined in equation (3.24). This gives further support that removing the intercept is an acceptable choice. When comparing this table to Table II, we can see that for the usual estimators, the Epanechnikov kernel resulted in slightly lower errors than the Gaussian. However for the correlation-adjusted estimators, the errors when using the Gaussian kernel were lower. So that is why the Gaussian kernel was initially selected for use in the method instead of the Epanechnikov.

Table III: GCV Estimates using Epanechnikov kernel

Correlation	Estimator	Optimal λ	GCV_{EMSPE}	Best one
None	Nadaraya-Watson (NW)	93	0.1216161	
	Local Linear (LL)	93	0.1216098	
Pearson $\frac{1}{\lvert r_{0j}\rvert}$	Correlation-Adjusted NW	110	0.09889429	
	Correlation-Adjusted LL	110	0.09889212	X
Spearman $\frac{1}{\lvert r_{0j}\rvert}$	Correlation-Adjusted NW	110	0.10013790	
	Correlation-Adjusted LL	110	0.10013550	

Table IV presents the GCV estimate of expected mean squared prediction error for all of the above mentioned estimators using the Gaussian kernel and $\frac{1}{r(|t_0-t_j|)}$ for the correlation adjustment defined in equation (3.20). From this table we can see that the best estimator is again the correlation-adjusted local linear, although it is only slightly better than the correlation-adjusted Nadaraya-Watson. However when comparing this table to Table II, we can see that using $\frac{1}{|r_{0j}|}$ as the correlation adjustment resulted in lower errors than when using

$\frac{1}{r(|t_0-t_j|)}$. This supports the idea that penalizing points based on their biological correlation with the sample at the target loci, r_{0j}, is more effective than using the estimate of the correlation function $r(|t_0-t_j|)$. So that is why $\frac{1}{|r_{0j}|}$ was selected for use as the correlation adjustment instead of $\frac{1}{r(|t_0-t_j|)}$.

Table IV: GCV Estimates using Gaussian kernel and $\frac{1}{r(|t_0-t_j|)}$

Estimator	Optimal λ	GCV_{EMSPE}	Best one
Correlation-Adjusted NW	56	0.1066691	
Correlation-Adjusted LL	56	0.1066674	X

Table V presents an investigation into the lower bound used for the correlation-adjusted kernel weight when using $\frac{1}{r(|t_0-t_j|)}$ and correlation-adjusted local linear estimator. From this table we can see that the ACF is much less sensitive to the choice of lower bound. All choices of the lower bound gave the same error.

Preliminary Summary 1

From the analysis above we can see that the best estimator so far is in Table II. It is the correlation-adjusted Nadaraya-Watson estimator defined in equation (3.22), using the Gaussian kernel, and the Spearman definition of r_{0j} in $\frac{1}{r_{0j}}$ for the correlation adjustment.

GCV investigation into best exponent on correlation-adjusted weight

Table VI presents the behavior of the Gaussian kernel for hypothetical values of distance and correlation using the optimal tuning parameters for the best estimator from preliminary summary 1 and the usual Nadaraya-Watson estimator from Table II. We can see that the

Table V: GCV Investigation of Lower Bound when using $\frac{1}{r(|t_0-t_j|)}$

Lower Bound	Optimal λ	GCV_{EMSPE}	Best one
0.0001	56	0.1066674	X
0.001	56	0.1066674	X
0.01	56	0.1066674	X
0.02	56	0.1066674	X
0.03	56	0.1066674	X
0.04	56	0.1066674	X
0.05	56	0.1066674	X
0.06	56	0.1066674	X
0.10	56	0.1066674	X
0.20	56	0.1066674	X
0.30	56	0.1066674	X

weights from both methods die off with the distance as usual. However the Gaussian kernel cannot tell the difference between the highly correlated points and the less correlated points, assigning the same weight to points at the same distance. The correlation-adjusted kernel weight, on the other hand, weights the highly correlated points much lower than the less correlated points. For example in Table VI, for points that are 1 base pair away from the target loci, the weight for a highly correlated point (0.44) is about 11 percent of the weight for a less correlated point (3.99) at the same distance. As a result, the effect of the correlated points on the fitting of the target point is reduced. Next it was investigated whether an 11

percent reduction was enough.

Table VI: Behavior of Gaussian kernel using $\frac{1}{|r_{0j}|}$

Distance	Correlation r_{0j}	$Gaussian_{\lambda=49}$	Corr-Adj $Gaussian_{\lambda=143}$ with $\frac{1}{|r_{0j}|}$
$\pm 1bp$	± 0.1	0.40	3.99
$\pm 1bp$	± 0.9	0.40	0.44
$\pm 5bp$	± 0.1	0.40	3.99
$\pm 5bp$	± 0.9	0.40	0.44
$\pm 50bp$	± 0.1	0.24	3.76
$\pm 50bp$	± 0.9	0.24	0.42
$\pm 100bp$	± 0.1	0.05	3.12
$\pm 100bp$	± 0.9	0.05	0.35
$\pm 200bp$	± 0.1	0	1.50
$\pm 200bp$	± 0.9	0	0.17
Quality		Poor	Good

Table VII presents an investigation into the exponent used for the correlation-adjusted kernel weight for the best estimator from preliminary summary 1. Increasing the exponent should help reduce the effect of the correlated points even more. From this table we can see that an exponent of 2 had the lowest error. Since the correlation-adjusted kernel weight $\frac{1}{r_{0j}{}^2}$ resulted in the lowest GCV error of all estimators tried in this investigation, this is the best weight to use for the correlation adjustment. Increasing the exponent any more than that provides no additional benefit.

Table VII: GCV estimates for different exponents on correlation-adjusted weight

Corr-Adj weight	Optimal λ	GCV_{EMSPE}	Best one
$\frac{1}{\lvert r_{0j}\rvert}$	143	0.08445823	
$\frac{1}{{r_{0j}}^2}$	122	0.08306051	X
$\frac{1}{\lvert r_{0j}\rvert^3}$	106	0.08365947	
$\frac{1}{{r_{0j}}^4}$	94	0.08330751	
$\frac{1}{\lvert r_{0j}\rvert^5}$	84	0.08310439	
$\frac{1}{{r_{0j}}^6}$	76	0.08317052	

In Table VIII, the behavior of the Gaussian kernel is again presented for hypothetical values of distance and correlation, this time using the optimal tuning parameter for $\frac{1}{{r_{0j}}^2}$ in Table VII. Now we can see that for points that are 1 base pair away from the target loci, the weight for a highly correlated point (0.49) is about 1 percent of the weight for a less correlated point (39.89) at the same distance. As a result, the effect of the correlated points on the fitting of the target point is severely reduced. Note that a slightly lower tuning parameter was also selected.

Preliminary Summary 2

From the analysis above we can see that the best estimator of all is in Table VII. It is the correlation-adjusted Nadaraya-Watson estimator defined in equation (3.22), using the Gaussian kernel, and the Spearman definition of r_{0j} in $\frac{1}{{r_{0j}}^2}$ for the correlation adjustment.

Table VIII: Behavior of Gaussian kernel using $\frac{1}{r_{0j}{}^2}$

Distance	Correlation r_{0j}	$Gaussian_{\lambda=49}$	Corr-Adj $Gaussian_{\lambda=122}$ with $\frac{1}{r_{0j}{}^2}$
$\pm 1bp$	± 0.1	0.40	39.89
$\pm 1bp$	± 0.9	0.40	0.49
$\pm 5bp$	± 0.1	0.40	39.86
$\pm 5bp$	± 0.9	0.40	0.49
$\pm 50bp$	± 0.1	0.24	36.68
$\pm 50bp$	± 0.9	0.24	0.45
$\pm 100bp$	± 0.1	0.05	28.51
$\pm 100bp$	± 0.9	0.05	0.35
$\pm 200bp$	± 0.1	0	10.41
$\pm 200bp$	± 0.9	0	0.13
Quality		Poor	Excellent

Plots of Smoothing Behavior

Figure 2 is a short range plot comparing the smoothed values for the best estimator from preliminary summary 1 that used $\frac{1}{|r_{0j}|}$ for the correlation adjustment, to the usual Nadaraya-Watson estimator from Table II. Figure 3 is a short range plot comparing the best estimator from preliminary summary 2 that used $\frac{1}{r_{0j}{}^2}$ for the correlation adjustment, to the usual Nadaraya-Watson from Table II. Since the GCV error when using $\frac{1}{r_{0j}{}^2}$ was the lowest of all, this is evidence that this is the best plot for all options tried.

Figure 4 is a long range plot comparing the best estimator from preliminary summary 2

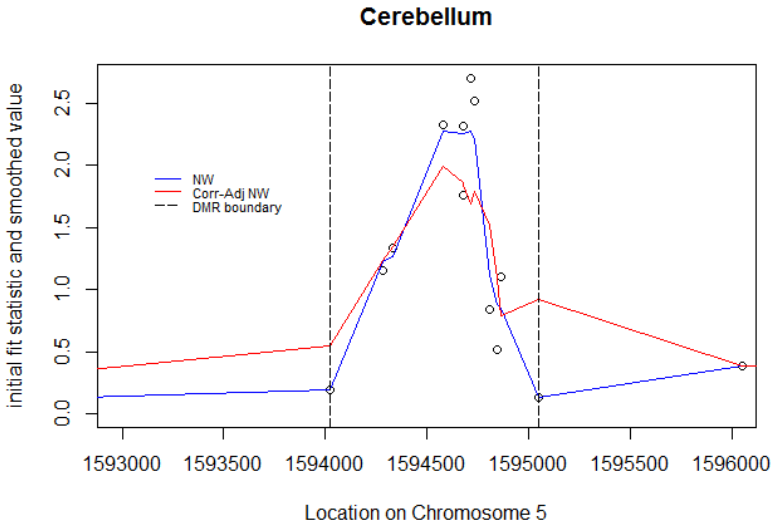

Figure 2: Close-range Smoother Comparison using $\frac{1}{|r_{0j}|}$

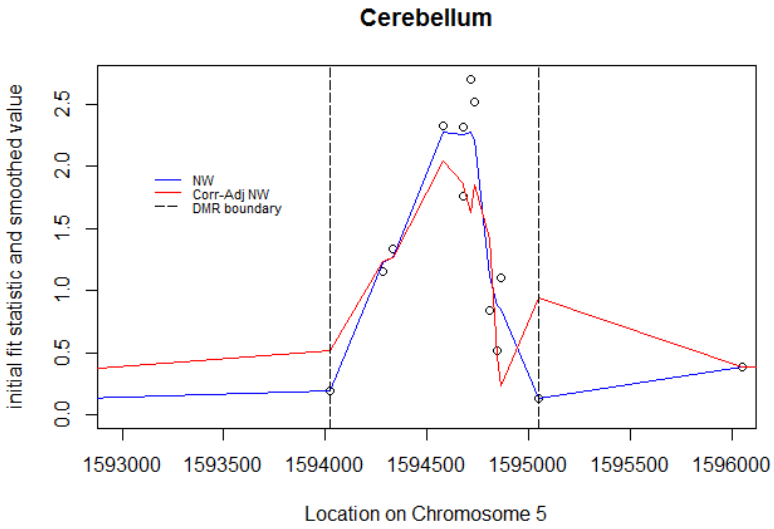

Figure 3: Close-range Smoother Comparison using $\frac{1}{r_{0j}{}^2}$

that used the Gaussian kernel and $\frac{1}{{r_{0j}}^2}$ for the correlation adjustment, to the usual Nadaraya-Watson with the Gaussian kernel from Table II. From this plot we can see that, when using the Gaussian kernel, the correlation adjusted Nadaraya-Watson estimator appears less sensitive to single CpG sites with high values than the usual Nadaraya-Watson, because it is the blue line that tends to reach up to estimate the smoothed value at a higher point than the red line.

