Likelihood-based haplotype frequency modeling using variable-order Markov chains

Aaron Baraff

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2022

Reading Committee:

Elizabeth A. Thompson, Chair

Sharon R. Browning

Thomas S. Richardson

Program Authorized to Offer Degree:
Statistics

University of Washington

Abstract

Likelihood-based haplotype frequency modeling using variable-order Markov chains

Aaron Baraff

Chair of the Supervisory Committee:
Professor Emeritus Elizabeth A. Thompson
Statistics

The localized haplotype-cluster model uses variable-order Markov chains (VOMCs) to create an empirical model for haplotype probabilities that adapts to the changing structure of linkage disequilibrium (LD) across the genome. By clustering partial haplotypes based on the Markov property as represented by a directed acyclic graph (DAG), the model is able to take advantage of context-sensitive conditional independencies to improve estimates of haplotype frequencies while still respecting the dependencies induced by LD. We introduce a method for training such models using regularized likelihood functions to prevent overfitting along with a method for cross-validation to select a regularization parameter which accounts for the high probability of out-of-sample haplotypes not accommodated by the model. When applied to dense single nucleotide polymorphism (SNP) markers from population data, our method obtains a better-fitting and more parsimonious model than the leading method.

In addition, we note that these models represent a VOMC defined in a single direction along the genome, which ignores the LD structure that could be represented by conditional independencies in the opposite direction. Therefore, fitting the model to the same data in the reverse direction along the genome usually results in different haplotype frequency estimates, which is an undesirable property for genomic models. We develop a method of reconciling two DAG models fit in opposite directions along the genome that takes advantage of the differing LD structure represented in both models to derive a new bidirectional model.

When trying to detect segments of identity by descent (IBD) among individuals, background LD can be a source of noise that obfuscates haplotypic similarity due to recent coancestry. Methods of IBD segment detection that do not account for LD can have a high false positive rate. We introduce a method for IBD segment detection using a hidden Markov model (HMM) that incorporates a DAG model in the hidden layer to adjust for LD. Unlike similar methods, ours models the full set of 15 IBD states among the four chromosomes of two individuals. When applied to simulated dense SNP marker data, our method provides more accurate IBD segment detection than other leading methods.

TABLE OF CONTENTS

Page

List of Figures iii

List of Tables xii

Glossary xiv

Chapter 1: Introduction 1

Chapter 2: Haplotype frequency modeling 7
2.1 Variable-order Markov chains 7
2.2 Directed acyclic graphical models 9

Chapter 3: Model estimation and selection 14
3.1 Likelihood estimation 14
3.2 Model selection 16
3.3 Selection of regularization parameter 24
3.4 1000 Genomes model selection example 27

Chapter 4: DAG model reversibility and reconciliation 38
4.1 Introduction and motivation 38
4.2 Vertex completeness and reversibility 40
4.3 Method for bidirectional model reconciliation 51
4.4 1000 Genomes reconciliation example 59

Chapter 5: IBD segment detection 65
5.1 Previous methods for IBD segment detection 65
5.2 Hidden Markov model for 15 IBD states 69
5.3 1000 Genomes IBD segment detection example 77

Chapter 6: Conclusion 92
6.1 Contributions 92
6.2 Future work 93

Appendix A: MRF parameter calculations 101

Appendix B: IBD figures 107

LIST OF FIGURES

Figure Number Page

1.1 *A simplified example of meiotic recombination. Two pairs of sister chromatids on the left are distinguished by color and have genetic markers at four loci with alleles A/a, B/b, C/c, and D/d. Before genetic recombination, the two blue chromatids have haplotypes ABCD, and the two yellow chromatids have haplotypes abcd. The two pairs of chromatids on the right show the result after one recombination event in the outer homologue partners between markers A/a and B/b and two recombination events in the inner homologue partners, one between markers B/b and C/c and one between markers C/c and D/d. From left to right, the new chromatids have haplotypes aBCD, abCd, ABcD, and Abcd.* 2

2.1 *An example DAG which models haplotype frequencies at four biallelic markers. Solid lines represent edges for allele 1, and dashed lines represent edges for allele 2.* . . 12

2.2 *Example DAGs representing the independence model along with first and second order Markov models for biallelic markers. Solid lines represent edges for allele 1, and dashed lines represent edges for allele 2.* 13

3.1 *Example of merging two vertices within a saturated tree DAG model. Solid lines represent edges for allele 1, and dashed lines represent edges for allele 2.* 19

3.2 *DAG model selection results at various values of the regularization parameter λ for our likelihood-based method and the scale parameter for Beagle v3.3. Figure (a) shows validation loss, and (b) shows AIC. The likelihood-based DAGs were fit in both directions along the chromosome, but their red and blue lines almost completely overlap into purple lines. Beagle results are represented by a green line. The plus sign and dotted lines represent the values at which validation loss is minimized.* . . 30

3.3 *DAG model selection results at various values of the regularization parameter λ for our likelihood-based method and the scale parameter for Beagle v3.3. Figure (a) shows log-likelihood, and (b) shows total number of vertices. The likelihood-based DAGs were fit in both directions along the chromosome, but their red and blue lines almost completely overlap into purple lines. Beagle results are represented by a green line. The plus sign and dotted lines represent the values at which validation loss is minimized.* 31

3.4 *DAG model selection results at various values of the regularization parameter λ for our likelihood-based method and the scale parameter for Beagle v3.3. Figure (a) shows proportion of vertices with two edges, and (b) shows log-likelihood against average number of vertices per marker. The likelihood-based DAGs were fit in both directions along the chromosome, but their red and blue lines almost completely overlap into purple lines. Beagle results are represented by a green line. The plus sign and dotted lines represent the values at which validation loss is minimized.* . . 33

3.5 *DAG model selection results at the value of the regularization parameter for our likelihood-based method and the scale parameter for Beagle v3.3 that minimizes validation loss. Solid lines show the number of vertices over a 10,000 marker moving average by chromosome position as measured in cM. Dashed lines show the average number of vertices per marker for each model. The likelihood-based DAGs were fit in both directions along the chromosome, but their red and blue lines almost completely overlap into purple lines. Beagle results are represented by a green line.* 35

3.6 *DAG model selection results at the value of the regularization parameter for our likelihood-based method and the scale parameter for Beagle v3.3 that minimizes validation loss. Solid lines show the log-likelihood and validation loss contributions over a 10,000 marker moving average by chromosome position as measured in cM. The likelihood-based DAGs were fit in both directions along the chromosome, but their red and blue lines almost completely overlap into purple lines. Beagle results are represented by a green line.* 36

4.1 *Cross section from markers 12,655 to 12,664 of DAG fit in the "forward" direction along the chromosome using regularization parameter $\lambda = 8$. Each vertex is proportional in area to the number of haplotypes it contains and includes a pie chart of the proportions of haplotypes belonging to the five superpopulations. Solid lines represent edges for allele 1, and dashed lines represent edges for allele 2. A plurality European path is outlined in green.* 41

4.2 *Cross section from markers 12,655 to 12,664 of DAG fit in the "backward" direction along the chromosome using regularization parameter $\lambda = 8$. Each vertex is proportional in area to the number of haplotypes it contains and includes a pie chart of the proportions of haplotypes belonging to the five superpopulations. Solid lines represent edges for allele 1, and dashed lines represent edges for allele 2. A plurality Admixed American path is outlined in purple.* 42

4.3 *Example DAGs which model haplotype frequencies at four biallelic markers. Figure (a) is in the "forward" direction with edges going from left to right. Figure (b) is in the "backward" direction with edges going from right to left. Solid lines represent edges for allele 1, and dashed lines represent edges for allele 2. These DAGs were constructed to represent different sets of conditional independencies and would not necessarily model equivalent haplotype frequencies from data.* 48

4.4 *Example DAGs which model haplotype frequencies at four biallelic markers along with edge weights derived from the MRF frequencies using conditional probabilities. Figure (a) is in the "forward" direction with edges going from left to right. Figure (b) is in the "backward" direction with edges going from right to left. Solid lines represent edges for allele 1, and dashed lines represent edges for allele 2.* 57

4.5 *Example DAGs which model haplotype frequencies at four biallelic markers along with edge weights derived from reconciling the DAGs from Figure 4.4. Figure (a) is in the "forward" direction with edges going from left to right. Figure (b) is in the "backward" direction with edges going from right to left. Solid lines represent edges for allele 1, and dashed lines represent edges for allele 2.* 58

4.6 *The number of vertices at each level of the "forward" DAG during reconciliation of the first 100 markers in the 1000 Genomes data. The red line is for when new haplotype paths branch off without restriction. The blue line is for when new paths rejoin the DAG when memory is exhausted. The purple, green, and yellow lines are for when new paths rejoin within a maximum path length of 40, 20, and 10 markers, respectively.* 60

4.7 *The log-likelihood and average number of vertices per marker of the reconciled bidirectional model at various values of the maximum path length. The green dashed line shows the average log-likelihood of the "forward" and "backward" DAG models. The red dashed line shows the log-likelihood under independent markers.* 64

5.1 *Example transmission of IBD through a pedigree. Founder haplotypes are assigned twelve distinct colors. IBD segments exist in the first cousins at the bottom of the pedigree where they share the same founder haplotype.* 66

5.2 *Pedigree used for the simulated example to test IBD segment detection methods. Siblings 31 and 32 from sibling parents 21 and 22 were used to obtain a greater variety of possible IBD states. The kinship coefficient between these two individuals is 5/16 = 0.3125.* 78

5.3 *IBD segment detection results from our Viterbi algorithm run in both directions compared to results from Beagle v4.1. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method. Parameters for the Viterbi algorithm are displayed for each result.* 80

5.4 *IBD segment detection results from our Viterbi algorithm when using the collapsed DAG with no LD structure ($\lambda = \infty$). Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method. Parameters for the Viterbi algorithm are displayed for each result.* 82

5.5 *IBD segment detection results from our Viterbi algorithm run in both directions compared to results from Beagle v4.1. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method. Parameters for the Viterbi algorithm are displayed for each result.* 84

5.6 *IBD segment detection results from our Viterbi algorithm run after preprocessing by error correction. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method. Parameters for the Viterbi algorithm are displayed for each result.* 85

5.7 *The distributions of estimator biases $\hat{\Phi}_j - \Phi_j$ used to calculate the coefficients of variance of RMSE reported in Table 5.4. The red boxplots represent the full 3,200/cM panel while the blue ones represent the pruned 75/cM marker panel. Outliers are plotted as circles outside the whiskers of the boxplots.* 90

B.1 *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at $\alpha = 10^{-2}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 108

B.2 *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at $\alpha = 10^{-3}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 109

B.3 *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at $\alpha = 10^{-4}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 110

B.4 *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at $\alpha = 10^{-5}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 111

B.5 *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at $\alpha = 10^{-6}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 112

B.6 *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at $\alpha = 10^{-7}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 113

B.7 *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-2}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 114

B.8 *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-3}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 115

B.9 *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-4}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 116

B.10 *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-5}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 117

B.11 *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-6}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 118

B.12 *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-7}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 119

B.13 *IBD segment detection results from our Viterbi algorithm when using a completely collapsed DAG with no LD structure ($\lambda = \infty$) at $\alpha = 10^{-6}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 120

B.14 *IBD segment detection results from our Viterbi algorithm when using a completely collapsed DAG with no LD structure ($\lambda = \infty$) at $\alpha = 10^{-8}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 121

B.15 *IBD segment detection results from our Viterbi algorithm when using a completely collapsed DAG with no LD structure ($\lambda = \infty$) at $\alpha = 10^{-10}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 122

B.16 *IBD segment detection results from our Viterbi algorithm when using a completely collapsed DAG with no LD structure ($\lambda = \infty$) at $\alpha = 10^{-12}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 123

B.17 *IBD segment detection results from our Viterbi algorithm when using a completely collapsed DAG with no LD structure ($\lambda = \infty$) at $\alpha = 10^{-14}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 124

B.18 *IBD segment detection results from our Viterbi algorithm when using a completely collapsed DAG with no LD structure ($\lambda = \infty$) at $\alpha = 10^{-16}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 125

B.19 *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at $\alpha = 10^{-4}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 126

B.20 *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at $\alpha = 10^{-5}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 127

B.21 *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at* $\alpha = 10^{-6}$ *and various values of* β *and* ϵ*. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal* λ *regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 128

B.22 *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at* $\alpha = 10^{-7}$ *and various values of* β *and* ϵ*. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal* λ *regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 129

B.23 *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at* $\alpha = 10^{-8}$ *and various values of* β *and* ϵ*. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal* λ *regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 130

B.24 *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at* $\alpha = 10^{-9}$ *and various values of* β *and* ϵ*. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal* λ *regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 131

B.25 *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at* $\alpha = 10^{-4}$ *and various values of* β *and* ϵ*. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal* λ *regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 132

B.26 *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at* $\alpha = 10^{-5}$ *and various values of* β *and* ϵ*. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal* λ *regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 133

B.27 *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-6}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 134

B.28 *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-7}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 135

B.29 *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-8}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 136

B.30 *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-9}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 137

B.31 *IBD segment detection results from Beagle v4.1 at various values of minimum LOD score. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 138

B.32 *IBD segment detection results from Beagle v4.1 at various values of minimum LOD score. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.* 139

LIST OF TABLES

Table Number Page

4.1 *Relationship between the "forward" and "backward" DAG models from Figure 4.3. The full set of haplotypes contained by each vertex is listed from left to right in the "forward" direction and right to left in the "backward". The complete subsets and the resultant reversed complete subsets are derived along with any vertices from the opposing DAG that contain these subsets. Any parameter restrictions in the MRF model that are imposed by the vertex are given.* 50

4.2 *Haplotype frequencies for each of the 16 possible haplotypes derived from the MRF model, the "forward" and "backward" DAG models, the reconciled DAG model and under independent markers. Multilocus LD is computed as the ratio of relative entropy between each distribution and the independent distribution to the maximum possible relative entropy.* 56

5.1 *Various representations of the 15 IBD states at a single locus for the four chromosomes of two individuals. In the state diagrams, the top dots represent the DNA copies for individual 1, and the bottom represent those for individual 2, and colors represent different patterns of IBD, red for all four individuals, green for a group of three, yellow for two pairs, and blue for a single pair (Glazner, 2014). In the partition notation, a and b represent the DNA copies for individual 1, and c and d represent those for individual 2. This kinship coefficient is the probability that a pair of randomly selected DNA copies from each individual are IBD.* 67

5.2 *Unscaled IBD state transition rate matrix resulting from the modified Chinese restaurant process on four chromosomes. The parameter β is the pairwise IBD probability, and $\eta = 1 - \beta$. The diagonal entries x_{jj} are not shown, but are defined such that the sums of the rows equal zero. That is, $x_{jj} = -\sum_{i \neq j} x_{ji}$ for all $j = 1, \ldots, 15$.* . . 74

5.3 *All possible edge switches that allow for maintenance of each of the 15 IBD states. For each set of equivalences, the bolded haplotype can have its allele switched in order to conform to the corresponding IBD state. If two haplotypes are bolded, then either allele could be switched. For example, to achieve the IBD state (ab)(cd) if the alleles are such that* $a = b = \mathbf{c} \neq \mathbf{d}$*, then either allele* **c** *can be switched to be equal to* **d***, or allele* **d** *can be switched to be equal to* **c***; hence they are both bolded. Where an allele is missing from a set of equivalencies, its relationship to the other alleles is irrelevant.* 76

5.4 *Performance of various kinship estimators as measured by 100 times the NRMSE on both the full 3,200/cM and the pruned 75/cM marker panels. The types of genetic information used for each estimator are given (Wang, 2019). The transition rate parameter* $\alpha = 10^{-4}$ *was used for the DAG HMM model, and for the no LD HMM model,* $\alpha = 10^{-14}$ *was used. The final line gives values based on pedigree kinship, 0.25 for siblings and 0.0039 for third cousins.* 88

GLOSSARY

AIC: Akaike information criterion

CM: centimorgan

DAG: directed acyclic graph

EM: expectation-maximization

GRM: genomic relationship matrix

HBD: homozygosity/homozygous by descent

HMM: hidden Markov model

IBD: identity/identical by descent

IBS: identity/identical by state

LD: linkage disequilibrium

MLE: maximum likelihood estimate

MRF: Markov random field

NRMSE: normalized root mean squared error

SNP: single nucleotide polymorphism

VCF: variant call format

VOMC: variable-order Markov chain

ACKNOWLEDGMENTS

First and foremost, I express my deepest gratitude to my advisor, Elizabeth Thompson, whose patience is eclipsed only by the depth and breadth of her knowledge and experience. Her support, financial, academic, and otherwise, made this all possible. Many thanks also to the rest of my committee members, past and present, Sharon Browning, Thomas Richardson, Jonathan Wakefield, Philip Green, and Mathias Drton, whose support for my project was unwavering through the years. Notably, it was my introduction to Sharon and her own research on haplotype models that inspired much of this work.

I have benefitted greatly from interactions with members of the Thompson group, including Steven Lewis, Ellen Wijsman, Fiona Grimson, Jesse Raffa, Jon Ranola, Serge Sverdlov, Bowen Wang, and Saonli Basu. Additional thanks are due to the members of UW's statistical genetics and population genetics seminars. I learned a tremendous amount from talking with and listening to all of you.

Thanks to all of my coworkers at the Seattle Epidemiologic Research and Information Center (ERIC). You have been a great sounding board for my ideas, my ramblings, and my grumblings through the latter years of this undertaking.

Finally, I want to thank my wife Shannon. I don't envy anyone having to deal with me on a day-to-day basis over the last few months, but she did that and more. I apologize for all of the pointless knowledge you had to learn in order to help me write and edit.

DEDICATION

to my partner in life and love, Shannon, and our adjectiveless daughter, Isabel, both of whom have known me only as a student

to Stella, Lily, Jiji, and Nebula for keeping my lap warm

Chapter 1

INTRODUCTION

Haplotypes are groups of alleles at multiple loci on a single chromosome that tend to be inherited together from a single parent. Genes found along a chromosome are not passed on to offspring independently from one another. Gametes produced through meiosis undergo genetic recombination where DNA sequences are reshuffled by the breaking and rejoining of chromosome segments. A consequence of this process is that recombination events occur at a rate roughly proportional to the physical distance between markers (Carroll, 2013). This rate of recombination is often used as its own measure of genetic distance called the centimorgan (cM), where 1 cM is defined as the distance between two loci for which 0.01 recombination events are expected. In the human genome, this is approximately equal to 1 million base pairs on average. Figure 1.1 shows an example of genetic recombination and the resultant haplotypes.

If two markers, one with alleles A/a and the other with alleles B/b did segregate independently from one another during meiosis, then we would expect the frequency of haplotype AB in the gametes to equal $p_{\mathrm{A}}p_{\mathrm{B}}$, where p_{A} and p_{B} are the allele frequencies for alleles A and B, respectively. However, if the marker loci are linked, then the true frequency of haplotype AB, denoted by p_{AB}, may differ from the product of the individual allele frequencies. This difference $D = p_{\mathrm{AB}} - p_{\mathrm{A}}p_{\mathrm{B}}$ is a quantification of linkage disequilibrium (LD), a measure of the degree to which alleles at different loci are non-randomly associated. LD can arise not just in the presence of genetic linkage but also of other factors such as natural selection, genetic drift, population structure, and inbreeding (Slatkin, 2008). The difference D specifically measures pairwise LD between two marker loci, but methods also exist for measuring LD from haplotype frequency distributions between markers at multiple loci.

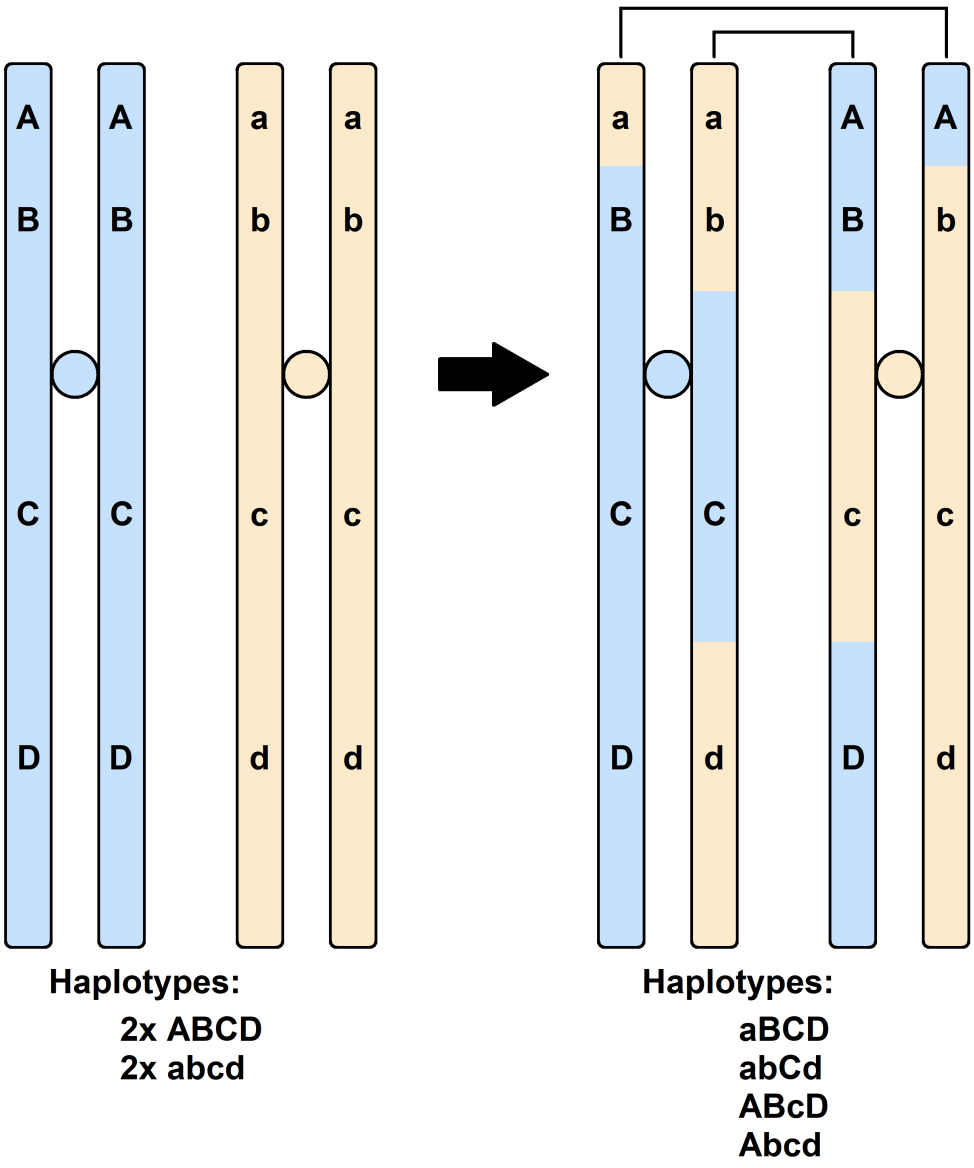

Figure 1.1: *A simplified example of meiotic recombination. Two pairs of sister chromatids on the left are distinguished by color and have genetic markers at four loci with alleles A/a, B/b, C/c, and D/d. Before genetic recombination, the two blue chromatids have haplotypes ABCD, and the two yellow chromatids have haplotypes abcd. The two pairs of chromatids on the right show the result after one recombination event in the outer homologue partners between markers A/a and B/b and two recombination events in the inner homologue partners, one between markers B/b and C/c and one between markers C/c and D/d. From left to right, the new chromatids have haplotypes aBCD, abCd, ABcD, and Abcd.*

Haplotypes have become an integral part of analyses involving fine-scale genetic data. Because they represent a group of alleles at multiple loci that are inherited together with little chance of being broken up by recent recombination, haplotypes can convey much more information about surrounding genes than the individual alleles alone. Studies have asserted that this additional information about the correlation between alleles, about LD structure, can be used to improve upon methods that treat multiple loci independently from one another. For example, combining markers with common alleles into a single haplotypic marker with multiple rare alleles can increase the power of LD-based association tests (Akey et al., 2001; Morris & Kaplan, 2002). Accommodating LD has also been seen to increase the accuracy of models used to identify tracts of identity by descent (IBD) between individuals (Albrechtsen et al., 2009; Browning & Browning, 2013; Han & Abney, 2011). Similarly, the structure of LD has been leveraged for better inference of local ancestry in admixed populations (Baran et al., 2012; Price et al., 2009).

An early step in analyses using haplotype data often involves inferring the haplotype phase. Marker alleles are typed on two copies of each chromosome, but when heterozygosity is present, it may not be known which allele belongs to which chromosome, information that is necessary for constructing haplotypes. An expectation-maximization (EM) algorithm can be used to resolve this ambiguity based on observed LD and assuming Hardy-Weinberg proportions (Excoffier & Slatkin, 1995). For a small number of markers, once phase can be determined, haplotype frequencies can be estimated by simply counting all haplotypes present in a sample. However, one major problem is that as the density of marker panels and the availability of whole-genome sequencing increases, haplotypes can become effectively unique among genomes in a sample. Methods exist to split haplotypes into usable blocks determined from the block-like patterns of LD, but true LD structure is more sophisticated than these patterns, allowing information to persist across blocks (Greenspan & Geiger, 2004; Zhang et al., 2003).

We propose a new method for estimating haplotype frequency models derived from variable-order Markov chains (VOMC). Because these Markov chain models do not have

a fixed order, they are excellent for modeling LD since they allow for longer memory in regions with high LD between markers and shorter memory in regions with low LD. The underlying models we explore are the same as those used by the Beagle software, which provides applications for haplotype phasing, imputation of ungenotyped variants, and IBD segment detection (Browning, 2006, 2008; Browning & Browning, 2007). However, we introduce a likelihood-based method for estimating such models which improves upon current methods and admits the use of regularization and validation for model selection.

We also propose a method for reconciling the estimates for VOMC haplotype frequency models fit in opposite directions along a chromosome. By nature, these models are defined in a particular direction, so the resulting frequency estimates depend on the orientation of the marker input. Since directions along chromosomes are arbitrary, this is an unfortunate side effect of using these models. Our novel method attempts to solve this problem by reconciling two models from opposite directions into a single bidirectional model which leverages the information about LD structure gained by conditional dependencies represented in both models instead of choosing between one or the other.

Finally, we propose a method of IBD segment detection that accomodates transitions among the full set of 15 IBD states across the four chromosomes of two individuals while accounting for LD as modeled by a VOMC. Brown et al. (2012) investigated the impact of LD on IBD segment detection and observed that not accounting for it can result in high false positive rates of IBD detection. While the Beagle software does include methods for IBD segment detection that incorporate LD, it reduces the IBD states between two individuals to an IBD or non-IBD binary, ignoring possibilities that can arise in the presence of consanguinity. We expand upon these methods to allow for the higher granularity of the full set of IBD states.

In Chapter 2, we introduce the notation and nomenclature necessary to discuss haplotype frequency estimation using VOMC models. We explain how the variable-order Markov property provides a good framework for modeling LD by allowing the length of its memory to change along a chromosome. We describe how a VOMC model can be represented by a

directed acyclic graph (DAG) and how the vertices and edges of this DAG correspond to haplotypes and their frequencies. Finally, we define context-sensitive conditional independencies and show how they are incorporated by DAG models.

In Chapter 3, we formalize maximum likelihood estimation (MLE) on DAG models given a sample of haplotypes. We then introduce our method for DAG model selection based on regularized likelihood optimization, including analysis of the computational issues and limitations. Our method for selection of the regularization parameter is also discussed along with its statistical implications. Finally, we apply our methods to high-density marker data obtained from the 1000 Genomes Project, and compare the results with those from the Beagle software.

In Chapter 4, we investigate how the arbitrary orientation of the markers along a chromosome can result in different DAG models depending on the directions in which they are fit. We explore the ways in which DAGs from opposite directions can express disparate sets of context-sensitive independencies, and we attempt to add mathematical rigor to this exploration by introducing the concepts of vertex completeness and reversibility. We then describe our method for reconciling these differences into a bidirectional model that incorporates the independencies from DAGs in each direction. Finally, we test this reconciliation method on DAG models fit on the 1000 Genomes Project data, including a discussion on issues arising from scaling to a large number of markers.

In Chapter 5, we review previous methods for the detection of IBD segments before introducing our own method that synthesizes and iterates upon them. We describe our hidden Markov model (HMM) for the full set of 15 IBD states on four chromosomes that incorporates a DAG model into the hidden layer to account for LD. We discuss the computational limits of inference from this HMM and how we reduced the number of possible hidden states in order to maintain computational tractability. Then we test our method on data simulated from the 1000 Genomes Project data, again comparing our results with those from the Beagle software as well as a number of other methods.

Finally, in Chapter 6, we summarize the original contributions to the literature explored

within this thesis. We claim that our methods both improve upon the performance of existing methods as well as increase understanding of the theories underlying those methods. We finish with a discussion of areas for further development of our methods and some possible directions for future research.

Chapter 2

HAPLOTYPE FREQUENCY MODELING

2.1 Variable-order Markov chains

Consider a sample of N haplotypes $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N$, where each haplotype

$$\mathbf{X}_i = (X_{i1}, X_{i2}, \ldots, X_{iM})$$

is a sequence of M biallelic single-nucleotide polymorphism (SNP) markers at loci ordered along a chromosome. When the number of markers M is small, haplotype frequencies in the sampled population can be easily and accurately estimated by a simple count of each observed sequence. By essentially allowing a single parameter for each unique haplotype in the sample, this saturated model accounts for any dependence between alleles at different loci in the form of linkage disequilibrium (LD). However, since the number of possible unique haplotypes is 2^M, it does not require too many markers before these simple counts are no longer effective for estimation.

One commonly used alternative for modeling frequencies when the number of markers is high, lies at the other extreme. If the alleles at each loci are assumed to be independent, the haplotype frequencies factor completely as:

$$\Pr(\mathbf{X} = \mathbf{x}) = \prod_{\ell=1}^{M} \Pr(X_\ell = x_\ell),$$

where the allele frequencies can be estimated accurately from the sample. This independence model ignores all LD and can give wildly inaccurate frequency estimates when markers are dense and highly dependent. We will show in a later chapter how using this model leads to major problems in the detection of segments of identity by descent (IBD) between haplotypes of dense markers.

A natural model for incorporating dependence across the sequence of markers is a Markov chain. Since allele frequencies and transition probabilities differ along the chromosome, any such Markov chain must be inhomogeneous. This leads to the haplotype frequencies factoring into conditional probabilities as follows:

$$\Pr(\mathbf{X} = \mathbf{x}) = \prod_{\ell=1}^{M} \Pr(X_\ell = x_\ell | X_{\ell-1} = x_{\ell-1}, X_{\ell-2} = x_{\ell-2}, \ldots, X_{\ell-k} = x_{\ell-k}),$$

where the fixed order k of the Markov chain represents the memory length of the process. These models have the benefit of incorporating LD while still keeping the number of parameters from growing exponentially with the number of markers.

Bühlmann and Wyner (1999) detail two major problems highlighting why fixed-order Markov chains are not always appropriate for estimation. First, the class of these models is not structurally rich. In our biallelic SNP example, a Markov chain of order k modeling frequencies for haplotypes of length M requires approximately $M2^k$ parameters. After the initial k markers, a model of order $k = 5$ will have 32 parameters per marker while a model of order $k = 6$ will have 64 parameters, but there is no possibility of a model in between with, say, 50 parameters per marker. This lack of granularity can make it difficult to find a fixed order that balances the trade-off between bias and variance. Second, since the number of parameters grows exponentially with the order k, only Markov chains with a relatively small fixed-order can be considered before the curse of dimensionality makes efficient estimation impossible.

Due to differences in marker spacing and recombination rates across a chromosome, it is understandable why fixed-order Markov chains would not be appropriate for modeling haplotype frequencies. Instead, we extend the class of Markov chain models to allow the length of the memory to change depending on the context of the memory itself. Specifically, a variable-order Markov chain (VOMC) has transition probabilities of the form

$$\begin{aligned}\Pr(X_\ell = x_\ell | X_{\ell-1} = x_{\ell-1}, X_{\ell-2} = x_{\ell-2}, \ldots) \\ = \Pr(X_\ell = x_\ell | X_{\ell-1} = x_{\ell-1}, X_{\ell-2} = x_{\ell-2}, \ldots, X_{\ell-\mathcal{K}} = x_{\ell-\mathcal{K}}),\end{aligned}$$

where the variable-order of the chain $\mathcal{K} = \mathcal{K}(x_{\ell-1}, x_{\ell-2}, \ldots)$ is itself a function of the previously observed variables, often called the context. For example, if the haplotype $\mathbf{X} = (X_1, X_2, \ldots, X_M)$ has the following properties at locus ℓ:

$$\begin{aligned}
&\Pr(X_\ell = 1 | X_{\ell-1} = 1, X_{\ell-2}, X_{\ell-3}, \ldots) \\
&\qquad = \Pr(X_\ell = 1 | X_{\ell-1} = 1), \\
&\Pr(X_\ell = 1 | X_{\ell-1} = 2, X_{\ell-2} = 1, X_{\ell-3}, \ldots) \\
&\qquad = \Pr(X_\ell = 1 | X_{\ell-1} = 2, X_{\ell-2} = 1), \quad \text{and} \\
&\Pr(X_\ell = 1 | X_{\ell-1} = 2, X_{\ell-2} = 2, X_{\ell-3} = 1, \ldots) \\
&\qquad = \Pr(X_\ell = 1 | X_{\ell-1} = 2, X_{\ell-2} = 2, X_{\ell-3} = 1) \\
&\qquad \neq \Pr(X_\ell = 1 | X_{\ell-1} = 2, X_{\ell-2} = 2, X_{\ell-3} = 2),
\end{aligned}$$

then we have constructed a VOMC where the distribution of X_ℓ has a memory of length one under the context $X_{\ell-1} = 1$, a memory of length two under the context $X_{\ell-1} = 2$ and $X_{\ell-2} = 1$, and a memory of length three under the context $X_{\ell-1} = 2$ and $X_{\ell-2} = 2$.

This larger class of models addresses both of the aforementioned problems. Because the length of the memory of the chain is allowed to vary along different sections of the chromosome depending on the context, VOMCs include many models that fall in between the large gaps of increasingly fixed-order Markov chains. Also, dependencies with long memories can be modeled in VOMCs by restricting them to specific contexts. The number of parameters need not expand exponentially when they do not need to be included for every possible context of a fixed order. In the example above, modeling the distribution of X_ℓ would only require four parameters instead of the eight required for a Markov chain with fixed order three.

2.2 Directed acyclic graphical models

In terms of graphical models, a VOMC can be represented by a directed acyclic graph (DAG). Specifically, we can define a weighted DAG $G = (V, E, w)$ with labeled edges. The vertex set

$V = \{v_{\ell,j}\}$ is indexed first by the topological ordering of the DAG from $\ell = 0, \ldots, M$ then by the number of vertices at each level. The edge set $E = \{e_{\ell,j,k} = (v_{\ell-1,j}, v_{\ell,j'}, k)\}$ contains ordered triples denoting the source vertex, the destination vertex, and the allele label of each edge. An individual edge $e_{\ell,j,k}$ in the DAG represents a marker at locus ℓ having allele k, so a haplotype $\mathbf{x} = (x_1, x_2, \ldots, x_M)$ is represented by a path $(e_{1,j_1,x_1}, e_{2,j_2,x_M}, \ldots, e_{M,j_M,x_M})$ through the DAG. The set of vertices at level ℓ form a partition of the possible haplotypes derived from the first ℓ markers, and an individual vertex $v_{\ell,j} = \{\mathbf{x}_{1:\ell}\}$ represents the set of haplotypes of length ℓ with paths that terminate in that vertex. These sets will serve as contexts in the underlying VOMC.

Note that the indices of an edge $e_{\ell,j,k} = (v_{\ell-1,j}, v_{\ell,j'}, k)$ do not contain a reference to the destination vertex $v_{\ell,j'}$. This is because for each allele k there can only be a single edge leaving any vertex, so the edge $e_{\ell,j,k}$ is well-defined without an index for j'. However, we may occasionally find it useful to refer to the destination vertex of an edge, so we define the function $d : E \to \mathbb{N}$ by

$$d(e_{\ell,j,k}) = d(v_{\ell-1,j}, v_{\ell,j'}, k) = j', \tag{2.1}$$

so each edge is mapped to the index of its destination. To somewhat simplify notations, the destination $d(e_{\ell,j,k})$ will often be written as $d(\ell, j, k)$ a function of the edge indices instead of the edge itself.

The edge set forms the domain of the weight function $w : E \to [0, 1]$, where the weight of each edge is the conditional probability of that edge's allele given the context defined by the vertex it comes from. That is,

$$w(e_{\ell,j,k}) = \Pr(X_\ell = k | \mathbf{X}_{1:\ell-1} \in v_{\ell-1,j}). \tag{2.2}$$

Defining the weight function in this way means that the set of haplotypes of length ℓ represented by a vertex share the same conditional distribution of marker frequencies at loci greater than ℓ. We also define the weight W of a vertex as the total probability of all haplotypes included within the vertex. Note that this joint probability can be computed by the

sum of the products of the edge weights along each path entering the vertex, so we have

$$W(v_{\ell,j}) = \Pr(\mathbf{X}_{1:\ell} \in v_{\ell,j}) = \sum_{\mathbf{x}_{1:\ell} \in v_{\ell,j}} \prod_{t=1}^{\ell} w(e_{t,j(\mathbf{x}_{1:t-1}),x_t}), \tag{2.3}$$

where $j(\mathbf{x}_{1:t-1})$ is the index of the vertex at level $t-1$ containing the haplotype $\mathbf{x}_{1:t-1}$. The weight function w serves as the set of parameters for the DAG model, and in the next chapter we will see how these can be estimated from a sample of haplotypes. Note that the sum of the conditional probabilities along edges leaving a single vertex must equal one, so for biallelic SNP markers, $w(e_{\ell,j,2}) = 1 - w(e_{\ell,j,1})$, which means that the number of parameters in a model reduces to the total number of vertices (excluding the single terminal vertex at level M).

Figure 2.1 shows an example of a DAG modeling haplotypes created from four biallelic markers. There is only a single vertex $v_{0,1}$ at level zero representing the empty set of haplotypes of length zero. There is also only a single vertex $v_{4,1}$ at level four representing the set of all haplotypes of length four which trivially have the same distribution of marker frequencies at further loci. The haplotype 1212 is represented by the path $(e_{1,1,1}, e_{2,1,2}, e_{3,2,1}, e_{4,3,2})$ through the edges corresponding to its marker alleles. The vertex $v_{2,1}$ represents the set of haplotypes $\{11, 21\}$ for which the distribution of (X_3, X_4) is identical given $(X_1, X_2) \in v_{2,1}$. Likewise, the vertex $v_{3,3}$ represents the set of haplotypes $\{121, 122, 221, 222\}$ for which the distribution of X_4 is identical.

When multiple edges in a DAG enter the same vertex, this indicates a merging of vertices corresponding to the union of the sets of haplotypes they represent. Probabilistically, this merging imposes restrictions on the model parameters in the form of context-sensitive conditional independencies. Explicitly, for every pair of haplotypes $\mathbf{x}_{1:\ell}, \mathbf{x}'_{1:\ell} \in v_{\ell,j}$, we have

$$(\mathbf{X}_{\ell+1:M} | \mathbf{X}_{1:\ell} = \mathbf{x}_{1:\ell}) \stackrel{d}{=} (\mathbf{X}_{\ell+1:M} | \mathbf{X}_{1:\ell} = \mathbf{x}'_{1:\ell}),$$

so $\mathbf{X}_{1:\ell}$ is conditionally independent of $\mathbf{X}_{\ell+1:M}$ given the context $\mathbf{X}_{1:\ell} \in v_{\ell,j}$. For example, since the vertex $v_{3,3}$ in Figure 2.1 represents the set of haplotypes $\{121, 122, 221, 222\}$, we have that $\mathbf{X}_{1:3} \perp\!\!\!\perp X_4 | X_2 = 2$. Note that this differs from the more general conditional

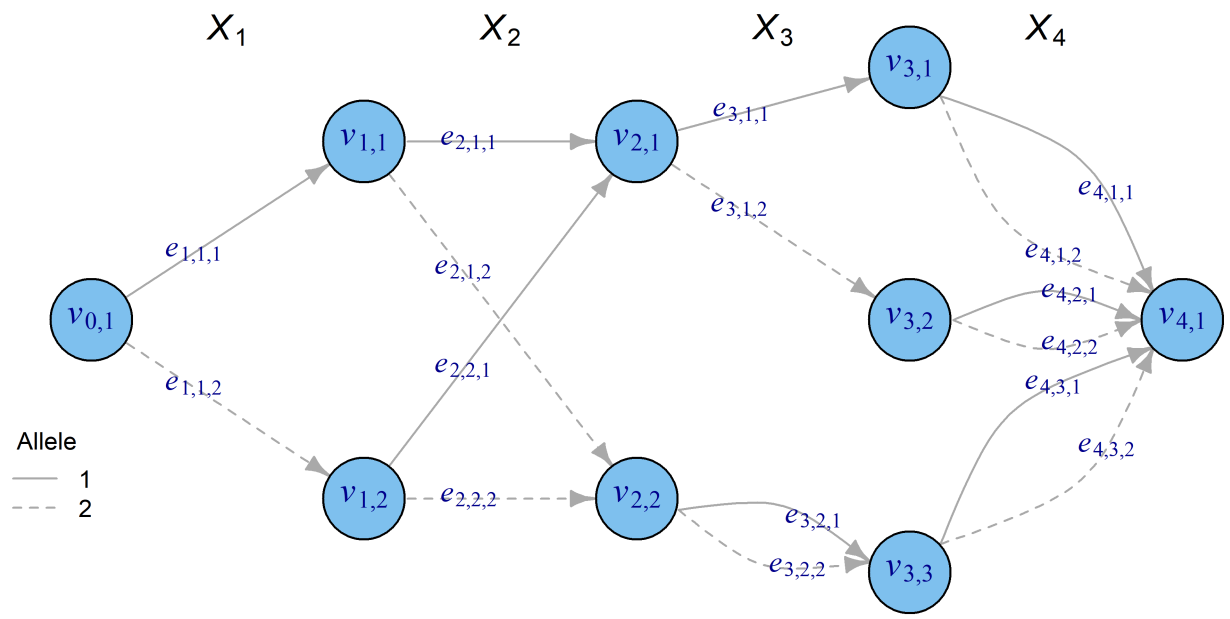

Figure 2.1: *An example DAG which models haplotype frequencies at four biallelic markers. Solid lines represent edges for allele 1, and dashed lines represent edges for allele 2.*

independence $\mathbf{X}_{1:3} \perp\!\!\!\perp X_4 | X_2$ since in this example, when $X_2 = 1$, the distribution of X_4 does depend on X_3 as well. Allowing for these sorts of context-sensitive independencies is what affords DAG models the granularity to form a much larger class than fixed order Markov models (Boutilier et al., 1996).