Figure 5 is a long range plot comparing the best estimator from preliminary summary 2 that used $\frac{1}{{r_{0j}}^2}$ for the correlation adjustment when using the Epanechnikov kernel instead of the Gaussian, to the usual Nadaraya-Watson from Table III that also uses the Epanechnikov. From this plot we can see that the usual Nadaraya-Watson estimator appears less sensitive to single CpG sites with high values than the correlation-adjusted Nadaraya-Watson, because it is the red line that tends to reach up to estimate the smoothed value at a higher point than the blue line. This suggests that while the Epanechnikov kernel may afford more computational efficiency as mentioned before [Cleveland, 1979], the Gaussian kernel may be more able to detect DMRs with smaller effect sizes. Since it is less sensitive to single CpG sites with high values, it may be more able to detect a consistent region with smaller effect sizes than the Epanechnikov. However simulation results showed that the Epanechnikov was actually slightly better than the Gaussian at detecting small effect sizes, so while the Gaussian kernel was used in the initial version of the method, the Epanechnikov was used in the final version of the method.

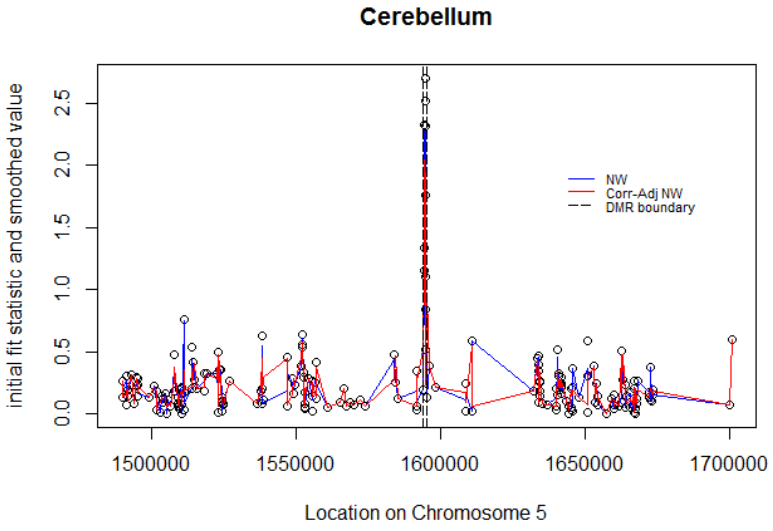

Figure 4: Long-range Smoother Comparison using Gaussian Kernel

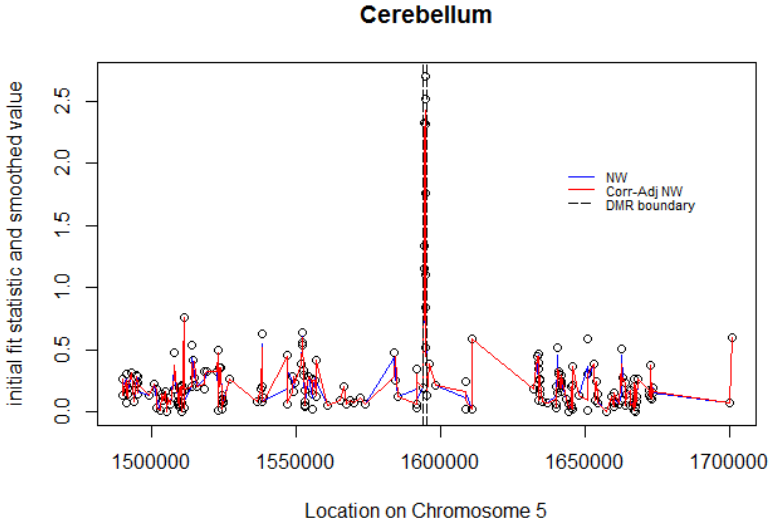

Figure 5: Long-range Smoother Comparison using Epanechnikov Kernel

3.5 Defining Regions of Interest

The last modification DMR Detector makes to the Bumphunter algorithm is related to how to define regions of interest. The smoothed values $\tilde{f}(t_j)$, $j = 1,...,J$ are collected, and regions of interest $\hat{R}_g$, $g = 1,..,\hat{G}$ are defined as follows. Bumphunter defines regions to be where at least 2 adjacent CpG sites have a smoothed value that is above some user-specified quantile cutoff (cutoffQ) of the smoothed values obtained from a null distribution generated using a bootstrap procedure that will be discussed below. The authors of Bumphunter investigated the differences between using the 95th, 99th, and 99.9th quantile for this cutoff, and found that the 99th quantile worked best for that method. DMR Detector also allows users to choose this cutoffQ, however whole genome simulations will be needed in order to truly assess the optimal value for DMR Detector, because small scale simulations were run using 25 CpG sites and imposing such a strict cutoff on so few CpG sites is not the same as for the whole genome.

However DMR Detector differs from Bumphunter by defining regions as not necessarily adjacent CpG sites, but rather as a region where at least 75 percent of the CpG sites are above the cutoffQ, but not necessarily adjacent to each other on the genome. From the literature review up to this point, all existing methods on the 450K array use adjacent CpG sites only, so this is a new definition of regions of interest for the 450K array.

The size of each region of interest is then calculated using a measure similar to the ”area” defined by Bumphunter [Jaffe et al., 2012]. Let area

$$A_g = \sum_{j \in \hat{R}_g} \tilde{f}(t_j) \tag{3.26}$$

for $g = 1, .., \hat{G}$.

3.6 Assigning Significance to Regions

Next, significance is assigned to the areas using a bootstrap procedure including adjustment for multiple testing. Bootstrapping retains all natural characteristics of the data, such as the presence of batch effects and correlated errors [Jaffe et al., 2012]. The bootstrap procedure is repeated many times, which generates a distribution of candidate regions. All regions identified in these bootstrap samples can be considered ”null” candidate regions occurring by chance.

Running the method B times produces the set of null areas $A^*_{g,b}$, $g = 1, .., \hat{G}^*_b$, $b = 1, ..., B$ representing an approximate null distribution of the areas. Let A^* represent the random variable corresponding to this distribution of null areas. Then empirical p-values are defined for each area as

$$p_g = P[A^* \geq A_g] \tag{3.27}$$

for A_g defined in equation (3.26) for $g = 1, .., \hat{G}$. There is then an adjustment for multiple testing using the false discovery rate (FDR) [Benjamini and Hochberg, 1995].

Benjamini and Hochberg (1995) proposed a method for controlling the false discovery rate (FDR) as an alternative to classical multiple comparison procedures for controlling the family wise error rate (FWER) [Benjamini and Hochberg, 1995]. The FWER is defined

as the probability of committing at least one Type I error. The FDR is the expected proportion of falsely rejected hypotheses. The authors explain that FWER control is not always needed. They say in some experiments, control of the probability of any error may be unnecessarily strict, because a small proportion of errors will not change the overall validity of the conclusion. Storey's optimal discovery procedure then determines the minimum FDR at which each area can be called significant, referred to as the q-value [Storey and Tibshirani, 2003, Storey, 2007]. This procedure allows for dependence among the p-values.

The FDR is defined as

$$FDR = E\left(\frac{F}{F+T}\right) = E\left(\frac{F}{S}\right)$$

where F is the number of false positives, T is the number of true positives, and $S = F + T$ [Storey and Tibshirani, 2003]. The q-value can be defined as the expected proportion of false positives incurred when calling a particular area significant. Thresholding q-values at level α produces a set of significant p-values such that the proportion α is expected to be false positives. Therefore we can also describe the q-value as the expected proportion of false positives among all p-values as or more extreme than the observed one.

The estimation of the FDR begins with the m p-values that are less than or equal to some threshold t, $0 < t \leq 1$. Denote the m p-values as $p_1, ..., p_m$ and let

$$F(t) = \#(null\ p_i \leq t; i = 1, ..., m)$$

and

$$S(t) = \#(p_i \leq t; i = 1, ..., m)$$

We want to estimate

$$FDR(t) = E\left(\frac{F(t)}{S(t)}\right)$$

Because we are estimating for many areas, m is considered to be large. This implies that

$$FDR(t) = E\left(\frac{F(t)}{S(t)}\right) \approx \left(\frac{EF(t)}{ES(t)}\right) \tag{3.28}$$

An estimate of $ES(t)$ is the observed $S(t)$, in other words, the number of observed p-values $\leq t$. Since p-values corresponding to truly null hypotheses are assumed to be uniformly distributed, this implies that the probability a null p-value is $\leq t$ is t. This implies that $EF(t) = m_0 t$, where m_0 is the number of truly null hypotheses, and that $\pi_0 = \frac{m_0}{m}$ is the proportion of hypotheses that are truly null, both of which are unknown and need to be estimated. The proportion can be estimated as

$$\hat{\pi}_0(\lambda) = \frac{\#(p_i > \lambda; i = 1, ..., m)}{m(1-\lambda)}$$

for tuning parameter λ. The authors suggest using $\lambda = 0.5$, but also suggest an automated procedure which will be discussed below.