In addition to representing variable-order Markov chains, DAG models can include all of the previously discussed models as well. The saturated model induced from counting all unique haplotypes can be represented by a tree in which each vertex has only a single path entering it. This model has no merged vertices and therefore no conditional independencies. The independence model, on the other hand, is the result of all vertices being merged into a single vertex at each level. The set of contexts in this single vertex includes every possible haplotype up to that locus, so the context-sensitive independencies reduce to $\mathbf{X}_{1:\ell} \perp\!\!\!\perp \mathbf{X}_{\ell+1:M}$ at each ℓ, and all markers become independent. Fixed order Markov chains of order k can be represented by DAGs with 2^k vertices at each level. Figure 2.2 shows examples of DAGs representing the independence model along with first and second order Markov models.

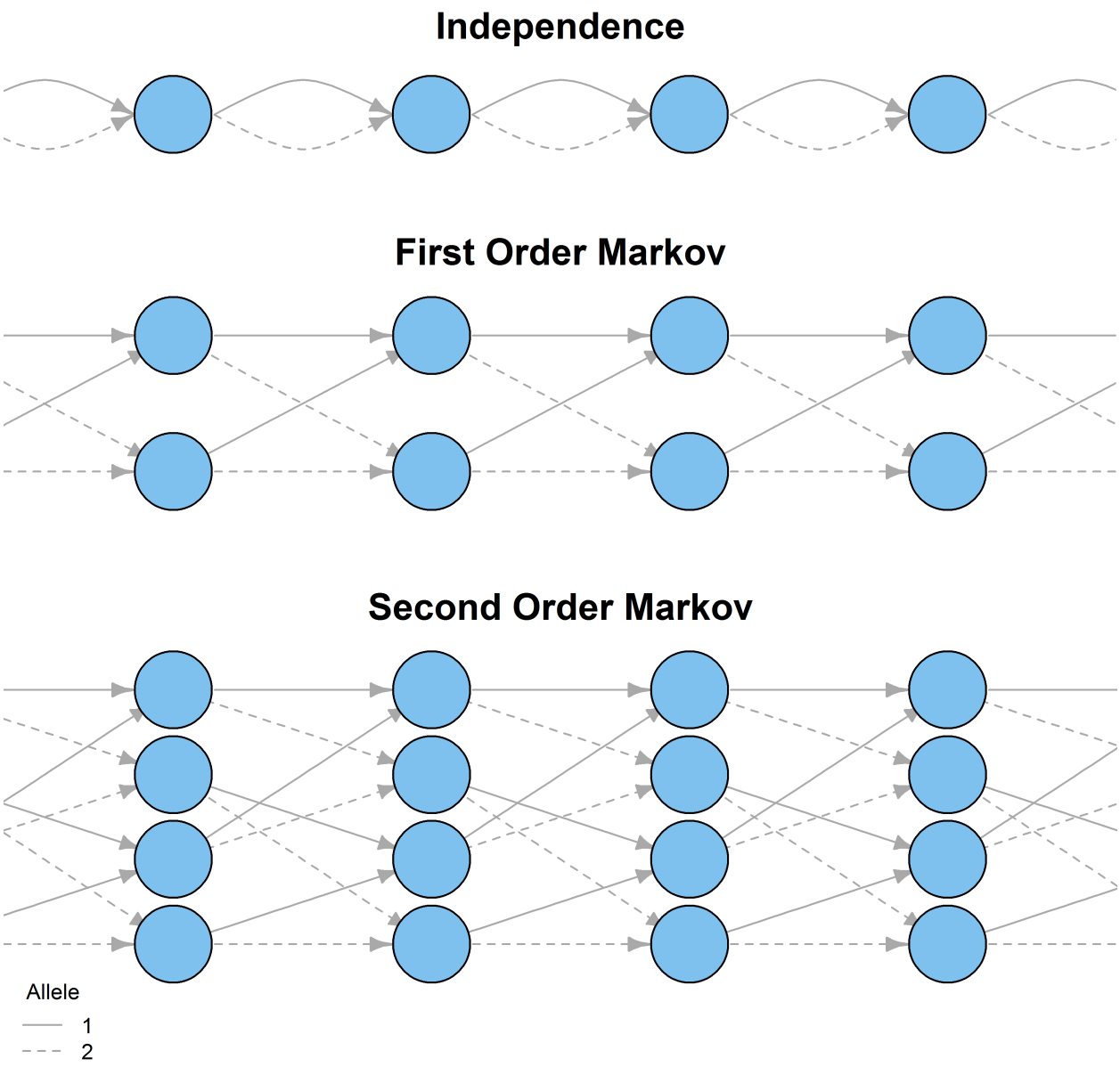

Figure 2.2: *Example DAGs representing the independence model along with first and second order Markov models for biallelic markers. Solid lines represent edges for allele 1, and dashed lines represent edges for allele 2.*

Chapter 3

MODEL ESTIMATION AND SELECTION

Given an independent sample of N phased haplotypes $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N$, our goal in this chapter is twofold. First, we aim to derive the likelihood of a directed acyclic graph (DAG) model $G = (V, E, w)$ and compute the weight function w which maximizes this likelihood. Our second objective is to select a DAG model which allows for a good trade-off between bias and variance. A large model with many vertices and edges will account for much of the linkage disequilibrium (LD) between markers, keeping bias low, but a smaller, more parsimonious model will allow for better estimation of edge weights, keeping variance low.

3.1 Likelihood estimation

As shown in the previous chapter, a single haplotype constitutes a unique path through a DAG, so its frequency can be estimated by the product of the edge weights along that path. Therefore, we can easily express the likelihood of the edge weight parameters w given an independent random sample of haplotypes from a population as follows:

$$\mathcal{L}(w|\mathbf{x}_1, \ldots, \mathbf{x}_N, V, E) = \prod_{i=1}^{N} \prod_{\ell=1}^{M} w(e_{\ell, j(\mathbf{x}_{i,1:\ell-1}), x_{i,\ell}}) = \prod_{e_{\ell,j,k} \in E} w(e_{\ell,j,k})^{n_{\ell,j,k}},$$

where $n_{\ell,j,k} = \sum_{i=1}^{N} [\mathbf{x}_{i,1:\ell-1} \in v_{\ell-1,j}, x_{i,\ell} = k]$ is the number of sampled haplotype paths which traverse the edge $e_{\ell,j,k}$. This provides us with a simple expression for the log-likelihood

$$l(w|\mathbf{x}_1, \ldots, \mathbf{x}_N, V, E) = \sum_{e_{\ell,j,k} \in E} n_{\ell,j,k} \log w(e_{\ell,j,k}), \tag{3.1}$$

and maximum likelihood estimates (MLEs)

$$\hat{w}(e_{\ell,j,k}) = \frac{n_{\ell,j,k}}{n_{\ell-1,j}}, \tag{3.2}$$

where $n_{\ell-1,j} = \sum_{i=1}^{N}[\mathbf{x}_{i,1:\ell-1} \in v_{\ell-1,j}]$ is the number of haplotype paths which pass through the vertex $v_{\ell-1,j}$.

Recall from Equation 2.3 that the weight of a vertex can be expressed as the sum of the products of the edge weights along each path entering the vertex. Also observe that a vertex at level ℓ can be expressed as the union of mutually exclusive sets derived from the vertices at the preceding level by taking

$$v_{\ell,j} = \bigcup_{j'} \{x_{1:\ell} : x_{1:\ell-1} \in v_{\ell-1,j'}, (v_{\ell-1,j'}, v_{\ell,j}, x_\ell) \in E\}.$$

Then we have

$$\begin{aligned}
W(v_{\ell,j}) &= \sum_{\mathbf{x}_{1:\ell} \in v_{\ell,j}} \prod_{t=1}^{\ell} w(e_{t,j(\mathbf{x}_{1:t-1}),x_t}) \\
&= \sum_{\mathbf{x}_{1:\ell} \in v_{\ell,j}} \left[\prod_{t=1}^{\ell-1} w(e_{t,j(\mathbf{x}_{1:t-1}),x_t})\right] w(e_{\ell,j(\mathbf{x}_{1:\ell-1}),x_\ell}) \\
&= \sum_{j'} \sum_{\mathbf{x}_{1:\ell-1} \in v_{\ell-1,j'}} \left[\prod_{t=1}^{\ell-1} w(e_{t,j(\mathbf{x}_{1:t-1}),x_t})\right] \sum_{x_\ell} w(e_{\ell,j',x_\ell}) \mathbb{1}\{d(\ell, j', x_\ell) = j\} \\
&= \sum_{j'} W(v_{\ell-1,j'}) \sum_{x_\ell} w(e_{\ell,j',x_\ell}) \mathbb{1}\{d(\ell, j', x_\ell) = j\},
\end{aligned}$$

where $j(\mathbf{x}_{1:t-1})$ is the index of the vertex at level $t-1$ containing the haplotype $\mathbf{x}_{1:t-1}$, and $\mathbb{1}\{d(\ell, j', x_\ell) = j\}\}$ is the indicator function for whether the destination vertex of the edge e_{ℓ,j',x_ℓ} is $v_{\ell,j}$ using the destination function Equation 2.1. That is, the weight of a vertex can also be expressed as the weighted average of the weights of the edges entering the vertex, weighted by the vertex weights of the source vertices for those edges. Now if we let the weight of vertex $v_{0,1}$ equal 1 and for the purposes of induction assume that the MLEs for $W(v_{\ell-1,j'})$ are

$$\hat{W}(v_{\ell-1,j'}) = \frac{n_{\ell-1,j'}}{N},$$

then we can show that the MLE for $W(v_{\ell,j})$ is

$$\begin{aligned}\hat{W}(v_{\ell,j}) &= \sum_{j'} \hat{W}(v_{\ell-1,j'}) \sum_{x_\ell} \hat{w}(e_{\ell,j',x_\ell}) \mathbb{1}\{d(\ell, j', x_\ell) = j\} \\ &= \sum_{j'} \frac{n_{\ell-1,j'}}{N} \sum_{x_\ell} \frac{n_{\ell,j',x_\ell}}{n_{\ell-1,j'}} \mathbb{1}\{d(\ell, j', x_\ell) = j\} \\ &= \frac{1}{N} \sum_{j'} \sum_{x_\ell} n_{\ell,j',x_\ell} \mathbb{1}\{d(\ell, j', x_\ell) = j\} = \frac{n_{\ell,j}}{N},\end{aligned}$$

since the sum $\sum_{j'} \sum_{x_\ell} n_{\ell,j',x_\ell} \mathbb{1}\{d(\ell, j', x_\ell) = j\}$ is exactly the number of sampled haplotype paths entering the vertex $v_{\ell,j}$.

These maximum likelihood estimates should be expected. Vertex weights represent the probability of all haplotypes included within the vertex, so it is natural that they would be estimated by the proportion of sampled paths that enter the vertex. Likewise, edge weights represent the conditional probability of a haplotype path leaving a vertex through that edge, so it is natural to estimate them by the proportion of sampled paths traversing that edge out of the sampled paths entering its source vertex. This result underscores the need for a parsimonious DAG for low variance estimation. The vertices at any level comprise a partition of the haplotypes of that length, and if this partition is too sparse, the denominators in the edge weight MLEs would be low, causing high variance estimates.

3.2 Model selection

A good DAG model for haplotype frequencies should be structurally rich enough to account for much of the LD in the population yet parsimonious enough to allow for precise estimation of its parameters. Finding the appropriate balance between a large low-bias DAG and a small low-variance DAG will be our primary concern in model selection.

If we start with the largest saturated model represented by the complete haplotype tree, model selection is equivalent to asking which vertices in the tree should be merged at each level. When two vertices merge, the resulting vertex represents the union of the two sets of haplotypes corresponding to the original vertices. Recall that the two vertices $v_{\ell,j}$ and $v_{\ell,j'}$

at level ℓ of the DAG induce the following context-sensitive independencies upon the model:

$$\mathbf{X}_{1:\ell} \perp\!\!\!\perp \mathbf{X}_{\ell+1:M} | \mathbf{x}_{1:\ell} \in v_{\ell,j} \quad \text{and} \quad \mathbf{X}_{1:\ell} \perp\!\!\!\perp \mathbf{X}_{\ell+1:M} | \mathbf{x}_{1:\ell} \in v_{\ell,j'}.$$

By merging these vertices, these independencies also merge to the broader

$$\mathbf{X}_{1:\ell} \perp\!\!\!\perp \mathbf{X}_{\ell+1:M} | \mathbf{x}_{1:\ell} \in v_{\ell,j} \cup v_{\ell,j'}. \tag{3.3}$$

An important consequence of merging two vertices is that all further downstream vertices in their subtrees also merge. More precisely, if the vertices $v_{\ell,j}$ and $v_{\ell,j'}$ merge, then for all alleles k, the next level's vertices $v_{\ell+1,d(\ell+1,j,k)}$ and $v_{\ell+1,d(\ell+1,j',k)}$ must also merge, where $d(\ell+1,j,k)$ and $d(\ell+1,j',k)$ are the indices of the destination vertices for the edges corresponding to allele k coming from $v_{\ell,j}$ and $v_{\ell,j'}$, respectively, as defined in Equation 2.1. That is, the entire subtrees rooted at $v_{\ell,j}$ and $v_{\ell,j'}$ merge into a single subtree.

To better understand why these downstream merges are necessary, consider two haplotypes $\mathbf{x}_{1:\ell} \in v_{\ell,j}$ and $\mathbf{x}'_{1:\ell} \in v_{\ell,j'}$ along with an allele k at marker $\ell+1$. This means that by appending the allele k to those haplotypes, we get that $(\mathbf{x}_{1:\ell}, k) \in v_{\ell+1,d(\ell+1,j,k)}$ and $(\mathbf{x}'_{1:\ell}, k) \in v_{\ell+1,d(\ell+1,j',k)}$. Thus, at subsequent markers we have the probability identity

$$\begin{aligned}
\Pr(\mathbf{X}_{\ell+2:M} = \mathbf{x}_{\ell+2:M} | \mathbf{X}_{1:\ell+1} = (\mathbf{x}_{1:\ell}, k)) &= \frac{\Pr(\mathbf{X}_{\ell+1:M} = (k, \mathbf{x}_{\ell+2:M}) | \mathbf{X}_{1:\ell} = \mathbf{x}_{1:\ell})}{\Pr(X_{\ell+1} = k | \mathbf{X}_{1:\ell} = \mathbf{x}_{1:\ell})} \\
&= \frac{\Pr(\mathbf{X}_{\ell+1:M} = (k, \mathbf{x}_{\ell+2:M}) | \mathbf{X}_{1:\ell} = \mathbf{x}_{1:\ell})}{\sum_{\mathbf{x}^*_{\ell+2:M}} \Pr(\mathbf{X}_{\ell+1:M} = (k, \mathbf{x}^*_{\ell+2:M}) | \mathbf{X}_{1:\ell} = \mathbf{x}_{1:\ell})} \\
&= \frac{\Pr(\mathbf{X}_{\ell+1:M} = (k, \mathbf{x}_{\ell+2:M}) | \mathbf{X}_{1:\ell} = \mathbf{x}'_{1:\ell})}{\sum_{\mathbf{x}^*_{\ell+2:M}} \Pr(\mathbf{X}_{\ell+1:M} = (k, \mathbf{x}^*_{\ell+2:M}) | \mathbf{X}_{1:\ell} = \mathbf{x}'_{1:\ell})}, \\
&\qquad \text{since } \mathbf{X}_{1:\ell} \perp\!\!\!\perp \mathbf{X}_{\ell+1:M} | \mathbf{x}_{1:\ell} \in v_{\ell,j} \cup v_{\ell,j'}, \\
&= \Pr(\mathbf{X}_{\ell+2:M} = \mathbf{x}_{\ell+2:M} | \mathbf{X}_{1:\ell+1} = (\mathbf{x}'_{1:\ell}, k)).
\end{aligned}$$

Therefore, merging $v_{\ell,j}$ and $v_{\ell,j'}$ induces the downstream context-sensitive independencies

$$\mathbf{X}_{1:\ell+1} \perp\!\!\!\perp \mathbf{X}_{\ell+2:M} | (\mathbf{x}_{1:\ell}, k) \in v_{\ell+1,d(\ell+1,j,k)} \cup v_{\ell+1,d(\ell+1,j',k)}, \tag{3.4}$$

which recurses further downstream until the end of the DAG. Figures 3.1a and 3.1b show an example of merging vertices in a tree DAG. Merging $v_{2,1}$ and $v_{2,2}$ would induce the merges

of $v_{3,1}$ and $v_{3,3}$ since they are children by the edge corresponding to allele 1 and the merges of $v_{3,2}$ and $v_{3,4}$ since they are children by the edge corresponding to allele 2. These merges then induce further merges among the four corresponding pairs in the next level, and so on until the end of the DAG is reached.

When a saturated DAG model is created from a sample of haplotypes, the resulting tree will not be anywhere close to complete. In the case of biallelic SNP markers, only a small proportion of the 2^M possible haplotypes are expected to be observed in a sample or even exist in the population. Because of this, most paths through the DAG will not exist, leading to many vertices with only a single outgoing edge. When these vertices are merged with vertices having multiple outgoing edges, new possible haplotypes enter the DAG model. In fact, these sorts of merges are the only way in which new haplotypes can enter the model and obtain nonzero frequency estimates. In Figure 3.1a, vertex $v_{2,3}$ does not have an outgoing edge corresponding to allele 2, so no haplotypes starting with the allele sequence $\{212\ldots\}$ exist within this model. However, if vertices $v_{2,3}$ and $v_{2,4}$ were to merge, the resulting vertex would have two outgoing edges, and haplotype paths beginning with the sequence $\{212\ldots\}$ would exist.

Our method for selecting a haplotype frequency model involves minimizing an objective function which includes a likelihood-based loss component and a regularization term penalizing the complexity of the DAG model. Specifically, we define the objective function for a model G given the sample of haplotypes by

$$\mathcal{J}_\lambda(G|\mathbf{x}_1,\ldots,\mathbf{x}_N) = -l(\hat{w}|\mathbf{x}_1,\ldots,\mathbf{x}_N,V,E) + \lambda|\{v_{\ell,j}\in V|\ell<M\}|, \tag{3.5}$$

where the l_0 regularization term penalizes the negative log-likelihood loss component by an amount proportional to the number of vertices in the DAG (excluding the terminal vertex $v_{M,1}$) with proportionality constant λ. This objective function gives us a basis by which to determine whether simplifying a DAG model by merging two vertices at the same level provides a sufficient trade-off in terms of the resulting decrease in likelihood.

By weighing the loss directly against the number of parameters, l_0 regularization gener-

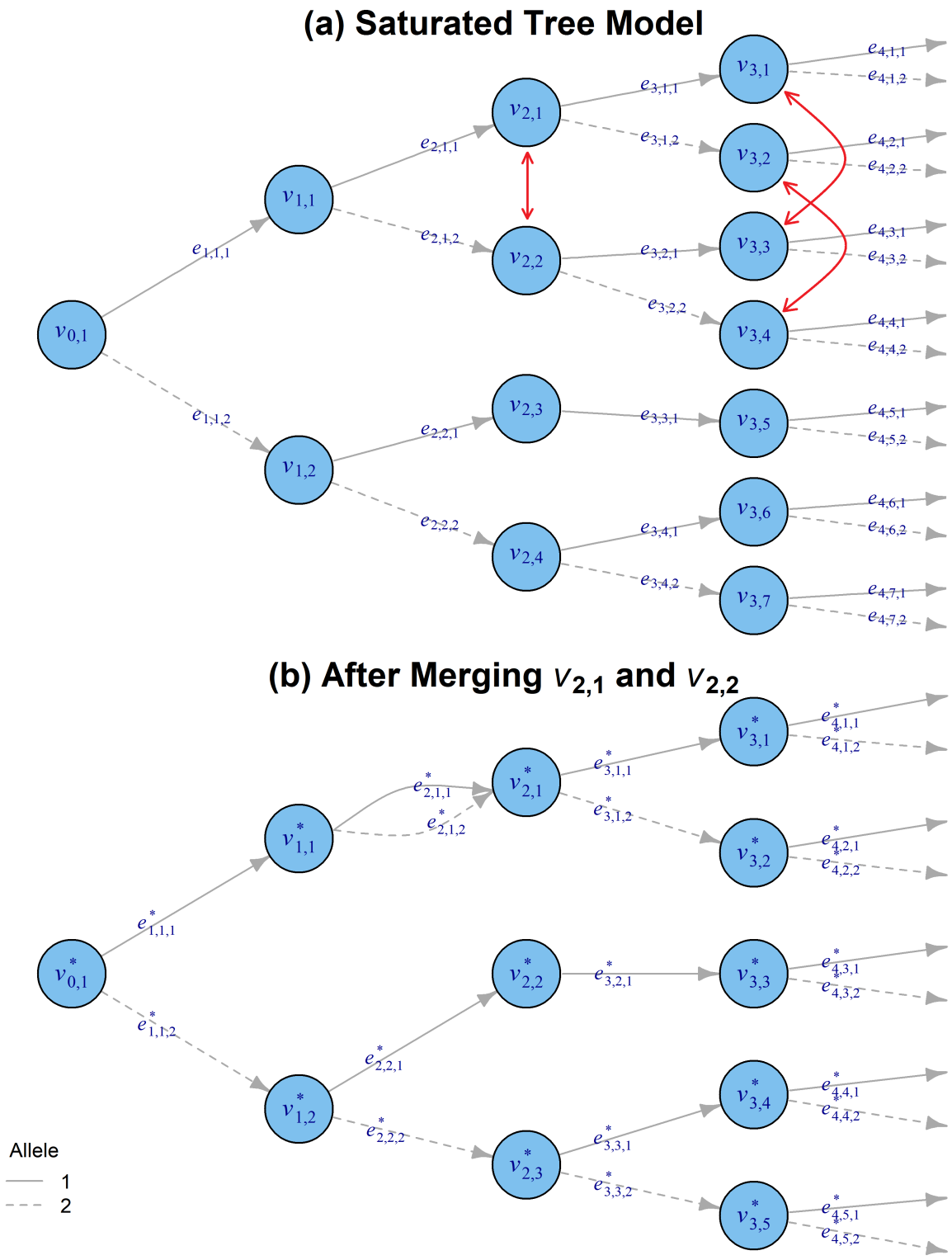

Figure 3.1: *Example of merging two vertices within a saturated tree DAG model. Solid lines represent edges for allele 1, and dashed lines represent edges for allele 2.*

ally finds a model with enough parameters to allow for a low-bias fit while still parsimonious enough to provide low-variance estimates. However, the l_0-norm term in the objective function is non-differentiable, so neither analytic nor numerical minimization is possible, and the optimization amounts to a best-subset selection of the parameters, which does not often scale well to a large number of parameters (Tibshirani et al., 2015). The most common solution to this problem is the convexification of the objective function through the use of an l_1-norm instead of the l_0-norm, a method known as the lasso (Tibshirani, 1996). This allows for fast numerical optimization with a large number of parameters while still encouraging sparsity and parsimony through the geometry of the l_1-norm. However, the standard lasso would not be appropriate for the selection of DAG models since we wish to encourage sparsity not by shrinking parameters to zero, which is the equivalent of deleting nodes, but by shrinking the difference between parameters to zero, which is the equivalent of merging nodes. This problem has much in common with the fused lasso, which regularizes the loss function with a penalty on the l_1-norm of differences between parameters (Tibshirani et al., 2005). An even further generalization of the lasso minimizes the objective function

$$\mathcal{J}_\lambda(\beta) = f(\beta) + \lambda||D\beta||_1,$$

allowing for a broad class of sparsity structures defined by the penalty matrix D (Tibshirani et al., 2015; Tibshirani & Taylor, 2011). Though because of the recursive nature of vertex merging in DAGs, even this generalized lasso can not capture the appropriate penalty structure. Vertices merging in level ℓ impose merges in level $\ell + 1$, but not vice-versa. The linear penalty matrix D is unable to encode these sorts of conditional operations. Convexification of the DAG model selection problem could greatly improve both performance and computational speed, but at this time it remains a topic for further research.