Then, by substitution into equation (3.28), this implies that

$$\hat{FDR}(t) = \frac{\hat{\pi}_0 mt}{S(t)} = \frac{\hat{\pi}_0 mt}{\#(p_i \leq t)}$$

This leads us to the more mathematical definition of q-value, that it is the minimum FDR that can be attained when calling a particular area significant. This implies

$$\hat{q}(p_i) = min_{p_i \leq t} F\hat{D}R(t)$$

The authors also suggest an automated procedure to choose the tuning parameter λ in $\hat{\pi}_0(\lambda)$. This procedure is as follows.

1. Let $p_{(1)} \leq p_{(2)} \leq ... \leq p_{(m)}$ be the ordered p-values.

2. For a range of values for λ, such as $\lambda = 0, 0.01, 0.02, ..., 0.95$, calculate

$$\hat{\pi}_0(\lambda) = \frac{\#(p_j > \lambda)}{m(1-\lambda)}$$

3. Let $\hat{f}$ be the natural cubic spline with 3 df of $\hat{\pi}_0(\lambda)$ on λ.

4. Set the estimate of π_0 to be

$$\hat{\pi}_0 = \hat{f}(1)$$

5. Calculate

$$\hat{q}(p_{(m)}) = min_{p_m \leq t} \frac{\hat{\pi}_0 mt}{\#(p_j \leq t)} = \hat{\pi}_0 p_{(m)}$$

6. For $i = m = 1, m-2, ..., 1$, calculate

$$\hat{q}(p_{(i)}) = min_{p_i \leq t} \frac{\hat{\pi}_0 mt}{\#(p_j \leq t)} = min\left(\frac{\hat{\pi}_0 m p_{(i)}}{i}, \hat{q}(p_{(i+1)})\right)$$

7. The estimated q-value for the ith most significant area is $\hat{q}(p_{(i)})$.

The q-values are then thresholded at level α to produce a set of significant DMRs such that the proportion α is expected to be false positives. These are the bumps the algorithm is searching for. Significant DMRs are then ranked according to their areas from largest to smallest and presented along with their q-values. This enables users to easily find the most significant DMRs, and to see how they relate to each other.

3.7 Final Methodology and Implementation Algorithm

The final algorithm will be implemented according to the following steps: 1. - 12.

1. Estimate any unmeasured confounders using SVA such that model (3.1) equals model (3.2), as in

$$\begin{aligned} Y_{ij} &= \mu(t_j) + \beta(t_j)V_i + \sum_{d=1}^{D} \phi_d(t_j)Z_{id} + \sum_{l=1}^{L} \gamma_{lj}U_{il} + \varepsilon_{ij} \\ &= \mu(t_j) + \beta(t_j)V_i + \sum_{d=1}^{D} \phi_d(t_j)Z_{id} + \sum_{k=1}^{K} \xi_{kj}h_{ik} + \varepsilon_{ij} \end{aligned}$$

creating model (3.3) defined as

$$Y_{ij} = \mu(t_j) + \beta(t_j)V_i + \sum_{d=1}^{D} \phi_d(t_j)Z_{id} + \sum_{k=1}^{K} \xi_{kj}\hat{h}_{ik} + \varepsilon_{ij}$$

which can be written as model (3.4) defined as

$$\boldsymbol{y}_j = \boldsymbol{X}\boldsymbol{\theta}_j + \boldsymbol{\varepsilon}_j$$

at each genomic location $j = 1, ..., J$.

2. Calculate the statistic $\hat{\beta}(t_j)$ for the effect of the variable of interest using the .getEstimate function in Bumphunter.

3. Collect the absolute value of that statistic defined as

$$\hat{f}(t_j) = |\hat{\beta}(t_j)|$$

4. Perform kernel smoothing using the Epanechnikov kernel defined in equation (3.11) as

$$K_\lambda(t_0, t) = D\left(\frac{|t - t_0|}{\lambda_c}\right)$$

where

$$D(t) = \begin{cases} \frac{3}{4}(1 - t^2), & if\ |t| \leq 1 \\ 0, & otherwise \end{cases}$$

with the correlation-adjusted kernel weight defined in equation (3.19) as

$$W(t_0, t_j) = \frac{1}{{r_{0j}}^2} K_\lambda(t_0, t_j)$$

with Spearman correlation defined as

$${r_{0j}}^S = \frac{12 \sum_{i=1}^{n} (R_i - \frac{n+1}{2})(S_i - \frac{n+1}{2})}{n(n^2 - 1)}$$

including lower bound defined in equation (3.21) as

$$r_{0j} = \begin{cases} {r_{0j}}^S, & if\ |{r_{0j}}^S| > 0.05 \\ 0.05, & if\ |{r_{0j}}^S| \leq 0.05 \end{cases}$$

within the correlation-adjusted Nadaraya-Watson estimator defined in equation (3.22) as

$$\tilde{f}(t_0) = \frac{\sum_{j=1}^{N_c} W(t_0,t_j)\hat{f}(t_j)}{\sum_{j=1}^{N_c} W(t_0,t_j)}$$

where

$$W(t_0,t_j) = \frac{1}{{r_{0j}}^2} K_\lambda(t_0,t_j)$$

resulting in smoothed values $\tilde{f}(t_j)$, $j = 1,...,J$, where the optimal tuning parameter λ_c will be determined using GCV defined as

$$GCV(\tilde{f})_{EMSPE} = \frac{1}{N_c} \sum_{j=1}^{N_c} \left[\frac{\hat{f}(t_j) - \tilde{f}(t_j)}{1 - tr(\boldsymbol{S}_c)/N_c} \right]^2$$

for $c = 1,...,C$ clusters.

5. Define regions of interest $\hat{R}_g$, $g = 1,..,\hat{G}$ where at least 75 percent of the smoothed CpG sites in the region are above some user-specified quantile threshold of the smoothed values $\tilde{f}(t_j)$, $j = 1,...,J$.

6. Calculate the area of each region defined in equation (3.26) as

$$A_g = \sum_{j \in \hat{R}_g} \tilde{f}(t_j)$$

for $g = 1, .., \hat{G}$.

7. Use bootstrap to produce the set of null areas $A^*_{g,b}$, $g = 1, .., \hat{G}^*_b$, $b = 1, ..., B$.

8. Assign significance by calculating p-values defined in equation (3.27) as

$$p_g = P[A^* \geq A_g]$$

9. Perform adjustment for multiple testing.

   (a) Let $p_{(1)} \leq p_{(2)} \leq ... \leq p_{(m)}$ be the ordered p-values.

   (b) For a range of values for λ, such as $\lambda = 0, 0.01, 0.02, ..., 0.95$, calculate

$$\hat{\pi}_0(\lambda) = \frac{\#(p_j > \lambda)}{m(1-\lambda)}$$

   (c) Let $\hat{f}$ be the natural cubic spline with 3 df of $\hat{\pi}_0(\lambda)$ on λ.

   (d) Set the estimate of π_0 to be

$$\hat{\pi}_0 = \hat{f}(1)$$

   (e) Calculate

$$\hat{q}(p_{(m)}) = min_{p_m \leq t} \frac{\hat{\pi}_0 mt}{\#(p_j \leq t)} = \hat{\pi}_0 p_{(m)}$$

   (f) For $i = m = 1, m-2, ..., 1$, calculate

$$\hat{q}(p_{(i)}) = min_{p_i \leq t} \frac{\hat{\pi}_0 mt}{\#(p_j \leq t)} = min\left(\frac{\hat{\pi}_0 m p_{(i)}}{i}, \hat{q}(p_{(i+1)})\right)$$

(g) The estimated q-value for the *i*th most significant area is $\hat{q}(p_{(i)})$.

10. The q-values are then thresholded at level α to produce a set of significant DMRs such that the proportion α is expected to be false positives.

11. Significant DMRs are then ranked according to their areas from largest to smallest and presented along with their q-values.

4. SIMULATION

DMR Detector and Bumphunter [Jaffe et al., 2012] were applied to simulated data and compared. Small scale simulations were run in our department computer lab. The data was simulated for 25 CpG sites on chromosome 5 according to model (3.1) with similar statistical properties to the Autism data that was analyzed using Bumphunter in a published paper [Ladd-Acosta et al., 2014]. To emulate the observed correlation in the Autism data, an autoregressive, lag 1 [AR(1)] process with coefficient 0.20 and a standard deviation of 0.23 as determined by the "arima" package in R [R Core Team, 2016] was used. Note that these values are very similar to the ones identified in the Bumphunter paper for the CHARM array. The methods were then compared in terms of empirical power and empirical family-wise type I error rate using 1000 Monte Carlo samples.

To assess power, one DMR was simulated for 12 consecutive CpG sites, the same size as the DMR identified on chromosome 5 for the Cerebellum in the Autism paper [Ladd-Acosta et al., 2014]. The group variable from the real dataset was used so that simulated effect sizes $\beta(t_j)$ defined in model (3.1) were realistic, and initially consisted of values between 0.13 and 2.7. Power was calculated as the number of times each method identified the true DMR divided by 1000. To assess family-wise type I error, we generated a null region with no DMRs, and calculated the family-wise type I error rate as the number

of times each method made at least one type I error divided by 1000. Since Bumphunter uses FWER for the multiple testing adjustment and DMR Detector uses FDR, significant DMRs were then identified for Bumphunter using a FWER threshold of 0.1, and for DMR Detector using an FDR threshold of 0.1.

For Bumphunter, the FWER calculated for the areas was used, which corresponds to the same areas as defined in the Bumphunter paper and in DMR Detector, and is specifically called ”fwerArea” in the Bumphunter software. The maxGap setting was initially set at maxGap=1000 to define a variable window size, which means that the loci must be within 1000 base pairs from each other to be considered as part of the same region of interest. As mentioned previously, Bumphunter recommends a smaller size of 300 base pairs for the CHARM array, because this array has a median distance between probes of 36 to 70 base pairs, depending on the array version [Jaffe et al., 2012]. DMRcate recommends 1000 base pairs for the 450K array since the median distance between probes is 300 base pairs [Peters et al., 2015].

For DMR Detector, because the number of p-values needing adjustment for multiple testing in this window was so small, the Storey q-value method [Storey and Tibshirani, 2003, Storey, 2007] gave errors, so the original method for controlling FDR by Benjamini and Hochberg [Benjamini and Hochberg, 1995] was used for these small scale simulations instead. The use of the Storey q-values will be demonstrated in the real data analysis. Also, DMRs were not allowed to consist of sites that were not contiguous in the simulation, this is because Bumphunter is not designed to detect DMRs defined in this way, so it will perform artificially poor and is not a fair comparison. The other settings used for DMR Detector were investigated more deeply in simulation.

To begin, the choices for ridge penalties λ_j defined in equation (3.5) were initially set to be very small, with specific values from a sequence of 0.001 to 0.01 by 0.001. The choices for smoothing parameter λ_c defined in equation (3.10) that controls the size of the kernel, such as the standard deviation in the case of the Gaussian kernel, were initially set to be values selected from the window size by tenths, as in 0.1 to 0.9 by 0.1 times the length of the window, which in this case is 25 CpG sites. Giving 9 choices for this parameter allows the GCV procedure ample opportunity to identify the optimal one. Fewer choices were investigated later to increase speed. The investigation began using 250 bootstrap samples to assign significance, since that was the default in the Bumphunter R package [Jaffe et al., 2012].

In this first simulation, the smoothing window size was initially set to be the full window of 25 CpG sites, which is essentially a fixed window size. As mentioned previously, intuition suggested that the method may need the ability to reach out and find some less correlated points, and that a larger, fixed window size might be preferable. However the method had no ability to detect the true DMR! Several different cutoffQ values were then tried, with the lowest being 0 since this is only a window of 25 CpG sites, and the highest being 0.99 since that was recommended in Bumphunter. Using a cutoffQ of 0 was found to be best, however adjusting the cutoffQ had no effect on the power. Even when the window size was increased from 25 CpG sites to 50 or 100, this severe power deficit persisted. So the next idea was to investigate the effect of using a variable window size like Bumphunter as defined by the maxGap parameter.

The maxGap parameter used in Bumphunter determines the maximum separation between CpG sites that can be considered as part of the same region of interest. Simulations

revealed that using this variable window size fixed the power deficiency for small values of the maxGap parameter. However then a new problem arose, as the type I error was inflated above the 0.1 FDR level.

Since 250 bootstraps seemed rather small, the next idea was to investigate the effect of increasing the number of bootstrap samples to 500 to see if that would reduce the type I error rate of DMR Detector. However 500 bootstraps gave very similar results to the results from 250 boostraps for both power and type I error, and the type I error rate for DMR Detector did not appear to reduce.

Next, the values for the ridge penalties λ_j defined in equation (3.5) were investigated in trying to identify the source of the inflated type I errors, keeping the number of bootstraps at 500. From these results it appeared that DMR Detector had a preference for smaller ridge penalties, as the power dropped off as the size of the penalties increased. The errors were still inflated, but this gave us a clue as to what the problem was.

Recall that ridge regression applies a shrinkage to the coefficients, so using GCV to determine the optimal tuning parameter at each loci could result in different amounts of shrinkage applied to each loci. This could potentially distort the shape of the true function across the genome. One idea to remedy this was to apply the same ridge penalty at each loci, which would apply uniform shrinkage, and should not distort the shape of the function across the genome.

Next, small, single values were investigated in order to determine if it was better to set the same penalty across all loci, or to continue to allow GCV to choose the optimum ridge penalty automatically. From these results it was determined that using a penalty of $\lambda_j = 0.007$ lead to the highest power and lowest error. However the errors were

still inflated. This is likely because $\lambda_j = 0.007$ was not the optimal tuning parameter everywhere, resulting in a poor fit at many loci. We know in ridge regression we need to determine the optimal tuning parameter in order to get a good fit [Hastie et al., 2009]. Note, though, that this setting resulted in the lowest error seen so far for DMR Detector. This confirmed our suspicions that the shrinkage was likely the source of the error inflation, that using uniform shrinkage was better, but was still problematic.

There are several good methods that can be used to address a potentially rank deficient design matrix other than ridge regression, these include Lasso regression, Elastic Net regression, Principal components regression, and Partial Least Squares [Hastie et al., 2009]. However use of any of these methods may also be problematic. Lasso and Elastic Net regression are also shrinkage methods like ridge, and would suffer from the same problem. Principal components regression and Partial Least Squares do not give an estimate of the coefficient for any single variable, but instead produce linear combinations of the input variables. Therefore these cannot be used to detect DMRs with respect to only a single variable like group.

This is likely why the authors of Bumphunter developed a new procedure for the initial fitting in their method, termed the ".getEstimate" function. They did not give any details on the procedure, they just called it as a least squares fit and referred users to their code as the only reference [Jaffe et al., 2012]. The next idea was to investigate the effect of using the .getEstimate function for the initial fitting instead of ridge regression.

Table IX presents results for empirical power and Table X presents results for empirical family-wise type I error for different choices for the maxGap parameter with the .getEstimate function used for the initial fitting instead of ridge regression. Figure 6

presents the corresponding power curve. At first 250 bootstraps were used to take a look. Immediately we can see that this fixed the error inflation problem. All empirical family-wise type I error rates for all settings of the maxGap parameter are now well below the 0.1 FDR, while still maintaining high power for lower values of maxGap. Additionally, the errors are lower than those for Bumphunter!

Note that the power for DMR Detector becomes 0 again for larger values of maxGap. This is because larger values of maxGap are approaching the fixed window size of 25 CpG sites. Recall that DMR Detector assigns high weights to points that have a low correlation with the target point, and that points that are further away are likely less correlated. When maxGap is large, points that are very far away are likely receiving high weights and therefore have a strong effect on the fitting of the target point, leading to a poor fit and low power. Using a smaller maxGap setting prevents this from happening, so that each target point is fit with points that are in a smaller neighborhood around the target point. Bumphunter is less prone to this problem, and still has high power with large values of maxGap, because the points are weighted solely according to distance from the target point. Therefore, points that are far away are receiving low weights anyway, and have very little effect on the fitting of the target point. Next the number of bootstraps was increased to 500 for confirmation.

Table XI presents results for empirical power and Table XII presents results for empirical family-wise type I error for different choices for the maxGap parameter using the .getEstimate function and 500 bootstrap samples. Figure 7 presents the corresponding power curve. Again we see the same trend, all empirical family-wise type I error rates for all settings of the maxGap parameter are well below the 0.1 FDR, while still maintaining

Table IX: Empirical Power for maxGap with .getEstimate and 250 bootstraps

maxGap	500	750	1000	1250	1500
DMR Detector	1.00	0.999	1.00	0	0
Bumphunter	1.00	1.00	1.00	0.920	0.923

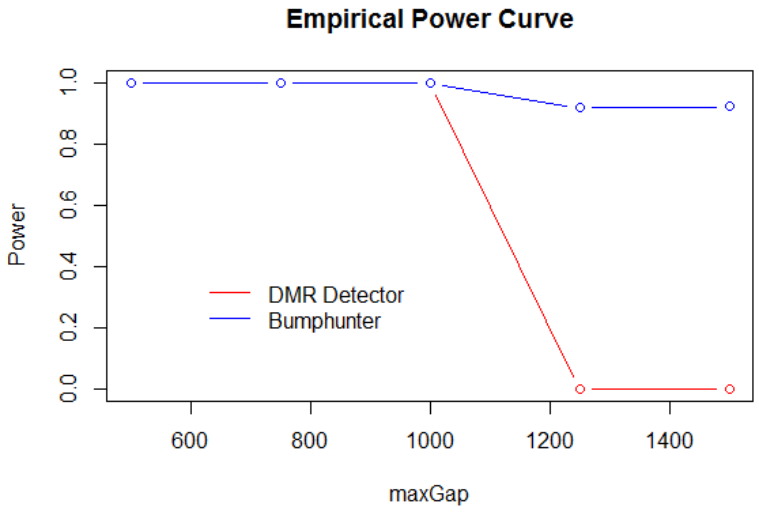

Figure 6: Empirical Power Curve for maxGap with .getEstimate and 250 bootstraps

high power for lower values of maxGap. Additionally, the errors are still consistently lower than those for Bumphunter! The best maxGap setting for both methods appears to be 1000. Next the number of boostrap samples was increased to 1000 for further confirmation.

Table XIII presents results for empirical power and Table XIV presents results for empirical family-wise type I error for different choices for the maxGap parameter using the .getEstimate function and 1000 bootstrap samples. Figure 8 presents the corresponding power curve. Again we see confirmation of the same trend of consistently lower type I errors than Bumphunter, controlled well below the 0.1 level, with high power for lower

Table X: Empirical Type I Error for maxGap with .getEstimate and 250 bootstraps

maxGap	500	750	1000	1250	1500
DMR Detector	0.075	0.072	0.071	0.057	0.065
Bumphunter	0.090	0.097	0.103	0.110	0.083

Table XI: Empirical Power for maxGap with .getEstimate and 500 bootstraps

maxGap	500	750	1000	1250	1500
DMR Detector	1.00	1.00	1.00	0	0
Bumphunter	1.00	1.00	1.00	0.943	0.922

values of maxGap. The best maxGap setting for both methods again appears to be 1000. Since we arrived at the same conclusions using only 500 bootstraps, the simulations will continue for other parameters using 500 bootstraps instead of 1000 due to time constraints, as 500 appears to be good enough. Next, we finally move on to investigate another parameter.