Fortunately, the ordered structure of the DAG model allows for an approximation of best-subset selection that is not intractably combinatorial in complexity. For a particular value of the regularization parameter λ, we attempt to find a model which minimizes the objective function J_λ from Equation 3.5 using a greedy algorithm which traverses each level

of the DAG from 1 to M and chooses the best merges among those vertices. We begin with the largest DAG model, a tree in which each uniquely observed haplotype has it own distinct path. At level ℓ, merges involving all pairs of vertices $v_{\ell,j}$ and $v_{\ell,j'}$ are compared by the differences

$$\Delta_{\ell,j,j'} = \mathcal{J}_\lambda(G|\mathbf{x}_1,\ldots,\mathbf{x}_N) - \mathcal{J}_\lambda(G_{\ell,j,j'}|\mathbf{x}_1,\ldots,\mathbf{x}_N), \tag{3.6}$$

where $G_{\ell,j,j'}$ is the DAG model obtained by merging the subtrees rooted at $v_{\ell,j}$ and $v_{\ell,j'}$. That is, when the subtrees rooted at $v_{\ell,j}$ and $v_{\ell,j'}$ have similar edge weights, the difference $\Delta_{\ell,j,j'}$ will tend to be larger since the drop in penalized parameters will outweigh the small change in fit as measured by the likelihood. Positive values of $\Delta_{\ell,j,j'}$ indicate a good merge that will reduce the objective function while negative values indicate a poor merge that will increase the objective function. The greedy algorithm computes the differences $\Delta_{\ell,j,j'}$ for each pair of vertices in a level and merges the pair with the largest nonnegative value. Then differences are computed for pairs with this new merged vertex, and again the pair with the largest nonnegative value is merged. This is repeated until there are no longer any good merges, until all of the differences are negative or only a single vertex remains. Then the algorithm moves to the next level and repeats the entire process.

In practice, it would be computationally prohibitive to compute the objective function differences $\Delta_{\ell,j,j'}$ obtained from merging each pair of vertices by actually constructing each merged DAG $G_{\ell,j,j'}$. Instead, they are computed through the single-level differences

$$\delta_{\ell,j,j'} = \sum_k \left[-n_{\ell,j,k} \log\left(\frac{n_{\ell,j,k}}{n_{\ell-1,j}}\right) - n_{\ell,j',k} \log\left(\frac{n_{\ell,j',k}}{n_{\ell-1,j'}}\right) + (n_{\ell,j,k} + n_{\ell,j',k}) \log\left(\frac{n_{\ell,j,k} + n_{\ell,j',k}}{n_{\ell-1,j} + n_{\ell-1,j'}}\right)\right] + \lambda,$$

which represent the contributions to the total difference $\Delta_{\ell,j,j'}$ obtained by merging each single pair of vertices in the subtrees rooted at $v_{\ell,j}$ and $v_{\ell,j'}$. Using these values, Equation 3.6 can be expressed by the recurrence relation

$$\Delta_{\ell,j,j'} = \delta_{\ell,j,j'} + \sum_k \Delta_{\ell+1,d(\ell+1,j,k),d(\ell+1,j',k)}. \tag{3.7}$$

That is, the change in objective function obtained by merging a pair of vertices is the sum of their single-level difference and the differences in objective functions obtained by merging their downstream vertices.

It is possible that a difference $\Delta_{\ell,j,j'}$ could be large in spite of the edge weights attached to $v_{\ell,j}$ and $v_{\ell,j'}$ being quite dissimilar if the edge weights of downstream subtrees are similar instead. In this case, it would not always be desirable to merge the vertices $v_{\ell,j}$ and $v_{\ell,j'}$ when the downstream vertices could be merged later. A modification to the greedy algorithm allows us to account for this possibility. While each $\Delta_{\ell,j,j'}$ is being computed recursively using Equation 3.7, if the running total becomes negative at any point during the depth-first traversal of the DAG, the computation is halted and a negative value is returned. That is, if any of the partial sums of single level differences $\delta_{\ell,j_0,j'_0} + \delta_{\ell+1,j_1,j'_1} + \delta_{\ell+2,j_2,j'_2} + \ldots + \delta_{\ell+2,j_m,j'_m}$ down merging paths is negative, then the entire merging of subtrees is disallowed. This includes the specific case in which $\Delta_{\ell,j,j'}$ is positive but $\delta_{\ell,j,j'}$ is negative, so merging the downstream pairs of vertices $v_{\ell+1,d(\ell+1,j,k)}$ and $v_{\ell+1,d(\ell+1,j',k)}$ for each allele k results in a lower objective value than simply merging the parent vertices $v_{\ell,j}$ and $v_{\ell,j'}$. This modification also speeds up the algorithm by allowing the recursive computations to be terminated quickly in the case of poor merges.

It may appear that computing the objective function differences $\Delta_{\ell,j,j'}$ using the recurrence relation Equation 3.7 would have exponential running time in the number of markers M. However, in the case of biallelic SNP markers, for instance, only a small fraction of the 2^M possible haplotypes will be observed in a sample. Hence, the initial saturated DAG tree will be very sparse. This means that most of the branching pathways quickly run out of data, leaving only a single path for each haplotype in the sample. Therefore, most of the downstream vertices do not have a corresponding vertex with which to merge, so most of the recursive differences $\Delta_{\ell+1,d(\ell+1,j,k),d(\ell+1,j',k)}$ in Equation 3.7 will equal zero. Overall, the greedy algorithm can run in $O(MN^2 \log N)$ time under the worst case when there are no good merges, linear in the number of markers M, polynomial in the number of samples N.

One computational issue does arise in storing the initial saturated DAG tree in memory.

Algorithm 1: *Greedy algorithm for selecting a haplotype frequency model.*

for $\ell \leftarrow 1$ **to** M **do**
  repeat
    foreach *pair of vertices* $v_{\ell,j}, v_{\ell,j'}$ **do**
      $\Delta_{\ell,j,j'} \leftarrow \delta_{\ell,j,j'} + \sum_k \Delta_{\ell+1,d(\ell+1,j,k),d(\ell+1,j',k)}$, returning -1 if the value is negative at any point in the recursion
    end
    merge $v_{\ell,j}, v_{\ell,j'}$ corresponding to maximum $\Delta_{\ell,j,j'}$ if $\Delta_{\ell,j,j'} \geq 0$ for some pair of vertices
  until *no merge occurs*;
end

Since this tree contains a single pathway for each haplotype in the sample, it requires space on the order of $O(MN)$, which is prohibitive when working with genome-wide population-level data that may contain millions of markers over thousands of samples. To alleviate these memory concerns, the algorithm operates on only a relatively small window of markers at any time, a window that dynamically changes in size along the genome. This windowing is made possible by two properties of the DAG. First, once the algorithm has reached level ℓ, the Markovian properties underlying the model ensure that none of the previous levels of the DAG are relevant or mutable, so they can be saved and flushed from working memory. Second, as previously mentioned, the recursion performed in computing the objective function differences $\Delta_{\ell,j,j'}$ using Equation 3.7 only goes so deep until it runs out of data. This means that none of the markers in the DAG beyond this point need to be read into memory until they are required. Therefore, as the algorithm iterates along the markers, only a windowed portion of the DAG is held in memory, and although this window fluctuates in size, it does not grow with the number of total markers M.

It is important to note that merging vertices always reduces the l_0 regularization term in Equation 3.5 without ever reducing the negative log-likelihood loss term, so when λ is

near zero, differences in the objective functions $\Delta_{\ell,j,j'}$ between the larger model G and the smaller model $G_{\ell,j,j'}$ will be tend to be negative. Therefore, few merges will occur in the algorithm, and the resulting model will have many parameters and retain substantial LD structure. Conversely, as λ increases, these same differences will be tend to be positive, so many merges will occur, and the resulting model will be more parsimonious and retain less LD structure.

3.3 Selection of regularization parameter

To select the best value for the regularization parameter λ, cross-validation can be used. Initially, at a particular value of λ, all of the haplotypes in the sample are used to determine the structure of the DAG model using the algorithm above. Then the sample of haplotypes is split into a larger training set and a smaller validation set. The edge weight parameters w for this DAG model are estimated by Equation 3.2 using the training data. This model is then tested by computing its negative log-likelihood loss given the haplotypes in the validation data. A more conventional cross-validation would have both constructed and estimated the DAG model on the training data, but there were two important reasons to modify this method. First, since the validation haplotypes were not used in constructing the model, it is quite possible that they would not have valid paths through the DAG since they might encounter vertices from which further marker alleles were not observed in the training haplotypes. Second, constructing the DAG is far more computationally expensive than estimating the DAG parameters, so in order to perform a k-fold cross-validation, it is much easier to perform this task only once.

Although using the entire sample of haplotypes to build the model ensures that all of the validation haplotypes will have paths through the DAG, it is quite possible that these paths will have edges with zero weight since they might encounter vertices from which further marker alleles were not observed in the training haplotypes. In order to avoid infinite validation loss in these cases, the counts of each edge in the DAG are seeded with a single observation prior to training. This guarantees that all edge weights will be positive and the

validation loss will be finite.

Interestingly, adding a single observation to each edge gives the resulting log-likelihood value on the validation haplotypes a specific interpretation in a Bayesian context. If the weights of the edges leaving each vertex are assigned independent uniform Dirichlet distributions over their simplexes, then computed validation loss is equivalent to the posterior predictive probability of the validation haplotypes given the training haplotypes and the DAG structure. To see this, we must introduce some new notation. For any haplotype $\mathbf{X} = (X_1, X_2, \ldots, X_M)$, let

$$Z_{\ell,j,k} = \begin{cases} 1 & \text{if } \mathbf{X}_{1:\ell-1} \in v_{\ell-1,j} \quad \text{and} \quad X_\ell = k \\ 0 & \text{otherwise} \end{cases},$$

be the indicator variable for whether the haplotype traverses the edge $e_{\ell,j,k}$. If each marker locus has K_ℓ alleles labeled $1, 2, \ldots, K_\ell$, then we also define the vectors

$$\begin{aligned} \mathbf{w}_{\ell,j} &= [w(e_{\ell,j,1}), w(e_{\ell,j,2}), \ldots, w(e_{\ell,j,K_\ell})]^T, \\ \mathbf{n}_{\ell,j} &= [n_{\ell,j,1}, n_{\ell,j,2}, \ldots, n_{\ell,j,K_\ell}]^T, \\ \mathbf{Z}_{\ell,j} &= [Z_{\ell,j,1}, Z_{\ell,j,2}, \ldots, Z_{\ell,j,K_\ell}]^T, \text{ and} \\ \boldsymbol{\alpha}_{\ell,j} &= [\alpha_{\ell,j,1}, \alpha_{\ell,j,2}, \ldots, \alpha_{\ell,j,K_\ell}]^T \end{aligned}$$

representing the edge weights, observed traversal counts, traversal indicators, and prior parameters, respectively, for each edge leaving the vertex $v_{\ell-1,j}$. Now if we assume prior distributions for the edge weights of the DAG conditional on the structure G

$$\mathbf{w}_{\ell,j} | G \overset{iid}{\sim} \text{Dirichlet}(\boldsymbol{\alpha}_{\ell,j}),$$

then their posterior probabilities given the observed training haplotypes are

$$\begin{aligned} \pi(w|\mathbf{x}_1, \ldots, \mathbf{x}_N, V, E) &\propto \mathcal{L}(w|\mathbf{x}_1, \ldots, \mathbf{x}_N, V, E) \cdot \pi(w|V, E) \\ &\propto \left(\prod_{e_{\ell,j,k} \in E} w(e_{\ell,j,k})^{n_{\ell,j,k}} \right) \cdot \left(\prod_{e_{\ell,j,k} \in E} w(e_{\ell,j,k})^{\alpha_{\ell,j,k}-1} \right) \\ &\propto \prod_{e_{\ell,j,k} \in E} w(e_{\ell,j,k})^{n_{\ell,j,k}+\alpha_{\ell,j,k}-1}, \end{aligned}$$

so the posterior distributions for the edge weights are

$$\mathbf{w}_{\ell,j}|\mathbf{x}_1,\ldots,\mathbf{x}_N,G \overset{iid}{\sim} \text{Dirichlet}(\mathbf{n}_{\ell,j}+\boldsymbol{\alpha}_{\ell,j}).$$

If we consider a new validation haplotype $\mathbf{X}^* = (X_1^*, X_2^*, \ldots, X_M^*)$, then its posterior predictive probability given the training haplotypes is

$$\begin{aligned}
p(\mathbf{X}^*=\mathbf{x}^*|\mathbf{x}_1,\ldots,\mathbf{x}_N,V,E) &= \int \mathcal{L}(w|\mathbf{x}^*,V,E)\cdot\pi(w|\mathbf{x}_1,\ldots,\mathbf{x}_N,V,E)dw \\
&= \int \left(\prod_{e_{\ell,j,k}\in E} w(e_{\ell,j,k})^{z^*_{\ell,j,k}}\right)\cdot\left(\prod_\ell\prod_j \frac{1}{B(\mathbf{n}_{\ell,j}+\boldsymbol{\alpha}_{\ell,j})}\right) \\
&\qquad \cdot\left(\prod_{e_{\ell,j,k}\in E} w(e_{\ell,j,k})^{n_{\ell,j,k}+\alpha_{\ell,j,k}-1}\right)dw \\
&= \left(\prod_\ell\prod_j \frac{1}{B(\mathbf{n}_{\ell,j}+\boldsymbol{\alpha}_{\ell,j})}\right)\int\left(\prod_{e_{\ell,j,k}\in E} w(e_{\ell,j,k})^{z^*_{\ell,j,k}+n_{\ell,j,k}+\alpha_{\ell,j,k}-1}\right)dw \\
&= \prod_\ell\prod_j \frac{B(\mathbf{z}^*_{\ell,j}+\mathbf{n}_{\ell,j}+\boldsymbol{\alpha}_{\ell,j})}{B(\mathbf{n}_{\ell,j}+\boldsymbol{\alpha}_{\ell,j})},
\end{aligned}$$

where B is the Euler beta function. Therefore, under the uniform Dirichlet priors where all $\alpha_{\ell,j,k}=1$, we have

$$\begin{aligned}
\Pr(X^*_\ell = k|\mathbf{x}^*_{1:\ell-1}\in v_{\ell-1,j},\mathbf{x}_1,\ldots,\mathbf{x}_N,V,E) &= \frac{B(\mathbf{z}^*_{\ell,j}+\mathbf{n}_{\ell,j}+\boldsymbol{\alpha}_{\ell,j})}{B(\mathbf{n}_{\ell,j}+\boldsymbol{\alpha}_{\ell,j})} \\
&= \frac{B(\mathbf{n}_{\ell,j}+\boldsymbol{\alpha}_{\ell,j})}{B(\mathbf{n}_{\ell,j}+\boldsymbol{\alpha}_{\ell,j})}\cdot\frac{n_{\ell,j,k}+\alpha_{\ell,j,k}}{\sum'_k n_{\ell,j,k'}+\alpha_{\ell,j,k'}} \\
&= \frac{n_{\ell,j,k}+1}{n_{\ell-1,j}+K_\ell},
\end{aligned}$$

since when $x_\ell = k$, $\mathbf{z}^*_{\ell,j}$ is the standard basis vector with a 1 in the kth position. That is, the probability of the new validation haplotype traversing edge $e_{\ell,j,k}$ given the context $v_{\ell-1,j}$ and the observed training haplotypes is simply the MLE $\hat{w}(e_{\ell,j,k})$ computed from the training data after all edge counts were seeded with a single observation.

3.4 1000 Genomes model selection example

To test our method for selecting a haplotype frequency model, we applied it to SNP marker data derived from phase 3 of the 1000 Genomes Project. This project was established to create the largest resource on human genetic variation to date (The 1000 Genomes Project Consortium, 2015). Low coverage whole-genome sequence data were available for 2,504 diploid individuals (5,008 haploids) from 26 populations, 661 from African populations, 504 from East Asian populations, 503 from European populations, 489 from South Asian populations, and 347 from Admixed American populations. From the variant call format (VCF) files containing these sequences, markers were selected using the following criteria: 1) must be a SNP; 2) must have been typed for all 2504 individuals; 3) must have a proper SNP name beginning with "rs"; 4) must have passed the generic filter; 5) must have quality score of 100; 6) must have exactly two alleles; 7) must have minor allele frequency of at least 1%; and 8) must be listed in the Rutgers map file with the same SNP name and physical position (Matise et al., 2007). This selection process, which mirrors that of Wang et al. (2017), results in over 11 million high quality biallelic SNPs across the 22 autosomes. For our tests, we used the 902,606 of these SNP markers from chromosome 1.

First, DAG models was created from the 902,606 phased markers at various values of the regularization parameter λ ranging from 2^{-5} to 2^4 using the full set of 5,008 haplotypes. Then these haplotypes were randomly partitioned for 10-fold cross-validation. For each fold, we estimated the edge weight parameters of the DAG models using the training set. From each model, we computed the negative log-likelihood loss on the training and validation sets as well as the number of vertices and edges and averaged all of these values across the 10 folds. Recall that when computing the validation loss, we seeded the DAG with an initial observation on each edge, making it equivalent to the negative posterior predictive probability of the validation data given the training data under the Dirichlet prior with all $\alpha_{\ell,j,k} = 1$.

For the purpose of comparison, we also used the Beagle version 3.3.2 software to create

DAG models using the same set of haplotypes and markers and computed the same metrics (Browning, 2006; Browning & Browning, 2007). Version 3 was chosen because the newer version 4 does not support the output of the resulting DAG model. However, version 3 was unable to run all 902,606 markers due to memory restrictions, so it was run in 19 non-overlapping windows of 50,000 markers, and these shorter DAGs were pasted together end-to-end. Instead of a regularization parameter, Beagle contains a scale parameter which controls the number of vertices in the resulting DAG model. Lower values of this parameter result in more vertices, and higher values result in fewer vertices. Scale values ranging from 0.6 to 4.0 were chosen.

Finally, everything above was repeated on the same 902,606 markers in the reverse direction along the chromosome. We note that DAG models and the VOMCs they represent are directional in nature. The conditional probabilities represented by edge weights (Equation 2.2) and the conditional independencies represented by merged vertices (Equation 3.3) both rely upon specific definitions of the "first" and "last" markers of the chromosome. Therefore, we would not expect DAG models fit and selected in one direction to be equivalent to those fit and selected in the other direction. Although a similar directional comparison could be made for Beagle, this was skipped for the purpose of simplicity. This issue of the directionality of DAG models will be explored in greater detail in the next chapter.

Figures 3.2-3.5 give the results from this test. In all of these plots for the likelihood-based method, values obtained from models fit in the "forward" direction are colored red while those from models fit in the "backward" direction are colored blue. Values obtained from the Beagle model are colored green. It is immediately noticeable that the results are nearly identical regardless of which direction the markers were considered as the red and blue lines overlap almost completely to form a single purple line. We found it somewhat surprising that these global measures are so similar despite the models differing locally. Furthermore, this was not the case in our initial results using many fewer markers. In some way, these local differences must be averaging out over a large number of markers.

Figure 3.2a shows the validation loss as measured by the posterior predictive probability

given the training haplotypes and DAG structure under the Dirichlet prior at various values of the regularization parameter λ for our likelihood-based method and the scale parameter for Beagle. Both of these curves obtained a minimum over these ranges of parameters, the likelihood-based method when λ equals 0.5 and Beagle when its scale equals 1.8. Notably, the likelihood-based method minimizes its curve at a lower value than Beagle does, indicating a better fit on the validation haplotypes. Figure 3.2b shows the Akaike information criterion (AIC) at the same values of the regularization and the scale parameters. AIC is a statistic that estimates the relative quality of models by penalizing the fit of the models as measured using likelihood by their parsimony as measured using the number of parameters (Akaike, 1998). It is interesting that these curves look nearly identical to those in Figure 3.2a, with minima at exactly the same values. This may indicate a possible statistical mechanism by which posterior predictive probabilities are asymptotically equivalent to AIC in these models, but we have no argument for such a mechanism at this time.