To investigate whether the correlation adjusted kernel weight is performing as hoped, simulations were performed by making the data more and more correlated. Table XV presents results for empirical power and Table XVI presents results for empirical family-wise type I error for different choices for the AR coefficient. Figure 9 presents the corresponding power curve. Recall that the ”arima” package in R [R Core Team, 2016] was used to emulate the Autism data using an AR(1) process with coefficient 0.20 and a standard deviation of 0.23. We then chose higher and higher values for the coefficient in

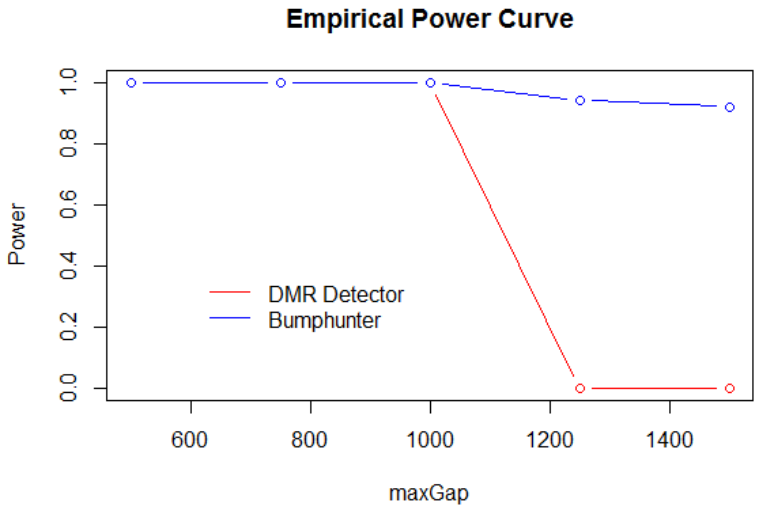

Figure 7: Empirical Power Curve for maxGap with .getEstimate and 500 bootstraps

Table XII: Empirical Type I Error for maxGap with .getEstimate and 500 bootstraps

maxGap	500	750	1000	1250	1500
DMR Detector	0.080	0.080	0.067	0.076	0.071
Bumphunter	0.105	0.089	0.087	0.106	0.099

order to make the data more and more correlated. From these results we can see that DMR Detector has higher power than Bumphunter as the correlation increases, however both methods begin to lose control of the type I error with higher values of the AR coefficient.

The poor control of the type I error in Bumphunter with higher values of the AR coefficient is likely because the loess smoother is not designed to handle correlated errors. The poor control of the type I error in DMR Detector with higher values of the AR coefficient is likely because there are fewer less correlated points nearby when the AR coefficient is very high. Recall that the correlation adjusted kernel weight heavily utilizes

Table XIII: Empirical Power for maxGap with .getEstimate and 1000 bootstraps

maxGap	500	750	1000	1250	1500
DMR Detector	1.00	1.00	1.00	0	0
Bumphunter	0.999	1.00	1.00	0.940	0.926

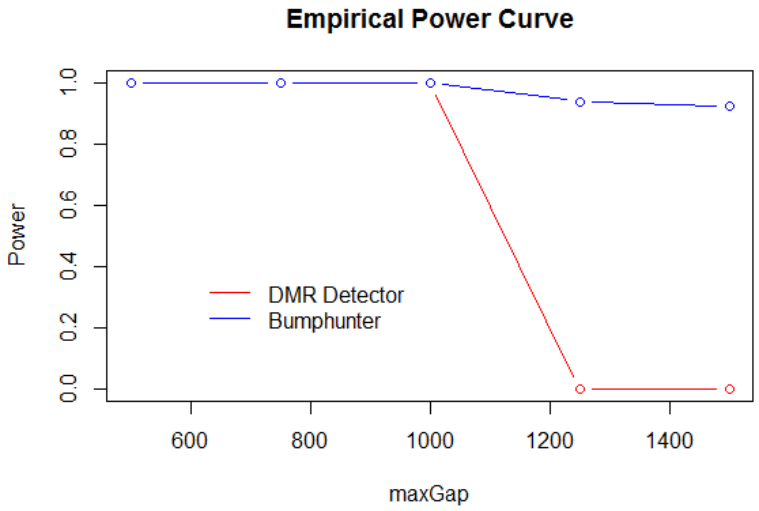

Figure 8: Empirical Power Curve for maxGap with .getEstimate and 1000 bootstraps

less correlated points. If the AR coefficient is very high, this could result in smoothing windows comprised entirely of highly correlated points, where the correlation adjusted kernel weight cannot work as intended.

Note that both methods control the error when the coefficient is 0.2, which was the observed AR coefficient for the 450K Autism data, but only DMR Detector controls it when it is 0.4. One idea for future work as a result of this finding is to estimate the AR coefficient for the new 850K array. Since there are 850,000 CpG sites instead of only 450,000 for the 450K array, intuition suggests the AR coefficient might be higher since the

Table XIV: Empirical Type I Error for maxGap with .getEstimate and 1000 bootstraps

maxGap	500	750	1000	1250	1500
DMR Detector	0.083	0.072	0.063	0.048	0.061
Bumphunter	0.102	0.094	0.092	0.085	0.107

sites are more densely populated on the array. If the AR coefficient for the 850K array is around 0.4, DMR Detector would clearly be the better method because Bumphunter does not control the error at the 0.1 level at that correlation. This idea will be added to future work.

Table XV: Empirical Power for AR coefficient

AR coefficient	0.2	0.4	0.6	0.8	0.9
DMR Detector	1.00	1.00	1.00	0.998	0.999
Bumphunter	1.00	1.00	0.994	0.975	0.898

Table XVI: Empirical Type I Error for AR coefficient

AR coefficient	0.2	0.4	0.6	0.8	0.9
DMR Detector	0.067	0.070	0.105	0.122	0.125
Bumphunter	0.087	0.109	0.104	0.119	0.112

Recall one goal of this research was to try to improve detection ability for small effect sizes. This was an issue raised through collaboration with Dr. Xiaoling Wang at Georgia Prevention Institute (GPI), and is also supported in the literature [Breton et al., 2017]. Dr.

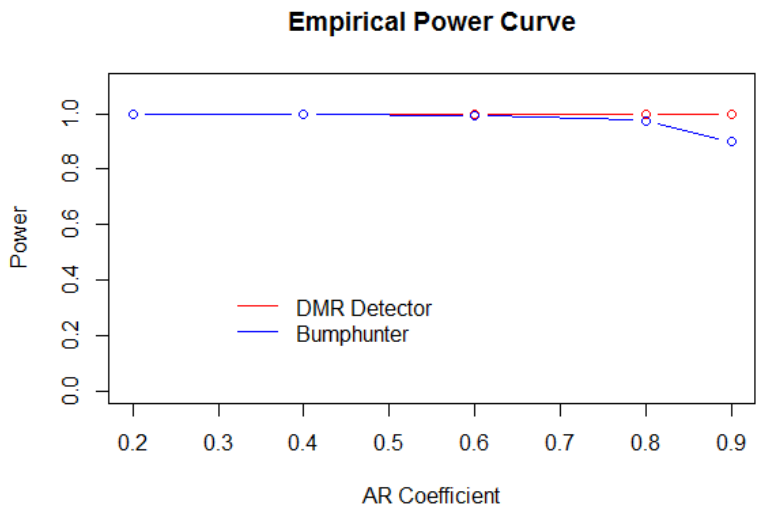

Figure 9: Empirical Power Curve for AR coefficient

Wang explained that for some complex diseases, the effect sizes can be rather small, and are often too small for current DMR finding methods to detect. In the proposal stage, a plot suggested that the Gaussian kernel may be better able to detect DMRs with smaller effect sizes than the Epanechnikov. This idea was also investigated in simulation. Table XVII presents simulation results for empirical power for different choices of kernel type and effect size. Figure 10 presents the corresponding power curve. Denote these effect sizes for $\beta(t_j)$ defined in model (3.1) as follows. The size XL=0.19-2.69, which was the original effect size seen in the Autism data, L=.19-2, M=.19-1.5, S=.19-1, XS=.19-0.5, XXS=.1-.19, XXXS=.01-.1. From these results we see that the two kernels perform very similarly, but the power of the Epanechnikov kernel actually appears to be slightly higher than the Gaussian as the effect size gets smaller. So the Epanechnikov kernel will become the kernel used in the method from now on. Next, we use DMR Detector with the Epanechnikov kernel and compare to Bumphunter.

Table XVII: Empirical Power for Kernel Type and Effect size

Effect size	XL	L	M	S	XS	XXS	XXXS
Gaussian	1.00	1.00	1.00	1.00	1.00	0.659	0.143
Epanechnikov	1.00	1.00	1.00	1.00	0.999	0.665	0.166

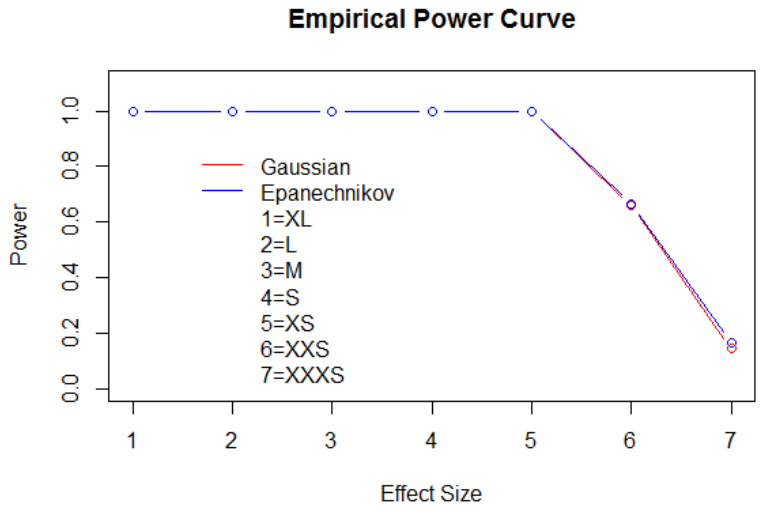

Figure 10: Empirical Power Curve for Kernel Type and Effect size

Table XVIII presents results for empirical power for different choices of effect size comparing DMR Detector with the Epanechnikov kernel to Bumphunter. Figure 11 presents the corresponding power curve. From these results we can see that DMR Detector has slightly higher power than Bumphunter as the effect size gets smaller and smaller. So this goal of the research was accomplished! However there is still more work to be done in this area because the increase in power is very small, and we can see that the power of both methods dies off as the effect size gets smaller and smaller. There are still more improvements that can be made in terms of detection ability for the 450K array. It seems

that the most consistent improvement of DMR Detector is a reduction in the family-wise type I error rate shown in previous simulations.