Figures 3.3a and 3.3b show, respectively, the log-likelihood and number of vertices for DAG models obtained from various values of the regularization and scale parameters. As expected, increasing the regularization parameter λ reduces the quality of the model fit as measured by likelihood while also reducing the size of the model by encouraging more merges among the vertices. At their minimum values for validation loss, the size of the DAGs resulting from the two methods are quite similar, 33.6 vertices per marker for the likelihood-based method and 35.9 vertices per marker for Beagle. Most importantly, although these optimal models are nearly the same size, the one obtained by the likelihood-based method has a substantially higher likelihood. Essentially, our optimal DAG model succeeds at modeling haplotype frequencies more accurately than Beagle with the same level of parsimony. In order for Beagle to obtain a model which fits the data as well as the optimal likelihood-based DAG, it must use a scale parameter of 0.7. This results in a model with 102.8 vertices per marker, which is nearly three times larger than the optimal likelihood-based DAG.

Figure 3.4a shows the proportion of vertices that are the source of two edges in the DAG models at various values of the regularization and scale parameters. We noticed that due to

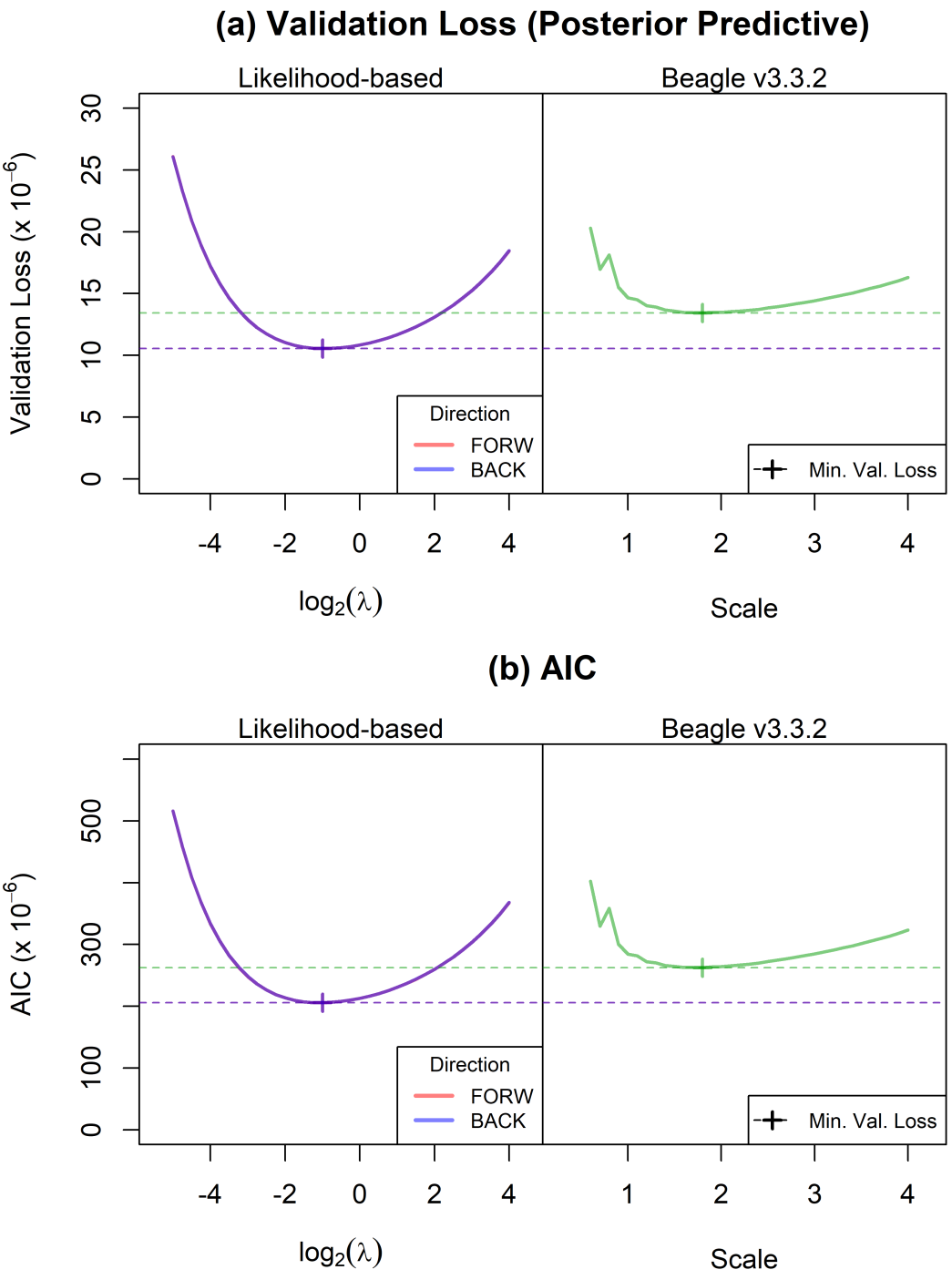

Figure 3.2: *DAG model selection results at various values of the regularization parameter λ for our likelihood-based method and the scale parameter for Beagle v3.3. Figure (a) shows validation loss, and (b) shows AIC. The likelihood-based DAGs were fit in both directions along the chromosome, but their red and blue lines almost completely overlap into purple lines. Beagle results are represented by a green line. The plus sign and dotted lines represent the values at which validation loss is minimized.*

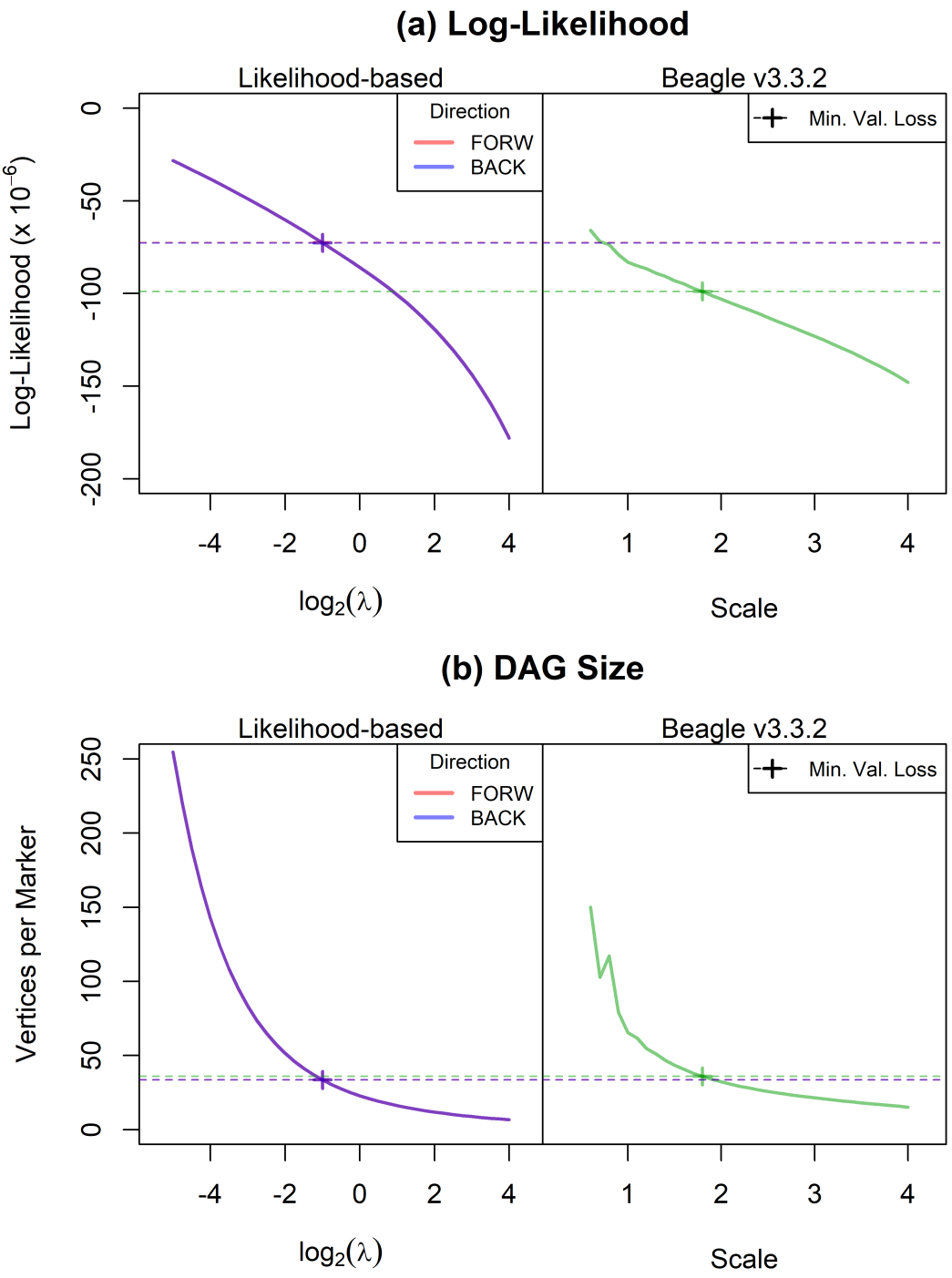

Figure 3.3: *DAG model selection results at various values of the regularization parameter λ for our likelihood-based method and the scale parameter for Beagle v3.3. Figure (a) shows log-likelihood, and (b) shows total number of vertices. The likelihood-based DAGs were fit in both directions along the chromosome, but their red and blue lines almost completely overlap into purple lines. Beagle results are represented by a green line. The plus sign and dotted lines represent the values at which validation loss is minimized.*

the high levels of LD in these dense markers, many vertices only had a single edge leaving them, indicating complete surety in the next allele given the contextual history. It is not surprising that increasing the regularization parameter λ increases the proportion of two-edge vertices since many of the single-edge vertices likely contain a smaller number of haplotypes and are prime candidates for merging. At their minimum values for validation loss, only 10.4% of vertices in the likelihood-based models and 8.2% in the Beagle model have two edges. That is, even though there are 33.6 vertices per marker on average (35.9 for Beagle), only 3.5 of them (2.9 for Beagle) actually provide a nontrivial conditional probability distribution. Furthermore, only these few vertices contribute to the log-likelihood calculations from Figure 3.3a. We note that there is a difference between these proportions for the likelihood-based and Beagle models, but we are unable to pinpoint the reasoning behind this. The merging criteria in the Beagle algorithm must slightly favor merges involving single-edge vertices.

Figure 3.4b shows the log-likelihood for the DAG models at various values of the number of vertices as imposed by the regularization and scale parameters. In essence, Figures 3.3a and 3.3b have been composed in such a way that each value for the regularization or scale parameter corresponds to both a log-likelihood and DAG size value, and these two values form the ordered pairs plotted in Figure 3.4b. This reformulation is notable because it illuminates how the likelihood-based models provide a better fit than the Beagle models at all sizes of the DAG. This gives us the best evidence yet that our method for DAG model selection improves upon that from the Beagle software.

Figure 3.5 shows how the size of the DAGs change along the chromosome for both the likelihood-based and Beagle models at their minimum values for validation loss. This is particularly interesting because although the likelihood-based and Beagle DAGs are similar in size overall, they are nearly inverted in how their sizes relate to chromosome position. The log-likelihood models have more vertices than average near the telomeres at the ends of the chromosome as well as a large spike in size near the centromere in the middle. In contrast, the Beagle model has fewer than average near the telomeres and a large drop near the centromere. Some research has suggested that LD is lower near the telomeres since this

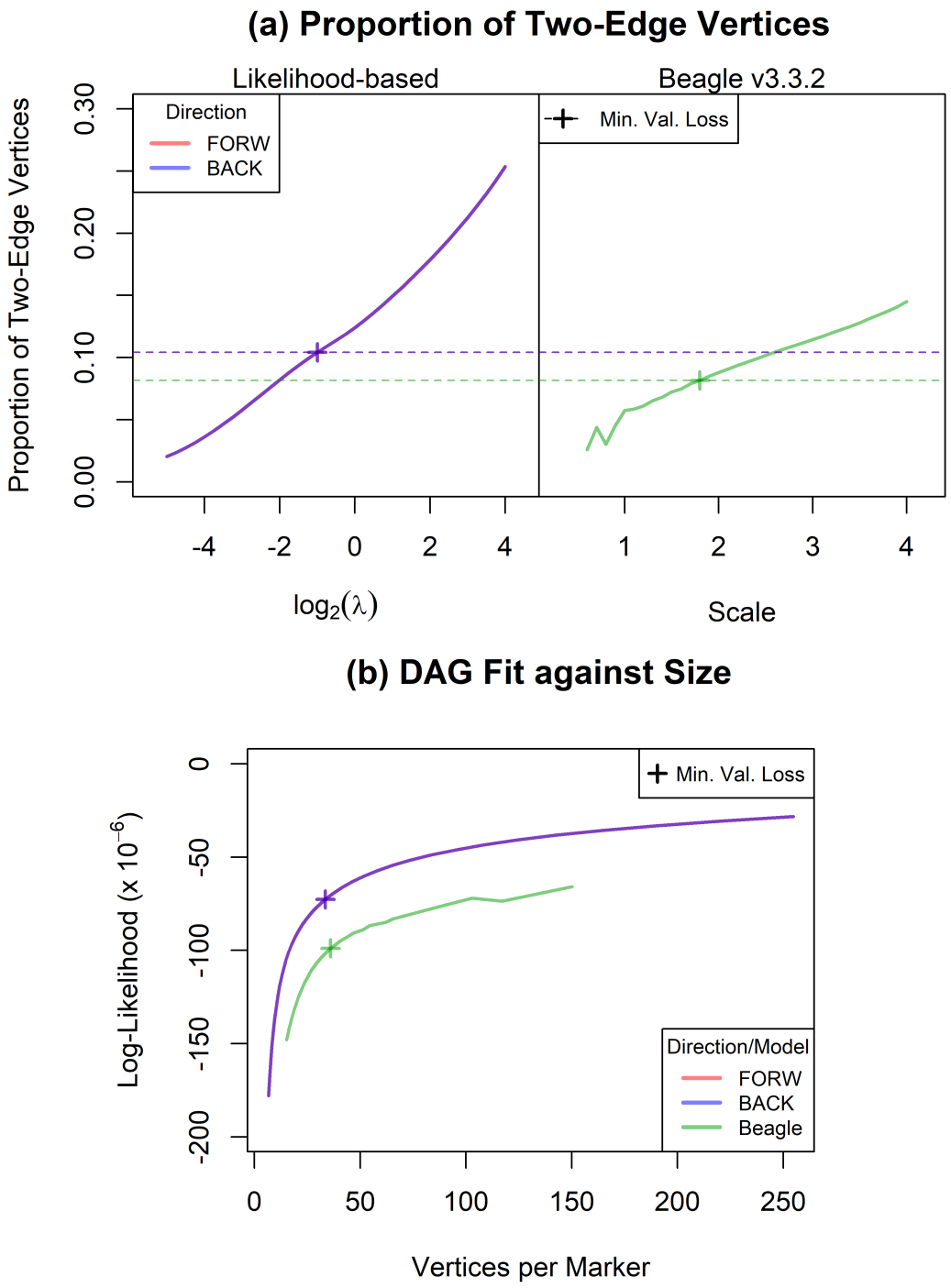

Figure 3.4: *DAG model selection results at various values of the regularization parameter λ for our likelihood-based method and the scale parameter for Beagle v3.3. Figure (a) shows proportion of vertices with two edges, and (b) shows log-likelihood against average number of vertices per marker. The likelihood-based DAGs were fit in both directions along the chromosome, but their red and blue lines almost completely overlap into purple lines. Beagle results are represented by a green line. The plus sign and dotted lines represent the values at which validation loss is minimized.*

is where recombination rates are highest during male meiosis and that LD is higher near the centromere where recombination rates tend to be low (Broman et al., 1998; Smith et al., 2005). This would indicate that we might expect fewer vertices near the telomeres as we see in the Beagle DAG yet more vertices near the centromere as we see in the likelihood based DAGs, so neither method seems entirely consistent here.

To investigate this further, we also broke down the quality of model fit by genetic position. Figure 3.6 shows how the log-likelihood and validation loss contributions by vertices in the DAGs change along the chromosome for both the likelihood-based and Beagle models at their minimum values for validation loss. We see that log-likelihood is lower for all models near the telomeres and the centromere, indicating greater difficulty in fitting those regions. However, the likelihood-based DAGs have better model fit than the Beagle DAG at all positions along the chromosome. It is unsurprising that model fit is better near the telomeres and centromere since in those regions the likelihood-based model had more vertices, but the likelihood-based models retain their better fit even in regions where they are more parsimonious than the Beagle DAG, such as from 50-200 cM excluding the centromere. Looking at the validation loss yields the same conclusions, so the apparent model fit superiority of the likelihood-based method is not simply an artifact of overfitting the data.

Both the likelihood-based and Beagle methods seem to have more difficulty fitting the regions near the telomeres and particularly near the centromere, but they appear to handle this difficulty in vastly different manners. The Beagle DAG has over 600 loci in the 10 cM region surrounding the centromere where it collapses to a single vertex, indicating linkage equilibrium between adjacent markers, an extremely unlikely assumption with markers as dense as these. Meanwhile, some of our testing has shown that the likelihood-based method can result in an overabundance of vertices in regions with very low LD, which might explain the greater number of vertices in these DAGs over these difficult regions. We believe that this happens because there tend to be more unique observed haplotypes in low LD regions, so the saturated tree model that initiates the merging algorithm starts much wider in these regions. In high LD regions where only a small number of different haplotypes are observed,

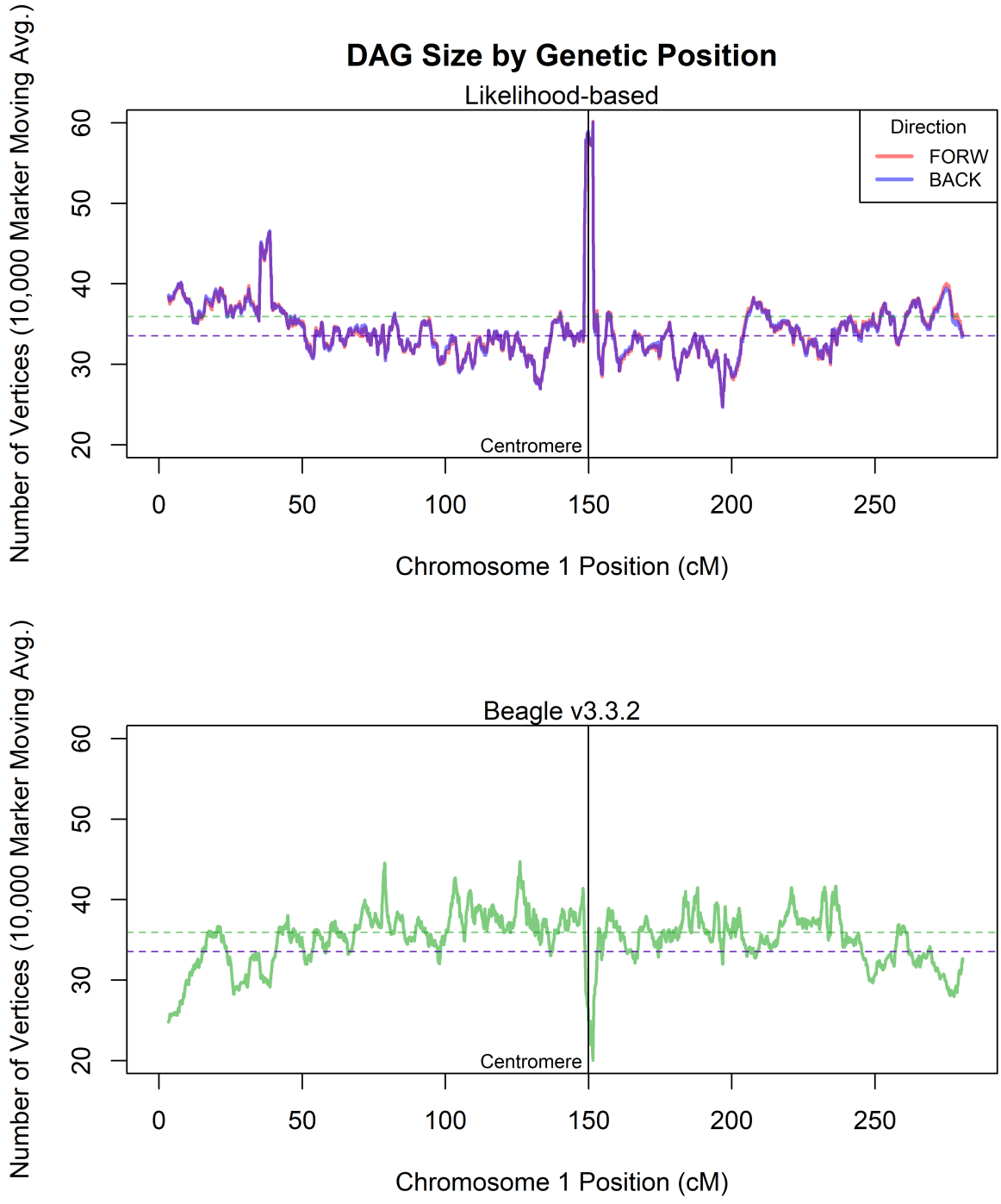

Figure 3.5: *DAG model selection results at the value of the regularization parameter for our likelihood-based method and the scale parameter for Beagle v3.3 that minimizes validation loss. Solid lines show the number of vertices over a 10,000 marker moving average by chromosome position as measured in cM. Dashed lines show the average number of vertices per marker for each model. The likelihood-based DAGs were fit in both directions along the chromosome, but their red and blue lines almost completely overlap into purple lines. Beagle results are represented by a green line.*