Table XVIII: Empirical Power for Effect size

Effect size	XL	L	M	S	XS	XXS	XXXS
DMR Detector	1.00	1.00	1.00	1.00	0.999	0.665	0.166
Bumphunter	1.00	1.00	1.00	1.00	0.993	0.659	0.144

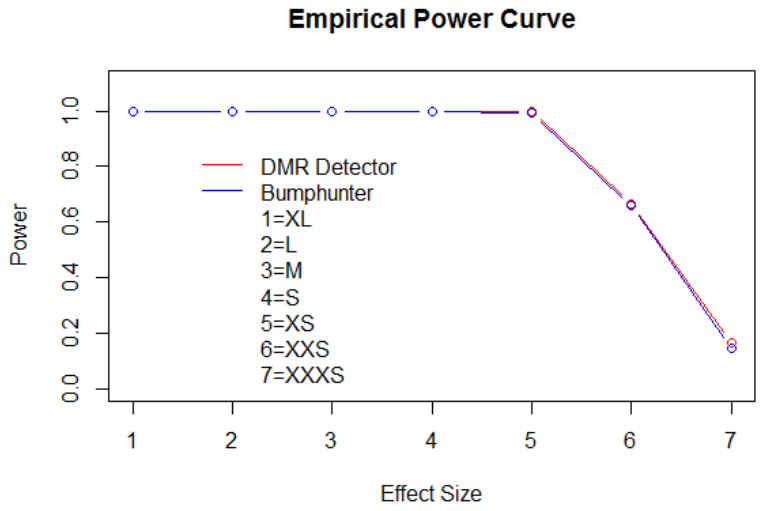

Figure 11: Empirical Power Curve for Effect size

The final simulation will investigate a reduction in the number of choices given to GCV for the smoothing parameter λ_c defined in equation (3.10) in order to increase speed. Table XIX presents results for empirical power and Table XX presents results for empirical family-wise type I error for different choices of λ_c. Figure 12 presents the corresponding power curve. Recall that λ_c controls the size of the kernel. Recall that these values were initially set to be values selected from the window size by tenths, as in 0.1 to 0.9 by 0.1

times the length of the window. Other choices were Quarters=(.25,.5,.75) times the length of the window, Thirds=(.3,.6) times the length of the window, and finally a single value of 0.3 as was used in Bumphunter. From these results we can see that all choices result in very similar values, with comparable power to Bumphunter, and consistently lower type I errors. Giving GCV the full 9 choices resulted in the lowest error, but giving it thirds or even the single 0.3 resulted in very similar errors and will be much faster. Next we will move on to the real data analysis using thirds as the choices for λ_c so that GCV can select the optimal value from those two in each smoothing window.

Table XIX: Empirical Power for Smoothing Parameter λ_c

Smoothing Parameter	Tenths	Quarters	Thirds	0.3
DMR Detector	1.00	1.00	1.00	1.00
Bumphunter	NA	NA	NA	1.00

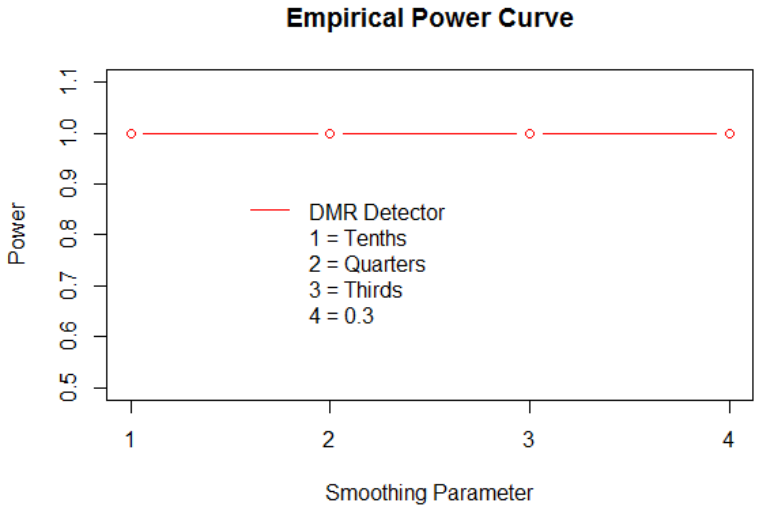

Figure 12: Empirical Power Curve for Smoothing Parameter λ_c

Table XX: Empirical Type I Error for Smoothing Parameter λ_c

Smoothing Parameter	Tenths	Quarters	Thirds	0.3
DMR Detector	0.068	0.073	0.069	0.069
Bumphunter	NA	NA	NA	0.087

5. REAL DATA ANALYSIS

The methods of DMR Detector and Bumphunter were applied to the published Autism GEO dataset that was analyzed using Bumphunter mentioned earlier [Ladd-Acosta et al., 2014], and the childhood obesity data from Dr. Wang at GPI. DMR Detector was found to be too slow to run whole genome on the LISA cluster without hitting the maximum wall time. Bumphunter was able to run without issue, and the only major difference between the coding in Bumphunter and DMR Detector is that Bumphunter sends parts of the code to C to get compiled. We know base programming languages are much faster than R, so one thing that will be added to future work is to learn a base programming language like C/C++/Perl in order to make the method run faster. Due to this, the real data analysis was performed on one chromosome for each data set.

5.1 Autism

The methods were applied to the Autism data on chromosome 5. Chromosome 5 was selected because one DMR was identified in this region for the Cerebellum in the Autism [Ladd-Acosta et al., 2014] paper. Initially a cutoffQ of 0.99 was used because this was the optimal value recommended in the Bumphunter [Jaffe et al., 2012] paper for genome-

wide analysis, however Bumphunter was not able to detect the DMR that was presented in the Autism paper with this cutoff using only one chromosome, so then a slightly lower cutoffQ of 0.95 was tried instead. The results from Bumphunter using a cutoffQ of 0.95 are presented in Table XXI. From this table we can see that Bumphunter was able to detect the same DMR from the Autism paper using this cutoff, and that it was the only significant DMR after adjustment for multiple testing, which was the same conclusion reached from the genome-wide analysis in the paper.

The results from DMR Detector are presented in Table XXII using the same cutoffQ of 0.95. From this table we can see that DMR Detector identified two DMRs that were significant after adjustment for multiple testing. The first DMR was in the same region as the one identified by Bumphunter in the Autism paper [Ladd-Acosta et al., 2014]. Note, however, that the start and end positions were different from those identified by Bumphunter, making the DMR identified by DMR Detector slightly smaller than the one identified by Bumphunter.

Since this is real data, it is not possible to be certain which boundaries are correct. However since DMR Detector was shown to have a lower empirical type I error rate than Bumphunter in simulation, it may be that the end points for Bumphunter are errors. Figure 13 presents a plot of this DMR on chromosome 5, with the different boundaries for each method marked. Indeed we can see from this plot that the samples for the Autism patients and control patients are less separated at the Bumphunter start and end points than they are at the start and end points identified by DMR Detector. It is possible that the samples are not actually differentially methylated there.

Also note that the second significant DMR identified by DMR Detector was not

comprised of contiguous CpG sites only, but rather a region where more than 75 percent of the region was above the cutoffQ of 0.95. This can be calculated by L (length of DMR) divided by ClusterL (length of cluster). Biological validation of this DMR was then attempted using a literature review, however this DMR was not found to be reported in any of the Autism literature. Recall that this analysis was performed on only one chromosome, so the method needs to run on the whole genome to confirm if it is actually significant genome-wide.

Table XXI: Bumphunter Autism Analysis

Chr	Start	End	Area	L	ClusterL	p-valueArea	fwerArea
5	1594021	1595048	21.59	12	12	9.51e-05	0.022
5	135414858	135416613	12.07	20	20	0.005	0.674
5	19988200	19989519	7.23	14	15	0.014	0.950
5	1110019	1110315	4.99	4	31	0.030	0.996
5	79865276	79865402	3.40	2	11	0.066	0.998

5.2 Childhood Obesity

The methods were then applied to the childhood obesity data that had corresponding gene expression data for validation. Recall that this is the dataset that has extremely small effect sizes. First, chromosome 22 was selected at random for analysis, but neither method was able to detect any significant DMRs. Next, a reverse procedure was used, where the chromosome for the most significantly expressed gene was located first, which was defined

Table XXII: DMR Detector Autism Analysis

Chr	Start	End	Area	Contiguous	L	ClusterL	p-value	q-value
5	1594282	1594863	16.56	1	10	12	8.12e-05	0.013
5	135414858	135416613	11.31	0	18	20	6.01e-04	0.049
5	218153	218452	3.13	1	3	15	1.27e-02	0.628
5	139228062	139228153	2.77	1	3	8	1.55e-02	0.628
5	173315974	173316748	1.63	1	3	17	3.92e-02	1

by identifying the gene with the lowest p-value. This was the NRG1 gene on chromosome 8. Then the real data analysis was performed on chromosome 8 using a cutoffQ of 0.95. The results from Bumphunter are presented in Table XXIII and the results from DMR Detector are presented in Table XXIV. However, again, neither method was able to detect any significant DMRs.

Since the effect sizes are expected to be very small in this data set, the next idea was to reduce the cutoffQ to 0, and not impose any cutoff at all. These results for Bumphunter are presented in Table XXV and these results from DMR Detector are presented in Table XXVI. Unfortunately, again, neither method was able to detect any significant DMRs. Note that the area of the largest DMR was 0.32 for Bumphunter and 0.12 for DMR Detector. These values indicate effect sizes in those regions that are in the XXS or XXXS range that were investigated in simulation. Recall that the power for both methods was found to be extremely low for effect sizes this small. This explains why neither method was able to detect any significant DMRs. There is more work to be done on detecting small effect sizes

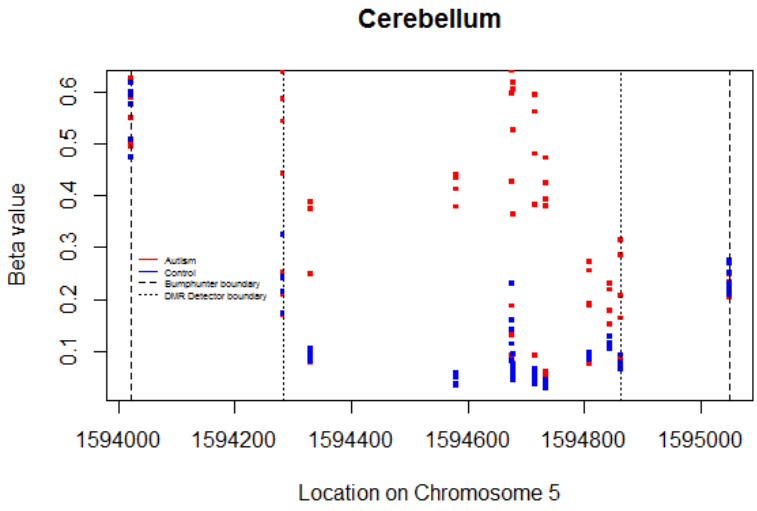

Figure 13: Plot of DMR in Cerebellum from Autism Paper

for the 450K array, so this will be added to future work.