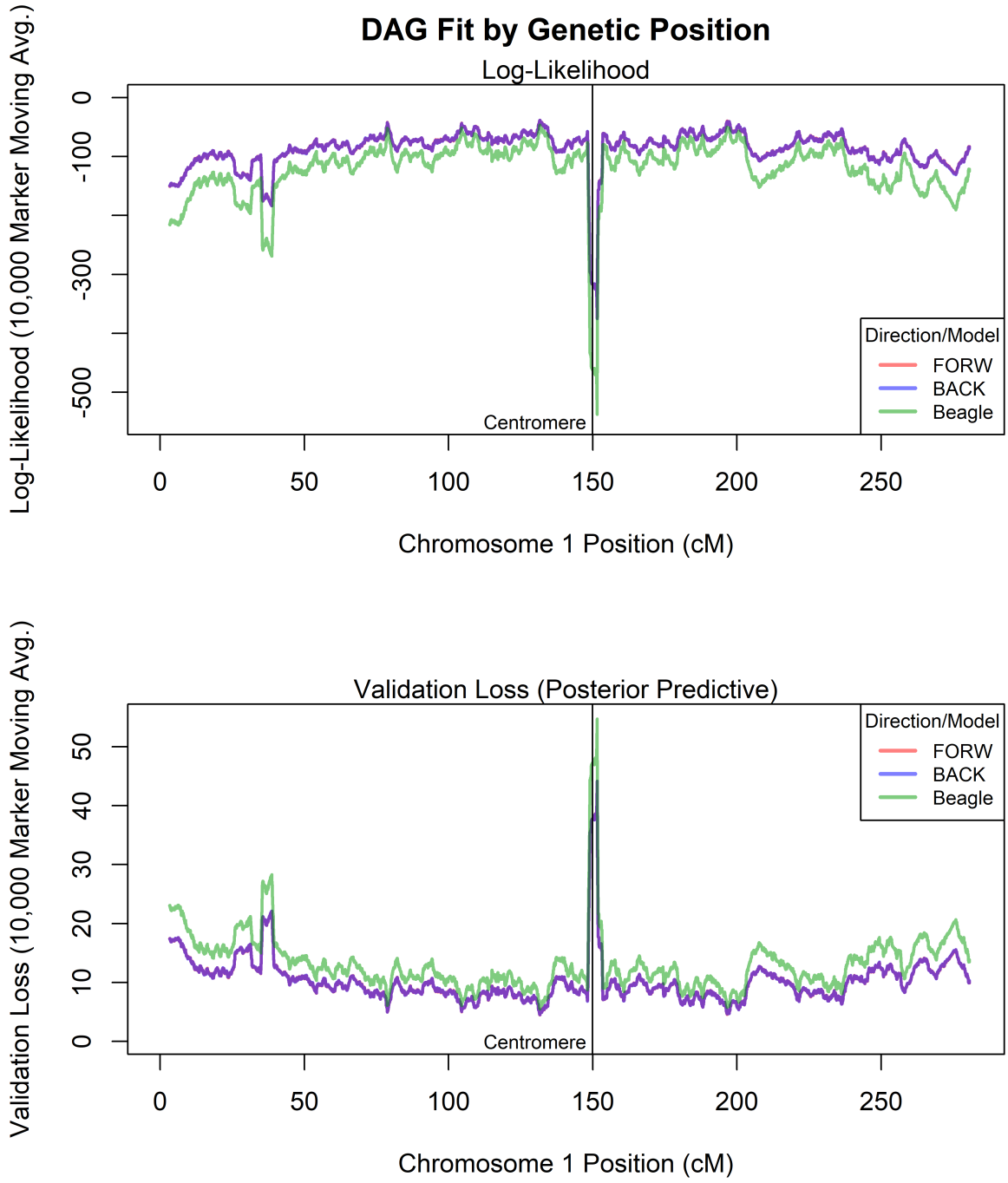

Figure 3.6: *DAG model selection results at the value of the regularization parameter for our likelihood-based method and the scale parameter for Beagle v3.3 that minimizes validation loss. Solid lines show the log-likelihood and validation loss contributions over a 10,000 marker moving average by chromosome position as measured in cM. The likelihood-based DAGs were fit in both directions along the chromosome, but their red and blue lines almost completely overlap into purple lines. Beagle results are represented by a green line.*

this saturated tree model starts with fewer vertices. Regardless, we do not have a satisfactory answer as to why we see the inverted DAG sizes from Figure 3.5. The behavior of both of these methods near the telomeres and centromere definitely merits further research.

Chapter 4

DAG MODEL REVERSIBILITY AND RECONCILIATION

4.1 Introduction and motivation

By their natures, the variable-order Markov chain (VOMC) models described in Chapter 2 are defined in a single direction along a chromosome. The linkage disequilibrium (LD) structure modeled by the conditional dependence relationships represented by a directed acyclic graph (DAG) in one direction does not account for any structure imposed by such relationships that could only be represented by a DAG in the opposite direction. The edge weights which make up the model parameters define conditional probabilities of alleles at a locus ℓ given the alleles at loci *less than* ℓ. However, the model could be reparameterized to the reverse orientation, and the new edge weights would define conditional probabilities of alleles at a locus ℓ given the alleles at loci *greater than* ℓ. Recall that merged vertices induce conditional independencies which result in restrictions on the model parameters, but these restrictions may not correspond to any possible mergings of vertices if the model was reparameterized in the opposite direction. That is, the possible model space of these DAGs depends on the direction in which the markers are considered, which is an undesirable property for genomic models in which such orientation should be arbitrary with regard to haplotype frequency modeling.

In this chapter we will seek to understand how the direction of a DAG model affects its possible model space and when the parameter restrictions represented by conditional independencies in one direction can and can not be modeled by a DAG in the opposite direction. That is, we will seek to define and characterize the concept of reversibility in the context of DAG models. Also, we will introduce a method for reconciling DAG models fit in opposite directions along the chromosome. This will allow us to create a single bidirectional

model that unifies the differing LD structure imposed by the directional DAGs, providing us with even further granularity in modeling haplotype frequencies.

As motivation for why we might desire to reconcile DAG models, consider a snapshot of two DAGs fit in opposite directions on the SNP marker data derived from phase 3 of the 1000 Genomes Project described previously in Chapter 3.4 (The 1000 Genomes Project Consortium, 2015). Recall that low coverage whole-genome sequence data were available for 2,504 diploid individuals, and SNP marker selection mirroring that of Wang et al. (2017) resulted in 902,606 high quality biallelic SNPs from chromosome 1. To further reduce this set of markers, we performed LD pruning using PLINK version 1.07. This pruning employs a sliding window in which markers are recursively discarded on the basis of having genotypic correlation $r^2 > 0.2$ (Purcell et al., 2007). This process resulted in a pruned set of 21,296 markers with a density of approximately 75 SNPs per cM. This reduction in LD was performed in order to allow us to display an interesting structure over a smaller number of markers. The full marker panel with a much higher density of 3,200 SNPs per cM results in a small number of long haplotype segments making for DAGs with little local structure.

Figures 4.1 and 4.2 show a cross section of DAGs fit in both the arbitrarily chosen "forward" and "backward" directions along the chromosome, respectively, encompassing 10 markers from those numbered 12,655 to 12,664. In both cases, the DAGs were created from the likelihood-based method defined in Chapter 3 with the regularization parameter λ equal to 8. Note that this is not the value of λ found to minimize validation loss by cross-validation, but instead a higher value that was chosen to keep the DAG models parsimonious enough to display. Additionally, these figures show how the LD structure modeled by the DAGs can distinguish ancestry in the form of five superpopulations, Africans (AFR), Admixed Americans (AMR), East Asians (EAS), Europeans (EUR) and South Asians (SAS), gathered within the 1000 Genomes data. Recall that vertices represent sets of haplotypes with paths that terminate at them in the DAG. Within each vertex, we have included a pie chart illustrating the proportion of haplotypes contained within that vertex belonging to each of the five superpopulations. Also, each vertex is proportional in area to the number of

haplotypes it contains.

Perhaps the most readily observable and noteworthy aspect of Figures 4.1 and 4.2 is how individuals from the African superpopulation have their own distinct paths through both DAGs and that those vertices that contain nearly exclusively Africans make up around one-third of the total number. Even in this relatively parsimonious DAG, Africans are easily distinguished from non-Africans and show evidence of greater genetic diversity and more population substructure, which concurs with prior genetic analyses (The 1000 Genomes Project Consortium, 2015; Tishkoff et al., 2009). We also see paths in both DAGs that have a majority of East Asian and South Asian individuals, although these are not as exclusive and pronounced as the African paths.

There are some differences of note between the two DAGs as well. In the "forward" DAG, the path outlined in green has a majority to near majority of European individuals. The only analogous path in the "backward" DAG does not distinguish Europeans from South Asians nearly as well. In the "backward" DAG, the path outlined in purple is the only set of vertices in either DAG that distinguishes Admixed Americans by counting them as at least a plurality of the haplotypes. These differences between the DAGs could become even more pronounced as the regularization parameter λ shrinks and the models become less parsimonious. Allowing DAG models to have different structures depending on the orientation of markers along the chromosome could have an impact on all of the various analyses that these haplotype frequency models may be used for, be it haplotype phasing, ancestry mapping, or IBD segment detection.

4.2 Vertex completeness and reversibility

First, we seek to understand when the vertices representing context-sensitive conditional independencies in one DAG model can be reversed to a corresponding vertex in a DAG model of the opposite direction. Recall that we define a haplotype of length M

$$\mathbf{x} = (x_1, x_2, \ldots, x_M)$$

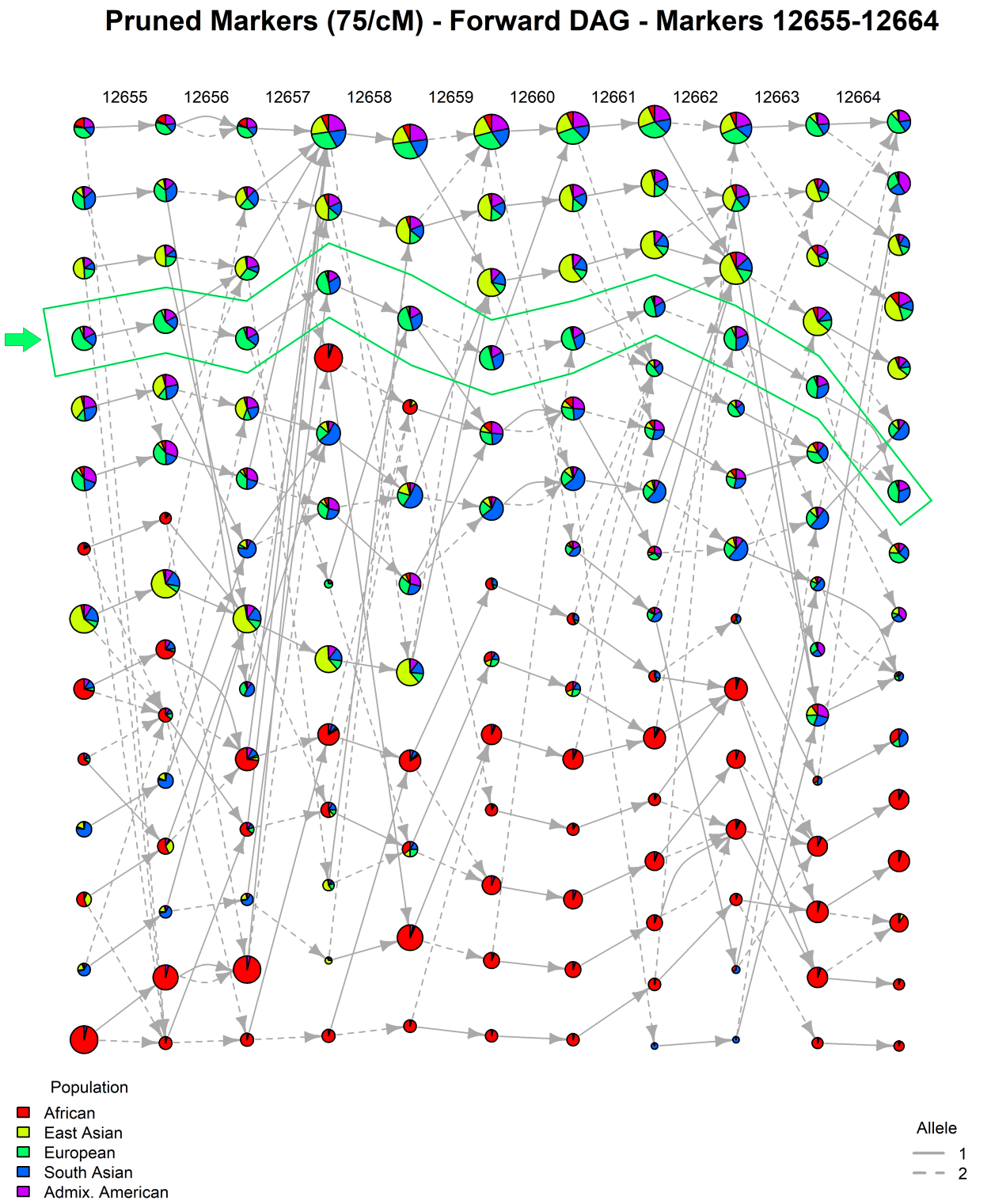

Figure 4.1: *Cross section from markers 12,655 to 12,664 of DAG fit in the "forward" direction along the chromosome using regularization parameter $\lambda = 8$. Each vertex is proportional in area to the number of haplotypes it contains and includes a pie chart of the proportions of haplotypes belonging to the five superpopulations. Solid lines represent edges for allele 1, and dashed lines represent edges for allele 2. A plurality European path is outlined in green.*

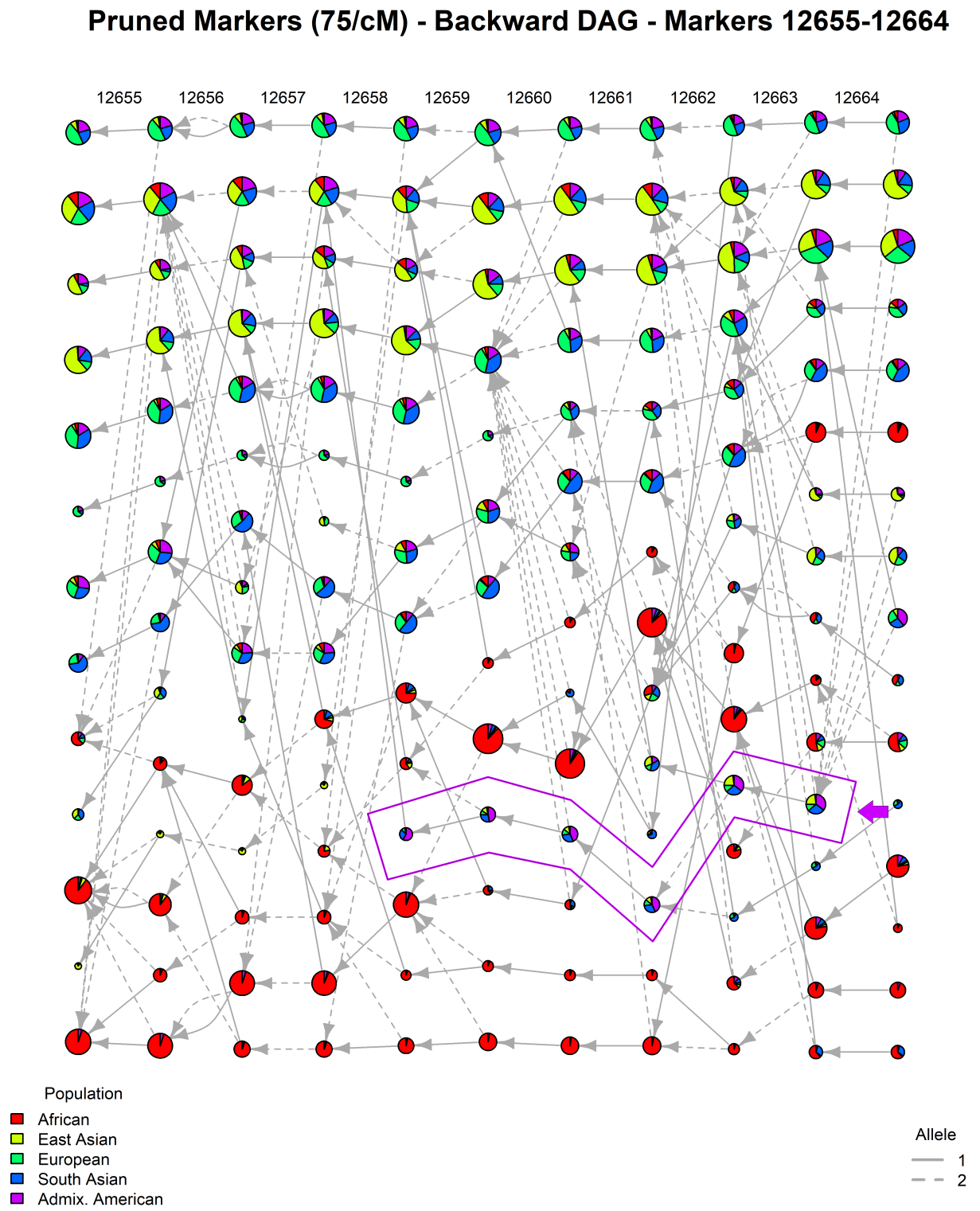

Figure 4.2: *Cross section from markers 12,655 to 12,664 of DAG fit in the "backward" direction along the chromosome using regularization parameter $\lambda = 8$. Each vertex is proportional in area to the number of haplotypes it contains and includes a pie chart of the proportions of haplotypes belonging to the five superpopulations. Solid lines represent edges for allele 1, and dashed lines represent edges for allele 2. A plurality Admixed American path is outlined in purple.*

as a sequence of M markers at loci ordered along a chromosome where the marker x_ℓ at each locus $\ell = 1, \ldots, M$ can take on any value from a set of alleles A_ℓ. For any number of markers $M \geq 1$ and any m such that $0 \leq m \leq M$, we will define a set H_M of haplotypes of length M as *complete from the left with degree m* if and only if there exist alleles $x_{m+1} \in A_{m+1}, \ldots, x_M \in A_M$ such that

$$H_M = \{(x_1, \ldots, x_m, x_{m+1}, \ldots, x_M) | x_\ell \in A_\ell, 1 \leq \ell \leq m\}.$$

That is, if the set is composed of haplotypes with every possible combination of alleles for the first m markers and fixed alleles for the last $M - m$ markers. For example, if we consider biallelic SNP markers, the set of haplotypes

$$\{11121, 11221, 12121, 12221, 21121, 21221, 22121, 22221\} = \{***21\}$$

is complete from the left with degree 3 since all $2^3 = 8$ combinations of the first three markers are paired with the haplotype 21 for the last two markers. The asterisks in the example set represent loci for which every possible allele or combination of alleles are contained within the set.

Similarly, for any number of markers $M \geq 1$ and any m such that $0 \leq m \leq M$, we will define a set H_M of haplotypes of length M as *complete from the right with degree m* if and only if there exist alleles $x_1 \in A_1, \ldots, x_{M-m} \in A_{M-m}$ such that

$$H_M = \{(x_1, \ldots, x_{M-m}, x_{M-m+1}, \ldots, x_M) | x_\ell \in A_\ell, M - m + 1 \leq \ell \leq M\}.$$

That is, if the set is composed of haplotypes with fixed alleles for the fixed $M - m$ markers and every possible combination of alleles for the last m markers. For example, if we again consider biallelic SNP markers, the set of haplotypes

$$\{12111, 12112, 12121, 12122\} = \{121**\}$$

is complete from the right with degree 2 since the haplotype 121 for the first three markers is paired with all $2^2 = 4$ combinations of the last two markers.