Table XXIII: Bumphunter Childhood Obesity Analysis with cutoffQ=0.95

Chr	Start	End	Area	L	ClusterL	p-valueArea	fwerArea
8	1893793	1893887	0.31	2	35	0.002	0.340
8	1327367	1328459	0.27	8	10	0.004	0.468
8	143859143	143859990	0.17	12	22	0.014	0.906
8	1364518	1365906	0.16	9	11	0.018	0.952
8	1792758	1792775	0.11	2	7	0.039	1

Table XXIV: DMR Detector Childhood Obesity Analysis with cutoffQ=0.95

Chr	Start	End	Area	Contiguous	L	ClusterL	p-value	q-value
8	145654565	145654780	0.04	1	4	14	0.057	1
8	17942993	17943293	0.04	1	2	19	0.057	1
8	144371537	144371779	0.03	1	3	19	0.094	1
8	65711522	65711658	0.03	1	3	13	0.094	1
8	87354422	87354470	0.02	1	2	15	0.212	1

Table XXV: Bumphunter Childhood Obesity Analysis with cutoffQ=0

Chr	Start	End	Area	L	ClusterL	p-valueArea	fwerArea
8	1893793	1893977	0.32	4	35	0.0003	0.320
8	1327367	1328707	0.28	10	10	0.0005	0.508
8	143858612	143859990	0.18	18	22	0.002	0.904
8	145014989	145020593	0.07	35	35	0.018	1
8	145725402	145731409	0.06	34	34	0.024	1

Table XXVI: DMR Detector Childhood Obesity Analysis with cutoffQ=0

Chr	Start	End	Area	Contiguous	L	ClusterL	p-value	q-value
8	1270577	1274686	0.12	1	14	14	0.004	0.413
8	143858414	143859990	0.12	1	22	22	0.004	0.413
8	685830	690059	0.10	1	15	15	0.009	0.685
8	25897201	25909599	0.08	1	58	58	0.024	0.931
8	145725402	145731409	0.08	1	34	34	0.024	0.931

6. CONCLUSION

In conclusion, a new statistical method was developed for detecting differentially methylated regions on the 450K array. The new method was developed by making small modifications to Bumphunter, including a new statistical contribution on how to perform kernel smoothing under dependence, and was compared to Bumphunter on both simulated and real data.

In simulation, the method was shown to have high power comparable to Bumphunter, with consistently lower family-wise type I error rate, controlled well below the 0.1 FDR. With small effect sizes, the method was shown to have slightly higher power than Bumphunter.

In the real data analysis of the Autism data, DMR Detector identified a DMR in the same region that was identified by Bumphunter, however the DMR was slightly smaller with different start and end points. DMR Detector also identified a potentially new DMR on chromosome 5, however the method needs to be run on the whole genome to assess if this DMR is significant genome-wide.

For the childhood obesity data, unfortunately, neither method was able to identify any significant DMRs. The effect sizes for these DMRs seemed to be extremely small, in the range where both methods were shown to have extremely low power in simulation. There

is obviously more work to be done on detecting small effect sizes for the 450K array, so this will be the goal of future work.

7. FUTURE WORK

There are several ideas for future work. The first idea is to apply the method to 850K data and run simulations. Recall that in the simulation we learned that if the AR coefficient for the 850K array is around 0.4, DMR Detector would clearly be the better method because Bumphunter does not control the error at the 0.1 level at that correlation.

Another idea for future work is to perform additional simulations for lower values of the maxGap parameter. It appears that DMR Detector has no power for large values of maxGap, but high power for lower values of maxGap. Running more simulations in the smaller range of the maxGap parameter may also be informative.

The next idea for future work is to extend the method to be robust to outliers. The loess smoother used in Bumphunter is a robust smoother [Cleveland, 1979, Hollander et al., 2015]. After performing the initial regression, additional regressions are performed involving weighting points by their residuals from the first regression to reduce the impact of outliers. A similar procedure could be followed here.

Also, there is future work that could be done for the new smoother itself. A full literature review could be performed, and the smoother could be compared with other methods for smoothing under dependence.

And finally, more work needs to be done on detecting small effect sizes. This new

method showed slight improvement in power over Bumphunter, but the power of both methods dies off as the effect size gets smaller and smaller. One potential idea is to modify the method to accept the average methylation from a meta-analysis, and create regions from that. There are still more improvements that can be made in terms of detection ability for the 450K array.

References

[Altman, 1990] Altman, N. S. (1990). Kernel Smoothing of Data With Correlated Errors. *Journal of the American Statistical Association*, 85(411):749–759.

[Benjamini and Hochberg, 1995] Benjamini, Y. and Hochberg, Y. (1995). Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, 57(1):289–300.

[Bibikova et al., 2011] Bibikova, M., Barnes, B., Tsan, C., Ho, V., Klotzle, B., Le, J. M., Delano, D., Zhang, L., Schroth, G. P., Gunderson, K. L., Fan, J.-B., and Shen, R. (2011). High density DNA methylation array with single CpG site resolution. *Genomics*, 98(4):288–295.

[Bibikova et al., 2009] Bibikova, M., Le, J., Barnes, B., Saedinia-Melnyk, S., Zhou, L., Shen, R., and Gunderson, K. L. (2009). Genome-wide DNA methylation profiling using infinium assay. *Epigenomics*, 1(1):177–200.

[Bibikova et al., 2006] Bibikova, M., Lin, Z., Zhou, L., Chudin, E., Garcia, E. W., Wu, B., Doucet, D., Thomas, N. J., Wang, Y., Vollmer, E., Goldmann, T., Seifart, C., Jiang, W., Barker, D. L., Chee, M. S., Floros, J., and Fan, J.-B. (2006). High-throughput DNA methylation profiling using universal bead arrays. *Genome Research*, 16(3):383–393.

[Breton et al., 2017] Breton, C. V., Marsit, C. J., Faustman, E., Nadeau, K., Goodrich, J. M., Dolinoy, D. C., Herbstman, J., Holland, N., LaSalle, J. M., Schmidt, R., Yousefi, P., Perera, F., Joubert, B. R., Wiemels, J., Taylor, M., Yang, I. V., Chen, R., Hew, K. M., Freeland, D. M. H., Miller, R., and Murphy, S. K. (2017). Small-Magnitude Effect Sizes in Epigenetic End Points are Important in Children's Environmental Health Studies: The Children's Environmental Health and Disease Prevention Research Center's Epigenetics Working Group. *Environmental Health Perspectives*, 125(4):511–526.

[Butcher and Beck, 2015] Butcher, L. M. and Beck, S. (2015). Probe Lasso: a novel method to rope in differentially methylated regions with 450k DNA methylation data. *Methods (San Diego, Calif.)*, 72:21–28.

[Cai, 2001] Cai, Z. (2001). Weighted nadaraya-watson regression estimation. *Statistics & Probability Letters*, 51(3):307–318.

[Chen et al., 2013] Chen, Y.-a., Lemire, M., Choufani, S., Butcher, D. T., Grafodatskaya, D., Zanke, B. W., Gallinger, S., Hudson, T. J., and Weksberg, R. (2013). Discovery of cross-reactive probes and polymorphic CpGs in the illumina infinium HumanMethylation450 microarray. *Epigenetics*, 8(2):203–209.

[Cleveland, 1979] Cleveland, W. S. (1979). Robust Locally Weighted Regression and Smoothing Scatterplots. *Journal of the American Statistical Association*, 74(368):829–836.

[Craven and Wahba, 1977] Craven, P. and Wahba, G. (1977). *Smoothing noisy data with spline functions: estimating the correct degree of smoothing by the method of*

generalized cross-validation. Dept. of Statistics, University of Wisconsin, Madison, Wisc.

[Day et al., 2013] Day, K., Waite, L. L., Thalacker-Mercer, A., West, A., Bamman, M. M., Brooks, J. D., Myers, R. M., and Absher, D. (2013). Differential DNA methylation with age displays both common and dynamic features across human tissues that are influenced by CpG landscape. *Genome Biology*, 14(9):R102.

[De Brabanter et al., 2011] De Brabanter, K., De Brabanter, J., Suykens, J. A. K., and Moor, B. D. (2011). Kernel Regression in the Presence of Correlated Errors. *Journal of Machine Learning Research*, 12.

[Dedeurwaerder et al., 2014] Dedeurwaerder, S., Defrance, M., Bizet, M., Calonne, E., Bontempi, G., and Fuks, F. (2014). A comprehensive overview of infinium HumanMethylation450 data processing. *Brief Bioinform*, 15(6):929–941.

[Dedeurwaerder et al., 2011] Dedeurwaerder, S., Desmedt, C., Calonne, E., Singhal, S. K., Haibe-Kains, B., Defrance, M., Michiels, S., Volkmar, M., Deplus, R., Luciani, J., Lallemand, F., Larsimont, D., Toussaint, J., Haussy, S., Roth, F., Rouas, G., Metzger, O., Majjaj, S., Saini, K., Putmans, P., Hames, G., van Baren, N., Coulie, P. G., Piccart, M., Sotiriou, C., and Fuks, F. (2011). DNA methylation profiling reveals a predominant immune component in breast cancers. *EMBO Mol Med*, 3(12):726–741.

[Du et al., 2010] Du, P., Zhang, X., Huang, C.-C., Jafari, N., Kibbe, W. A., Hou, L., and Lin, S. M. (2010). Comparison of beta-value and m-value methods for quantifying methylation levels by microarray analysis. *BMC Bioinformatics*, 11(1):587.

[Eckhardt et al., 2006] Eckhardt, F., Lewin, J., Cortese, R., Rakyan, V. K., Attwood, J., Burger, M., Burton, J., Cox, T. V., Davies, R., Down, T. A., Haefliger, C., Horton, R., Howe, K., Jackson, D. K., Kunde, J., Koenig, C., Liddle, J., Niblett, D., Otto, T., Pettett, R., Seemann, S., Thompson, C., West, T., Rogers, J., Olek, A., Berlin, K., and Beck, S. (2006). DNA methylation profiling of human chromosomes 6, 20 and 22. *Nature Genetics*, 38(12):1378–1385.

[Efron and Tibshirani, 1994] Efron, B. and Tibshirani, R. J. (1994). *An Introduction to the Bootstrap*. CRC Press.

[Esteller, 2002] Esteller, M. (2002). CpG island hypermethylation and tumor suppressor genes: a booming present, a brighter future. *Oncogene*, 21(35):5427–5440.

[Fisher, 1948] Fisher, R. A. (1948). Questions and Answers #14. *The American Statistician*, 2(5):30–31.

[Fuller, 1996] Fuller, W. A. (1996). *Introduction to Statistical Time Series*. John Wiley & Sons.

[Golub et al., 1979] Golub, G. H., Heath, M., and Wahba, G. (1979). Generalized Cross-Validation as a Method for Choosing a Good Ridge Parameter. *Technometrics*, 21(2):215–223.

[Graybill, 1976] Graybill, F. A. (1976). *Theory and Application of the Linear Model*. Duxbury Press.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition*. Springer Science & Business Media.

[Herman and Baylin, 2003] Herman, J. G. and Baylin, S. B. (2003). Gene silencing in cancer in association with promoter hypermethylation. *N. Engl. J. Med.*, 349(21):2042–2054.

[Hochberg and Benjamini, 1990] Hochberg, Y. and Benjamini, Y. (1990). More powerful procedures for multiple significance testing. *Statistics in Medicine*, 9(7):811–818.

[Hollander et al., 2015] Hollander, M., A. Wolfe, D., and Chicken, E. (2015). *Nonparametric Statistical Methods: Hollander/Nonparametric Statistical Methods*. John Wiley & Sons, Inc., Hoboken, NJ, USA.

[Jaffe et al., 2012] Jaffe, A. E., Murakami, P., Lee, H., Leek, J. T., Fallin, M. D., Feinberg, A. P., and Irizarry, R. A. (2012). Bump hunting to identify differentially methylated regions in epigenetic epidemiology studies. *Int J Epidemiol*, 41(1):200–209.

[Kechris et al., 2010] Kechris, K. J., Biehs, B., and Kornberg, T. B. (2010). Generalizing moving averages for tiling arrays using combined p-value statistics. *Statistical Applications in Genetics and Molecular Biology*, 9:Article29.

[Kuan et al., 2010] Kuan, P. F., Wang, S., Zhou, X., and Chu, H. (2010). A statistical framework for Illumina DNA methylation arrays. *Bioinformatics*, 26(22):2849–2855.

[Ladd-Acosta et al., 2014] Ladd-Acosta, C., Hansen, K. D., Briem, E., Fallin, M. D., Kaufmann, W. E., and Feinberg, A. P. (2014). Common DNA methylation alterations in multiple brain regions in autism. *Molecular Psychiatry*, 19(8):862–871.

[Leek et al., 2010] Leek, J. T., Scharpf, R. B., Bravo, H. C., Simcha, D., Langmead, B., Johnson, W. E., Geman, D., Baggerly, K., and Irizarry, R. A. (2010). Tackling the widespread and critical impact of batch effects in high-throughput data. *Nat. Rev. Genet.*, 11(10):733–739.

[Leek and Storey, 2007] Leek, J. T. and Storey, J. D. (2007). Capturing Heterogeneity in Gene Expression Studies by Surrogate Variable Analysis. *PLoS Genet*, 3(9):e161.

[Leek and Storey, 2008] Leek, J. T. and Storey, J. D. (2008). A general framework for multiple testing dependence. *Proceedings of the National Academy of Sciences of the United States of America*, 105(48):18718–18723.

[Liptak,] Liptak, T. On the combination of independent tests. *Magyar Tudomanyos. Akademia Matematikai Kutato Intezetenek Kozlemenyei*, 3(1958):171–197.

[Liu, 2001] Liu, X.-H. (2001). Kernel smoothing for spatially correlated data. Retrospective Theses and Dissertations.

[Marsit et al., 2011] Marsit, C. J., Koestler, D. C., Christensen, B. C., Karagas, M. R., Houseman, E. A., and Kelsey, K. T. (2011). DNA methylation array analysis identifies profiles of blood-derived DNA methylation associated with bladder cancer. *Journal of Clinical Oncology: Official Journal of the American Society of Clinical Oncology*, 29(9):1133–1139.

[Morris et al., 2014] Morris, T. J., Butcher, L. M., Feber, A., Teschendorff, A. E., Chakravarthy, A. R., Wojdacz, T. K., and Beck, S. (2014). ChAMP: 450k Chip Analysis Methylation Pipeline. *Bioinformatics*, 30(3):428–430.

[Nadaraya, 1964] Nadaraya, E. A. (1964). On Estimating Regression. *Theory of Probability and Its Application*, 9.

[Opsomer et al., 2001] Opsomer, J., Wang, Y., and Yang, Y. (2001). Nonparametric Regressin with Correlated Errors. *Statistical Science*, 16(2):134–153.

[Pedersen et al., 2012] Pedersen, B. S., Schwartz, D. A., Yang, I. V., and Kechris, K. J. (2012). Comb-p: software for combining, analyzing, grouping and correcting spatially correlated P-values. *Bioinformatics (Oxford, England)*, 28(22):2986–2988.

[Peters et al., 2015] Peters, T. J., Buckley, M. J., Statham, A. L., Pidsley, R., Samaras, K., V Lord, R., Clark, S. J., and Molloy, P. L. (2015). De novo identification of differentially methylated regions in the human genome. *Epigenetics & Chromatin*, 8:6.

[Price et al., 2013] Price, E. M., Cotton, A. M., Lam, L. L., Farr, P., Emberly, E., Brown, C. J., Robinson, W. P., and Kobor, M. S. (2013). Additional annotation enhances potential for biologically-relevant analysis of the illumina infinium HumanMethylation450 BeadChip array. *Epigenetics & Chromatin*, 6(1):4.

[R Core Team, 2016] R Core Team (2016). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

[Rakyan et al., 2011] Rakyan, V. K., Down, T. A., Balding, D. J., and Beck, S. (2011). Epigenome-wide association studies for common human diseases. *Nat. Rev. Genet.*, 12(8):529–541.

[Robinson et al., 2014] Robinson, M. D., Kahraman, A., Law, C. W., Lindsay, H., Nowicka, M., Weber, L. M., and Zhou, X. (2014). Statistical methods for detecting differentially methylated loci and regions. *Frontiers in Genetics*, 5:324.

[Rocke, 1993] Rocke, D. M. (1993). On the Beta Transformation Family. *Technometrics*, 35(1):72–81.

[Samuel A. Stouffer, 1949] Samuel A. Stouffer, E. A. S. (1949). The American soldier: Adjustment during army life. Volume I of studies in social psychology in World War II. 1.

[Satterthwaite, 1946] Satterthwaite, F. E. (1946). An Approximate Distribution of Estimates of Variance Components. *Biometrics Bulletin*, 2(6):110–114.

[Shaffer, 1995] Shaffer, J. P. (1995). Multiple Hypothesis Testing. *Annual Review of Psychology*, 46(1):561–584.

[Sheather, 2004] Sheather, S. J. (2004). Density Estimation. *Statistical Science*, 19(4):588–597.

[Smyth, 2004] Smyth, G. K. (2004). Linear models and empirical bayes methods for assessing differential expression in microarray experiments. *Statistical Applications in Genetics and Molecular Biology*, 3:Article3.

[Smyth et al., 2005] Smyth, G. K., Ritchie, M., Thorne, N., Wettenhall, J., and Shi, W. (2005). Limma: linear models for microarray data. In *Bioinformatics and Computational Biology Solutions using R and Bioconductor*, pages 397–420. Springer.

[Sofer et al., 2013] Sofer, T., Schifano, E. D., Hoppin, J. A., Hou, L., and Baccarelli, A. A. (2013). A-clustering: a novel method for the detection of co-regulated methylation regions, and regions associated with exposure. *Bioinformatics*, 29(22):2884–2891.

[Sproul et al., 2011] Sproul, D., Nestor, C., Culley, J., Dickson, J. H., Dixon, J. M., Harrison, D. J., Meehan, R. R., Sims, A. H., and Ramsahoye, B. H. (2011). Transcriptionally repressed genes become aberrantly methylated and distinguish tumors of different lineages in breast cancer. *Proceedings of the National Academy of Sciences of the United States of America*, 108(11):4364–4369.

[Storey, 2007] Storey, J. D. (2007). The Optimal Discovery Procedure: A New Approach to Simultaneous Significance Testing. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, (3):347.

[Storey and Tibshirani, 2003] Storey, J. D. and Tibshirani, R. (2003). Statistical significance for genomewide studies. *Proceedings of the National Academy of Sciences*, 100(16):9440–9445.

[Venables and Ripley, 2002] Venables, W. N. and Ripley, B. D. (2002). *Modern Applied Statistics with S*. Springer, 4th ed edition.

[Wang et al., 2012] Wang, D., Yan, L., Hu, Q., Sucheston, L. E., Higgins, M. J., Ambrosone, C. B., Johnson, C. S., Smiraglia, D. J., and Liu, S. (2012). IMA: an

r package for high-throughput analysis of illumina's 450k infinium methylation data. *Bioinformatics*, 28(5):729–730.

[Warden et al., 2013] Warden, C. D., Lee, H., Tompkins, J. D., Li, X., Wang, C., Riggs, A. D., Yu, H., Jove, R., and Yuan, Y.-C. (2013). COHCAP: an integrative genomic pipeline for single-nucleotide resolution DNA methylation analysis. *Nucleic Acids Research*, 41(11):e117.

[Watson, 1964] Watson, G. S. (1964). Smooth Regression Analysis. *Sankhy: The Indian Journal of Statistics; Series A*, 26.

[Wu et al., 2013] Wu, D., Gu, J., and Zhang, M. Q. (2013). FastDMA: an infinium humanmethylation450 beadchip analyzer. *PloS One*, 8(9):e74275.

[Zaykin et al., 2002] Zaykin, D. V., Zhivotovsky, L. A., Westfall, P. H., and Weir, B. S. (2002). Truncated product method for combining P-values. *Genetic Epidemiology*, 22(2):170–185.

[Zhang et al., 2012] Zhang, X., Mu, W., and Zhang, W. (2012). On the analysis of the illumina 450k array data: Probes ambiguously mapped to the human genome. *Front Genet*, 3.