We can use this concept of complete sets to characterize the reversibility of vertices within a DAG, to define when the independenicies induced by a vertex from a DAG modeling one direction along the chromosome correspond to independenicies induced by a vertex in the opposite direction. Let $G^{(f)} = (V^{(f)}, E^{(f)}, w^{(f)})$ and $G^{(b)} = (V^{(b)}, E^{(b)}, w^{(b)})$ be DAGs defined in arbitrarily chosen "forward" and "backward" directions, respectively. Recall that vertices in a DAG model represent the set of haplotypes with paths that terminate in that vertex. Now consider a vertex $v_\ell^{(f)} = \{\mathbf{x}_{1:\ell}\}$ containing haplotypes of length ℓ ordered in the "forward" direction along a chromosome with M markers. This vertex induces the following context-sensitive independence:

$$\mathbf{X}_{1:\ell} \perp\!\!\!\perp \mathbf{X}_{\ell+1:M} | \mathbf{x}_{1:\ell} \in v_\ell^{(f)}.$$

If this vertex contains a subset of haplotypes $H_\ell^{(f)}$ that is complete from the left with degree $j \geq 1$, then we have

$$\begin{aligned}
&\mathbf{X}_{1:\ell} \perp\!\!\!\perp \mathbf{X}_{\ell+1:M} | \mathbf{X}_{1:\ell} \in v_\ell^{(f)} \\
&\quad \Longrightarrow \mathbf{X}_{1:\ell} \perp\!\!\!\perp \mathbf{X}_{\ell+1:M} | \mathbf{X}_{1:\ell} \in H_\ell^{(f)}, && \text{since } H_\ell^{(f)} \subseteq v_\ell^{(f)}, \\
&\quad \Longleftrightarrow \mathbf{X}_{1:\ell} \perp\!\!\!\perp \mathbf{X}_{\ell+1:M} | \mathbf{X}_{1:\ell} \in \{* \cdots * \, x_{j+1} \cdots x_\ell\}, && \text{since } H_\ell^{(f)} \text{ is complete from the left with degree } j, \\
&\quad \Longleftrightarrow \mathbf{X}_{1:j} \perp\!\!\!\perp \mathbf{X}_{\ell+1:M} | X_{j+1} = x_{j+1}, \ldots, X_\ell = x_\ell, && \text{since } x_{j+1}, \ldots, x_\ell \text{ are the fixed alleles from } H_\ell^{(f)}, \\
&\quad \Longleftrightarrow \mathbf{X}_{1:j} \perp\!\!\!\perp \mathbf{X}_{j+1:M} | \mathbf{X}_{j+1:M} \in \{x_{j+1} \cdots x_\ell * \cdots *\}, && \text{since every haplotype in this set includes the fixed alleles,} \\
&\quad \Longleftrightarrow \mathbf{X}_{1:j} \perp\!\!\!\perp \mathbf{X}_{j+1:M} | \mathbf{X}_{j+1:M} \in H_{M-j}^{(b)}, && \text{where } H_{M-j}^{(b)} \text{ is complete from the right with degree } M - \ell, \\
&\quad \Longrightarrow \mathbf{X}_{1:j} \perp\!\!\!\perp \mathbf{X}_{j+1:M} | \mathbf{X}_{j+1:M} \in v_{M-j}^{(b)},
\end{aligned} \tag{4.1}$$

for some vertex $v_{M-j}^{(b)} = \{\mathbf{x}_{j+1:M}\}$ in the "backward" direction along the chromosome. That is, if a vertex $v_\ell^{(f)}$ in the "forward" DAG contains a subset that is complete from the left with degree j, then it can be reversed to correspond to a vertex $v_{M-j}^{(b)}$ in the "backward" DAG

which induces the equivalent context-sensitive independence for haplotypes in that subset. Specifically, we have the reversible context-sensitive independencies

$$\mathbf{X}_{1:\ell} \perp\!\!\!\perp \mathbf{X}_{\ell+1:M} | \mathbf{x}_{1:\ell} \in H_\ell^{(f)} \iff \mathbf{X}_{1:j} \perp\!\!\!\perp \mathbf{X}_{j+1:M} | \mathbf{X}_{j+1:M} \in H_{M-j}^{(b)},$$

for sets of haplotypes $H_\ell^{(f)}$ and $H_{M-j}^{(b)}$ that are complete from the left and right, respectively, and contain the same fixed alleles.

Note that any set of haplotypes that is complete from the left or right with degree m must contain subsets that are also complete from the left or right, respectively, with degree m', for all $m' < m$. To see this, consider the set

$$H_M = \{(x_1, \ldots, x_m, x_{m+1}, \ldots, x_M) | x_\ell \in A_\ell, 1 \leq \ell \leq m\}$$

that is complete from the left with degree m and has fixed alleles $x_{m+1} \in A_{m+1}, \ldots, x_M \in A_M$. Now for all $x_m \in A_m$, we have that

$$\{(x_1, \ldots, x_{m-1}, x_m, \ldots, x_M) | x_\ell \in A_\ell, 1 \leq \ell \leq m-1\} \subseteq H_M,$$

where these subsets are all by definition complete from the left with degree $m-1$. By induction, this holds for all $m' < m$, and by a symmetrical argument, for sets that are complete from the right as well. To illustrate this, consider the previous example set of biallelic SNP markers $\{***21\}$ that is complete from the left with degree 3. We can see that both $\{**121\} \subseteq \{***21\}$ and $\{**221\} \subseteq \{***21\}$, where each of these subsets are complete from the left with degree 2. Furthermore, the four subsets $\{*1121\}$, $\{*1221\}$, $\{*2121\}$, and $\{*2221\}$ that are complete from the left with degree 1 are also contained within this set. This is a consequence of Equation 3.4 from Chapter 3, which states that merging vertices induces downstream context-sensitive independencies that recurse to the end of the DAG. In the framework of reversibility, this means that vertices containing complete subsets of haplotypes in one direction along the chromosome that induce context-sensitive independencies corresponding to vertices in the opposite direction also induce all of the requisite downstream independencies corresponding to the downstream merging of vertices in that opposite direction.

There is an important limitation of reversibility to keep in mind when applied to fitting DAG models to haplotype data using the methods described in Chapter 3. Although merged vertices in one direction along the chromosome should induce the merging of certain vertices in the opposite direction, in practice, DAGs in each direction are constructed independently of one another by way of a greedy algorithm that does not ensure a global maximum. This means that these DAG models may not, and usually will not, respect these reversible context-sensitive independencies. It is for this reason that we propose a method for reconciling these discrepencies, resulting in DAG models that are reversible, that estimate the same haplotype frequencies regardless of direction along the chromosome.

As a demonstration of how these concepts of reversibility can play out in practice, consider again the example from Figure 2.1 of a DAG modeling haplotypes created from four biallelic markers $X_1, \ldots, X_4$. In Figure 4.3a, this DAG is labeled as being in the "forward" direction and is accompanied in Figure 4.3b with another DAG modeling the same haplotypes in the "backward" direction. Note that these DAGs were not selected using data and would not necessarily model equivalent haplotype frequencies from data; they were constructed for the purpose of illustration. In addition, consider also parameterizing the haplotype frequency distribution not by conditional probabilities ordered in a direction but instead in the fashion of a Markov random field (MRF) represented by a complete undirected graph with the markers as vertices (Koller & Friedman, 2009). Specifically, if we let C be the set of all cliques in the undirected graph, where a clique is defined as any complete induced subgraph, we can express the joint distribution of the haplotypes as proportional to the product of clique potential functions Φ_c for all cliques $c \in C$ (Clifford & Hammersley, 1971). Note that since our MRF is on a complete graph with four vertices, the set of cliques C consists of all 15 subsets of markers, four singleton sets, six pairwise sets, six sets of size three, and one set of all four markers. For these biallelic markers, the clique potential functions will be defined as

$$\Phi_c(\mathbf{x}_{1:4}) = \exp\left(\beta_c \prod_{j \in c} [x_j = 2]\right) = \exp\left(\beta_c \prod_{j \in c} z_j\right),$$

where $z_j = [x_j = 2]$ is the indicator function for whether allele j is 2. Overall, we have

$$\begin{aligned}\Pr(\mathbf{X}_{1:4} = \mathbf{x}_{1:4}) &\propto \prod_{c \in C} \Phi_c(\mathbf{x}_c) \qquad (4.2)\\ &= \exp(\beta_1 z_1 + \beta_2 z_2 + \beta_3 z_3 + \beta_4 z_4 + \beta_{12} z_1 z_2 + \beta_{13} z_1 z_3 + \beta_{14} z_1 z_4 + \beta_{23} z_2 z_3 + \beta_{24} z_2 z_4 \\ &\quad + \beta_{34} z_3 z_4 + \beta_{123} z_1 z_2 z_3 + \beta_{124} z_1 z_2 z_4 + \beta_{134} z_1 z_3 z_4 + \beta_{234} z_2 z_3 z_4 + \beta_{1234} z_1 z_2 z_3 z_4),\end{aligned}$$

a log-linear model on the indicator variables z_j that includes all possible interaction effects of all orders.

Reparameterizing the haplotype frequency model as an MRF provides us the benefit of observing the effects of the conditional independencies induced by vertex merges in directional DAG models on the parameters of the MRF model on an undirected graph. For example, if we consider vertex $v_{3,1}^{(f)}$ in the "forward" DAG Figure 4.3a, we see that it is the destination of haplotype paths 111 and 211, which means that it imposes the context-sensitive independence

$$\mathbf{X}_{1:3} \perp\!\!\!\perp X_4 | \mathbf{X}_{1:3} \in \{111, 211\}.$$

To see how this translates to the MRF, we can set the odds of $X_4 = 2|\mathbf{X}_{1:3} = 111$ equal to the odds of $X_4 = 2|\mathbf{X}_{1:3} = 211$, so we have

$$\begin{aligned}\frac{\Pr(\mathbf{X}_{1:4} = 1112)}{\Pr(\mathbf{X}_{1:4} = 1111)} = \frac{\Pr(\mathbf{X}_{1:4} = 2112)}{\Pr(\mathbf{X}_{1:4} = 2111)} &\Longrightarrow \frac{\exp(\beta_4)}{1} = \frac{\exp(\beta_1 + \beta_4 + \beta_{14})}{\exp(\beta_1)} \\ &\Longrightarrow \exp(\beta_4) = \exp(\beta_4 + \beta_{14}) \\ &\Longrightarrow \beta_{14} = 0.\end{aligned}$$

This means that the vertex $v_{3,1}^{(f)}$ is equivalent to the parameter restriction $\beta_{14} = 0$ in the MRF. That is, under this specific parameterization, $\beta_{14} = 0$ corresponds to independence between markers X_1 and X_4 given the context $\mathbf{X}_{2:3} = 11$.

Table 4.1 gives a complete characterization of the relationship between the "forward" and "backward" DAG models from Figure 4.3, both in terms of reversibility of vertices and parameter restrictions in the MRF model. For each vertex in the DAGs, the full set of

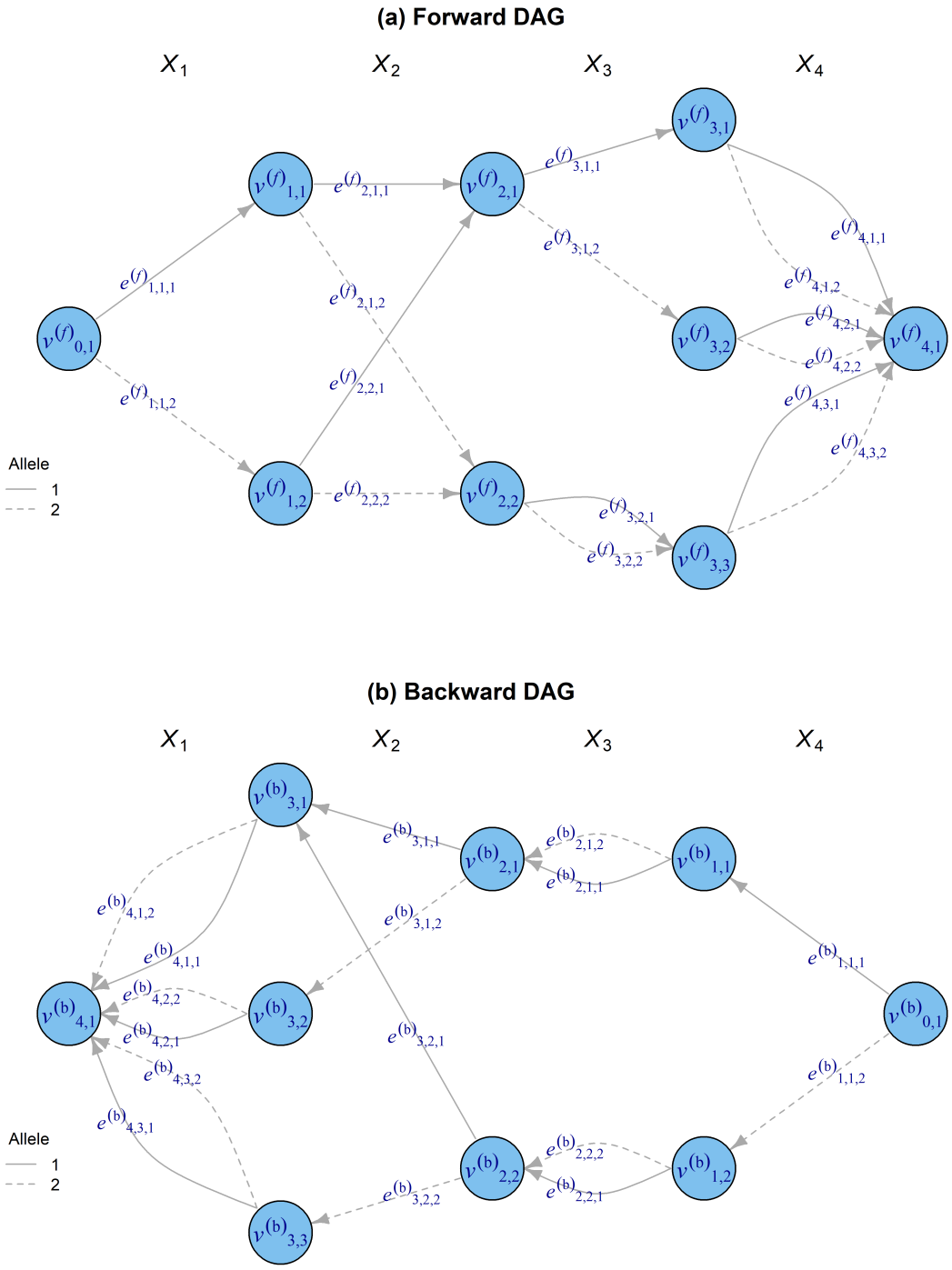

Figure 4.3: *Example DAGs which model haplotype frequencies at four biallelic markers. Figure (a) is in the "forward" direction with edges going from left to right. Figure (b) is in the "backward" direction with edges going from right to left. Solid lines represent edges for allele 1, and dashed lines represent edges for allele 2. These DAGs were constructed to represent different sets of conditional independencies and would not necessarily model equivalent haplotype frequencies from data.*

haplotypes it contains is listed. Note that in the "forward" direction, these haplotypes are written from left to right starting with marker X_1 while in the "backward" direction, they are written from right to left starting with X_4. For instance, haplotype 121 is $\mathbf{x}_{1:3}$ in the "forward" DAG and $\mathbf{x}_{4:2}$ in the"backward" DAG. From these sets of haplotypes, the complete subsets are derived along with the resultant reversed complete subsets in the opposite direction. Then, any vertices from the opposing DAG that contain these reversed complete subsets are listed. These are the vertices that should have been induced by the DAG in the other direction in accordance with Equation 4.1, and they correspond to conditional independencies that are represented in both DAG models that would not need to be reconciled. Finally, any parameter restrictions in the MRF model that are imposed by the vertex are given. Calculations for all of these parameter restrictions can be found in Appendix A.

One notable result from Table 4.1 is that the vertex $v_{2,1}^{(f)}$ imposes two of the same MRF model parameter restrictions as its downstream vertices $v_{3,1}^{(f)}$ and $v_{3,2}^{(f)}$, namely $\beta_{14} = \beta_{134} = 0$. This is to be expected as we have already shown that conditonal independencies are propagated downstream by merged vertices, but it is illustrative of how these conditional independencies directly relate to the parameter restrictions. Also, we can see that by Equation 4.1, the vertex $v_{2,1}^{(f)}$ along with its downstream vertices $v_{3,1}^{(f)}$ and $v_{3,2}^{(f)}$ in the "forward" direction are reversible with the vertex $v_{3,1}^{(b)}$ in the "backward" direction since the set of haplotypes represented by $v_{3,1}^{(b)}$ includes all of the "forward" vertices' reversed complete subsets. Unsurprisingly, $v_{3,1}^{(b)}$ therefore imposes the same parameter restrictions as $v_{2,1}^{(f)}$. Finally, we note that like vertex $v_{2,1}^{(f)}$, vertex $v_{2,2}^{(f)}$ is reversible since it contains a complete subset $\{*2\}$, but unlike $v_{2,1}^{(f)}$, its reversed complete subset $\{2**\}$ is not contained within any single vertex in the "backward" direction. In order for its conditional independecies to be included in both directions, the vertices $v_{3,2}^{(b)} = \{211, 221\}$ and $v_{3,3}^{(b)} = \{212, 222\}$ would need to be merged in the "backward" DAG.

If we combine all of the MRF model parameter restrictions in the "forward" DAG model,

Forward Vertex	Forward Haplotypes	Complete Subsets	Reversed Subsets	Backward Vertices	Parameter Restrictions
$v_{1,1}^{(f)}$	$\{1\}$	—	—	—	—
$v_{1,2}^{(f)}$	$\{2\}$	—	—	—	—
$v_{2,1}^{(f)}$	$\{11, 21\}$	$\{*1\}$	$\{1**\}$	$v_{3,1}^{(b)}$	$\beta_{13} = 0$ $\beta_{14} = 0$ $\beta_{134} = 0$
$v_{2,2}^{(f)}$	$\{12, 22\}$	$\{*2\}$	$\{2**\}$	—	$\beta_{13} + \beta_{123} = 0$ $\beta_{14} + \beta_{124} = 0$ $\beta_{134} + \beta_{1234} = 0$
$v_{3,1}^{(f)}$	$\{111, 211\}$	$\{*11\}$	$\{11*\}$	$v_{3,1}^{(b)}$	$\beta_{14} = 0$
$v_{3,2}^{(f)}$	$\{112, 212\}$	$\{*12\}$	$\{12*\}$	$v_{3,1}^{(b)}$	$\beta_{14} + \beta_{134} = 0$
$v_{3,3}^{(f)}$	$\{121, 221,$ $122, 222\}$	$\{*21\}, \{*22\}$	$\{21*\}, \{22*\}$	—	$\beta_{14} + \beta_{124} = 0$ $\beta_{34} + \beta_{234} = 0$ $\beta_{134} + \beta_{1234} = 0$

Backward Vertex	Backward Haplotypes	Complete Subsets	Reversed Subsets	Forward Vertices	Parameter Restrictions
$v_{1,1}^{(b)}$	$\{1\}$	—	—	—	—
$v_{1,2}^{(b)}$	$\{2\}$	—	—	—	—
$v_{2,1}^{(b)}$	$\{11, 21\}$	—	—	—	$\beta_{13} = 0$ $\beta_{23} = 0$ $\beta_{123} = 0$
$v_{2,2}^{(b)}$	$\{12, 22\}$	—	—	—	$\beta_{13} + \beta_{134} = 0$ $\beta_{23} + \beta_{234} = 0$ $\beta_{123} + \beta_{1234} = 0$
$v_{3,1}^{(b)}$	$\{111, 112,$ $121, 122\}$	$\{1**\}$	$\{*1\}$	$v_{2,1}^{(f)}, v_{3,1}^{(f)}, v_{3,2}^{(f)}$	$\beta_{13} = 0$ $\beta_{14} = 0$ $\beta_{134} = 0$
$v_{3,2}^{(b)}$	$\{211, 221\}$	—	—	—	$\beta_{13} + \beta_{123} = 0$
$v_{3,3}^{(b)}$	$\{212, 222\}$	—	—	—	$\beta_{13} + \beta_{123} + \beta_{134}$ $+\beta_{1234} = 0$

Table 4.1: *Relationship between the "forward" and "backward" DAG models from Figure 4.3. The full set of haplotypes contained by each vertex is listed from left to right in the "forward" direction and right to left in the "backward". The complete subsets and the resultant reversed complete subsets are derived along with any vertices from the opposing DAG that contain these subsets. Any parameter restrictions in the MRF model that are imposed by the vertex are given.*

we get that

$$\beta_{13} = \beta_{14} = \beta_{123} = \beta_{124} = \beta_{134} = \beta_{1234} = 0 \quad \text{and} \quad \beta_{34} = -\beta_{234},$$

which is a total of seven restrictions. This makes sense because this DAG has nine total vertices, the result of seven merges in the saturated DAG of $2^4 = 16$ vertices. Likewise, from the "backward" DAG model, we get that

$$\beta_{13} = \beta_{14} = \beta_{23} = \beta_{123} = \beta_{134} = \beta_{234} = \beta_{1234} = 0,$$

also seven restrictions. However, combining restrictions both directions gives us

$$\beta_{13} = \beta_{14} = \beta_{23} = \beta_{34} = \beta_{123} = \beta_{124} = \beta_{134} = \beta_{234} = \beta_{1234} = 0,$$

only nine total restrictions since many overlap from each direction. The parameter restrictions where the DAG models from these two directions do not overlap are what could cause haplotype frequency estimates derived from them to differ and why reconciliation is required in order for DAGs fit in opposing directions to provide equivalent estimates.

4.3 Method for bidirectional model reconciliation

We have seen that DAG models fit in opposite directions along the chromosome may not encompass the same set of context-sensitive conditional independencies, and while some vertices in one direction can be reversed to correspond to vertices in the other direction, not all can. This means that in practice, haplotype frequency estimates from a DAG are dependent on the arbitrarily chosen direction in which markers are read, a clearly undesirable property of these models. The Beagle software attempts to circumvent this problem by alternating directions with each iteration of its phasing algorithm, but this still does not account for how different conditional independencies are represented in each direction (Browning & Browning, 2007). Although phase estimates may stabilize with successive iterations, there is no reason to believe that the observed state probability estimates from their hidden Markov model would converge when the structure of the underlying model differs by orientation.

In order to solve this issue, we propose an algorithm that reconciles haplotype frequency estimates between two DAG models in different directions in a way that respects the LD structure modeled by each.

Our method for reconciling these directional differences amounts to iteratively recomputing the parameters of a DAG model in one direction with respect to the parameter restrictions imposed by the structure of a DAG model in the opposite direction. In doing so, the two models will converge to a single bidirectional model which includes all of the conditional independencies from both. Again, let $G^{(f)} = (V^{(f)}, E^{(f)}, w^{(f)})$ and $G^{(b)} = (V^{(b)}, E^{(b)}, w^{(b)})$ be DAGs defined in arbitrarily chosen "forward" and "backward" directions, respectively, where vertex sets $V^{(f)}$ and $V^{(b)}$, edge sets $E^{(f)}$ and $E^{(b)}$, and weight functions $w^{(f)}$ and $w^{(b)}$ are defined as in Chapter 2 except with the topological ordering of the DAGs oriented in opposite directions. We seek to rewrite the conditional probabilities represented by the edge weights from the "forward" DAG in terms of the edge weights from the "backward" DAG. Utilizing basic properties of probability, we get

$$\begin{aligned}
w^{(f)}(e^{(f)}_{\ell,i,k}) &= \Pr(X_\ell = k | \mathbf{X}_{1:\ell-1} \in v^{(f)}_{\ell-1,i}) = \frac{\Pr(\mathbf{X}_{1:\ell-1} \in v^{(f)}_{\ell-1,i}, X_\ell = k)}{\Pr(\mathbf{X}_{1:\ell-1} \in v^{(f)}_{\ell-1,i})} \\
&= \frac{\sum_j \Pr(\mathbf{X}_{1:\ell-1} \in v^{(f)}_{\ell-1,i}, X_\ell = k, \mathbf{X}_{\ell+1:M} \in v^{(b)}_{M-\ell,j})}{\Pr(\mathbf{X}_{1:\ell-1} \in v^{(f)}_{\ell-1,i})} \\
&= \frac{1}{\Pr(\mathbf{X}_{1:\ell-1} \in v^{(f)}_{\ell-1,i})} \sum_j \Pr(\mathbf{X}_{1:\ell-1} \in v^{(f)}_{\ell-1,i} | X_\ell = k, \mathbf{X}_{\ell+1:M} \in v^{(b)}_{M-\ell,j}) \\
&\qquad \times \Pr(X_\ell = k | \mathbf{X}_{\ell+1:M} \in v^{(b)}_{M-\ell,j}) \\
&\qquad \times \Pr(\mathbf{X}_{\ell+1:M} \in v^{(b)}_{M-\ell,j}) \\
&= \frac{1}{W^{(f)}(v^{(f)}_{\ell-1,i})} \sum_j \alpha^{(b)}_{d(e^{(b)}_{M-\ell,j,k})}(v^{(f)}_{\ell-1,i}) \times w^{(b)}(e^{(b)}_{M-\ell+1,j,k}) \times W^{(b)}(v^{(b)}_{M-\ell,j}),
\end{aligned} \tag{4.3}$$

where d is the destination function defined in Equation 2.1, and

$$\alpha^{(b)}_j(v^{(f)}_{\ell,i}) = \Pr(\mathbf{X}_{1:\ell} \in v^{(f)}_{\ell,i} | \mathbf{X}_{\ell+1:M} \in v^{(b)}_{M-\ell,j}), \tag{4.4}$$

That is, the "forward" edge probabilities are a weighted average of the "backward" edge probabilities, weighted by $\alpha^{(b)}$ and vertex weights in both directions. Recall that vertex

weights are defined in Equation 2.3 as the sum of the probabilities of all haplotype paths entering the vertex with $W^{(f)}(v_{0,1}^{(f)}) = 1$, so if we start to reconcile edge weights in locus order $\ell = 1, \ldots, M$, then the "forward" vertex weight $W^{(f)}(v_{\ell-1,i}^{(f)})$ from Equation 4.3 depends only on previously reconciled edges.

We see that the $\alpha^{(b)}$ weights from Equation 4.4 are defined as probabilities of the histories of markers in the "forward" DAG conditional upon future markers in the backward DAG. If we note that

$$\alpha_1^{(b)}(v_{0,1}^{(f)}) = \Pr(v_{0,1}^{(f)} = \varnothing | \mathbf{X}_{1:M} \in v_{M,1}^{(b)}) = 1,$$

then we can define $\alpha^{(b)}$ recursively by

$$\begin{aligned}
\alpha_j^{(b)}(v_{\ell,i}^{(f)}) &= \Pr(\mathbf{X}_{1:\ell} \in v_{\ell,i}^{(f)} | \mathbf{X}_{\ell+1:M} \in v_{M-\ell,j}^{(b)}) \\
&= \sum_{(i',k):e_{\ell,i',k}^{(f)} \in E^{(f)}} \Pr(\mathbf{X}_{1:\ell-1} \in v_{\ell-1,i'}^{(f)}, X_\ell = k | \mathbf{X}_{\ell+1:M} \in v_{M-\ell,j}^{(b)}) \\
&= \sum_{(i',k):e_{\ell,i',k}^{(f)} \in E^{(f)}} \Pr(\mathbf{X}_{1:\ell-1} \in v_{\ell-1,i'}^{(f)} | X_\ell = k, \mathbf{X}_{\ell+1:M} \in v_{M-\ell,j}^{(b)}) \\
&\qquad \times \Pr(X_\ell = k | \mathbf{X}_{\ell+1:M} \in v_{M-\ell,j}^{(b)}) \\
&= \sum_{(i',k):e_{\ell,i',k}^{(f)} \in E^{(f)}} \Pr(\mathbf{X}_{1:\ell-1} \in v_{\ell-1,i'}^{(f)} | \mathbf{X}_{\ell:M} \in v_{M-\ell,d(e_{M-\ell,j,k}^{(b)})}^{(b)}) \times w^{(b)}(e_{M-\ell,j,k}^{(b)}) \\
&= \sum_{(i',k):e_{\ell,i',k}^{(f)} \in E^{(f)}} \alpha_{d(e_{M-\ell,j,k}^{(b)})}^{(b)}(v_{\ell-1,i'}^{(f)}) \times w^{(b)}(e_{M-\ell,j,k}^{(b)}),
\end{aligned}$$

which expresses the $\alpha^{(b)}$ values of a vertex at one level as a weighted average of the $\alpha^{(b)}$ values of its parent vertices at the previous level, weighted by the appropriate edge weights in the opposite direction. Since the edge weights in Equation 4.3 are reconciled in locus order $\ell = 1, \ldots, M$, we can optimize the recursive formula for $\alpha^{(b)}$ using dynamic programming, only storing the values for the immediately preceding locus.

Once all of the edge weights in the "forward" DAG have been reconciled, the process is mirrored and repeated for the "backward" DAG in the other direction. This continues until the log-likelihoods of the DAGs as defined by Equation 3.1 have converged. At this point, the two DAGs from opposite directions have been reconciled, and either can be used

as the new bidirectional model as their haplotype frequency estimates are now equivalent. We have not offered any assurances that convergence always occurs through this iterative process, but this has never been an issue in our experience. If we seek to combine the likelihood-based model selection method defined in Chapter 3 with reconciliation, then at each value of the regularization parameter λ, we would create DAG models in each direction as before, but after estimating the edge weight parameters using the training data, we would then reconcile the two DAGs. At this point validation loss can be computed on either DAG and cross-validation and optimization proceed as before.

We will demonstrate our reconciliation method on the DAGs modeling haplotypes created from four biallelic markers $X_1, \ldots, X_4$ in Figure 4.3. Recall that we could alternatively parameterize the haplotype frequency distribution on these four markers in the fashion of an MRF as shown in Equation 4.2. For this particular example, we used the parameter values

$$\begin{array}{llll} \beta_1 = -0.2 & \beta_{12} = -1.5 & \beta_{123} = 0.3 & \beta_{1234} = -0.4, \\ \beta_2 = 1.5 & \beta_{13} = 0.6 & \beta_{124} = -0.8 & \\ \beta_3 = 0.5 & \beta_{14} = 0.2 & \beta_{134} = 0.5 & \\ \beta_4 = -0.4 & \beta_{23} = -0.9 & \beta_{234} = -0.8 & \\ & \beta_{24} = 1.5 & & \\ & \beta_{34} = -0.6 & & \end{array}$$

which were chosen to specifically emphasize the parameter restrictions that differed between the "forward" and "backward" DAG models.

Table 4.2 shows the haplotype frequencies derived from this MRF model for each of the 16 possible haplotypes. It also shows frequencies derived from the "forward" and "backward" DAGs using the requisite conditional probabilities in each direction given the MRF model frequencies as well as frequencies from the bidirectional DAG computed by the reconciliation method. Finally, it shows the frequencies that would occur if we assumed independence between the four markers, again given the MRF model frequencies. Note here that we would not claim either of the DAG models to be well-selected for estimating the MRF model haplotype frequencies. These models were chosen because they had a mix of overlapping and conflicting parameter restrictions, making their estimates different enough to require

substantial reconciliation.

We see from this table that different sets of conditional dependencies can have a widely varied effect on haplotype frequencies. In some cases, such as the most common haplotype 1212, reconciling the directional DAGs shrinks the frequency estimate toward what we would expect under independent markers. It shifts from 29.8% and 32.0% in the "forward" and "backward" DAG models, respectively, down to 25.5% in the reconciled bidirectional model, closer to the 19.7% expected under independence. This is not true for all haplotypes, however, as there is still LD present in the reconciled model. The final line of Table 4.2 gives the multilocus LD measure for each haplotype frequency distribution as defined in Liu and Lin (2005). This value ranges from zero to one and is computed as the ratio of the Kullback-Leibler divergence, or relative entropy, between each distribution and the independent distribution assumed under linkage equilibrium to the maximum possible relative entropy. By measuring LD in this manner, we can clearly see that the reconciled bidirectional model gives us haplotype frequencies closer in distribution to independence than either of the directional DAG models.

Figure 4.4 shows the DAG models along with the edge weights and vertex weights used to compute their corresponding haplotype frequencies in Table 4.2. Figure 4.5 shows the same DAG models after the edge and vertex weights have been reconciled using our method. Recall from Table 4.1 that since vertex $v_{2,2}^{(f)}$ contains a complete subset, it is reversible, but since its reversed complete subset is not contained within any single vertex in the "backward" direction, there is no vertex in the "backward" DAG to correspond with it. We observed that the vertices $v_{3,2}^{(b)}$ and $v_{3,3}^{(b)}$ would need to be merged to have a corresponding vertex. Now we see in Figure 4.5 that these two vertices have the same edge weights after reconciliation, so as expected, the method imposed the conditional independencies induced by vertex $v_{2,2}^{(f)}$ in the "forward" DAG onto the vertices $v_{3,2}^{(b)}$ and $v_{3,3}^{(b)}$ in the "backward" DAG. We see a similar equalizing of edge weights elsewhere in these DAGs, such as between vertices $v_{2,1}^{(f)}$ and $v_{2,2}^{(f)}$ and between $v_{1,1}^{(b)}$ and $v_{1,2}^{(b)}$. This example illustrates the effectiveness of our method on a small number of markers.

Haplotype	**MRF**	**Forward DAG**	**Backward DAG**	**Reconciled DAG**	**Independent**
1111	0.027	0.020	0.030	0.041	0.051
1112	0.018	0.015	0.031	0.026	0.070
1121	0.045	0.045	0.035	0.024	0.030
1122	0.016	0.027	0.010	0.015	0.041
1211	0.121	0.163	0.092	0.139	0.143
1212	0.364	0.298	0.320	0.255	0.197
1221	0.081	0.058	0.110	0.082	0.084
1222	0.060	0.107	0.104	0.150	0.116
2111	0.022	0.029	0.044	0.060	0.019
2112	0.018	0.022	0.046	0.038	0.026
2121	0.066	0.066	0.052	0.035	0.011
2122	0.049	0.039	0.015	0.022	0.015
2211	0.022	0.029	0.027	0.025	0.052
2212	0.036	0.053	0.040	0.045	0.072
2221	0.036	0.010	0.032	0.015	0.031
2222	0.016	0.019	0.013	0.027	0.042
Multilocus LD	0.135	0.102	0.094	0.064	0.000

Table 4.2: *Haplotype frequencies for each of the 16 possible haplotypes derived from the MRF model, the "forward" and "backward" DAG models, the reconciled DAG model and under independent markers. Multilocus LD is computed as the ratio of relative entropy between each distribution and the independent distribution to the maximum possible relative entropy.*

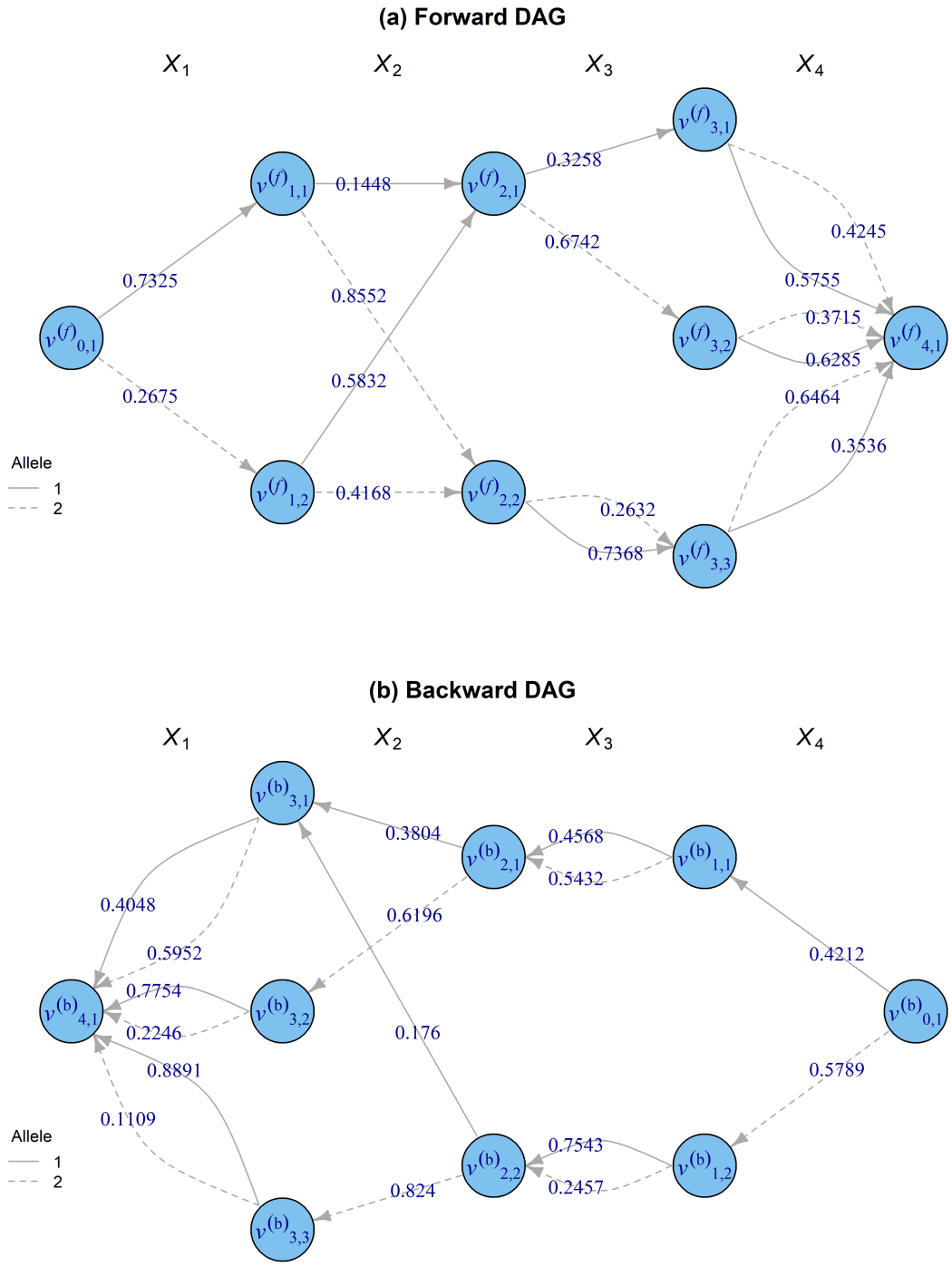

Figure 4.4: *Example DAGs which model haplotype frequencies at four biallelic markers along with edge weights derived from the MRF frequencies using conditional probabilities. Figure (a) is in the "forward" direction with edges going from left to right. Figure (b) is in the "backward" direction with edges going from right to left. Solid lines represent edges for allele 1, and dashed lines represent edges for allele 2.*

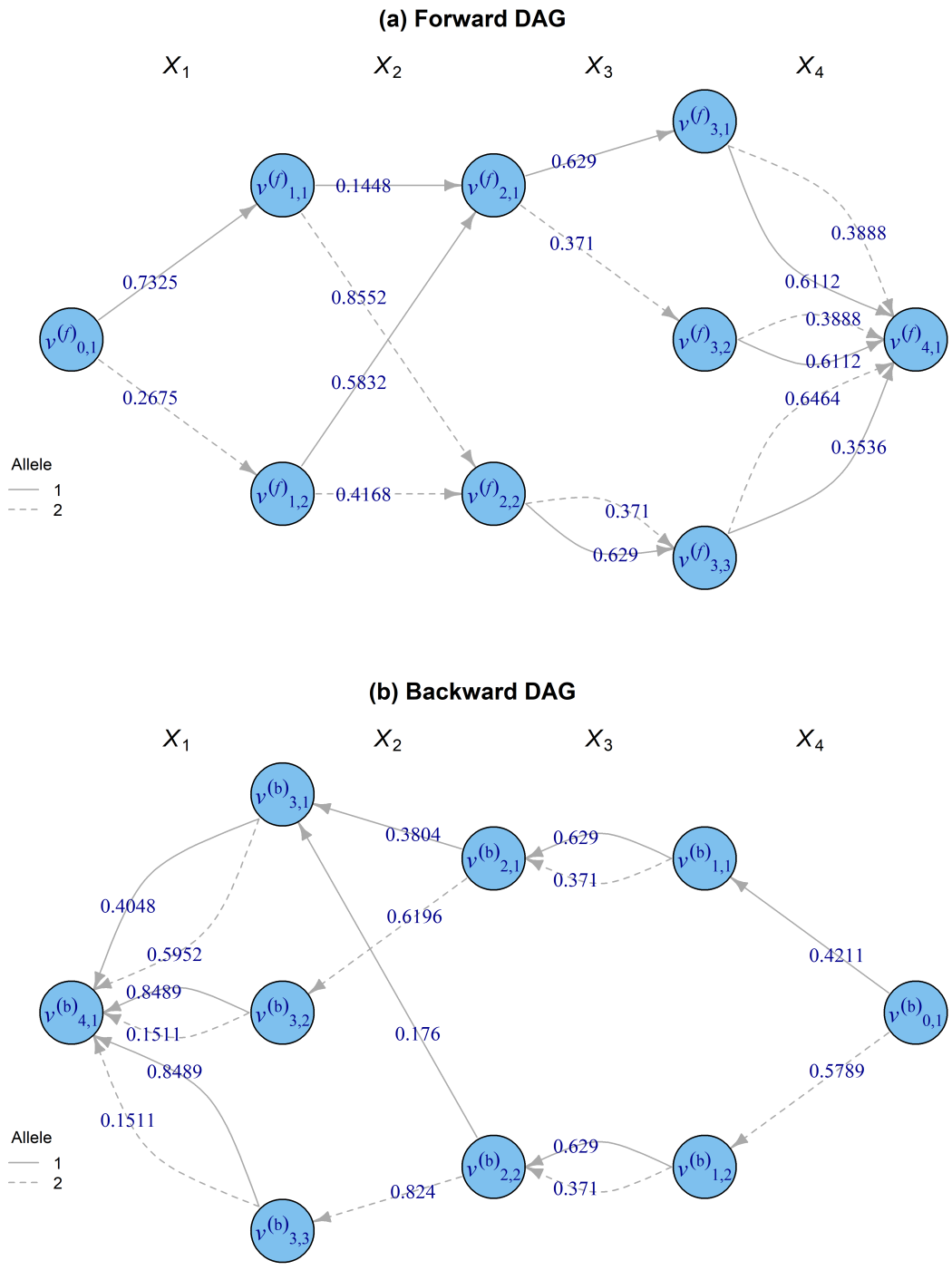

Figure 4.5: *Example DAGs which model haplotype frequencies at four biallelic markers along with edge weights derived from reconciling the DAGs from Figure 4.4. Figure (a) is in the "forward" direction with edges going from left to right. Figure (b) is in the "backward" direction with edges going from right to left. Solid lines represent edges for allele 1, and dashed lines represent edges for allele 2.*

4.4 1000 Genomes reconciliation example

To test our reconciliation method on a large number of markers, we applied it to the same 1000 Genomes Project data used to create Figures 4.1 and 4.2 (The 1000 Genomes Project Consortium, 2015). Recall that these SNP markers were available for 2,504 diploid individuals and pruned from a total of 902,606 down to 21,296 markers with a density of approximately 75 SNPs per cM. Again, DAGs in both directions along chromosome 1 were created from the likelihood-based method defined in Chapter 3 with the regularization parameter $\lambda = 8$.

Performing the reconciliation method on these DAG models introduced a complication that was not present in the small marker example. As we saw in Figure 3.4a, when we have a DAG created from dense markers with high LD, a large majority of its vertices may have only a single edge. Often these edge weights with values of exactly one will be reconciled to have values less than one. This indicates that haplotypes with no existing paths in that DAG do have paths in the DAG of the opposite direction. In order to accommodate these new haplotypes, their paths must be built out by creating new edges and vertices during the reconciliation. This ensures that when the process is complete, both DAG models result in frequency estimates for the same set of haplotypes.

The problem comes in that once we start building out new paths from a vertex, the DAG can start expanding ceaselessly beyond feasibility. Unlike preexisting vertices which contain haplotypes found in the sample, these new vertices cannot be merged to restore parsimony to the DAG. Without sample data, we are unable to compute the likelihoods necessary to compare the differences $\Delta_{\ell,j,j'}$ between two vertices as defined in Equation Equation 3.6. The red line in Figure 4.6 shows the number of vertices at each level of the "forward" DAG during the reconciliation of the first 100 markers in the 1000 Genomes data when new haplotype paths are allowed to branch off without restriction. We can see that after the first 10 to 15 markers the DAG size begins to grow exponentially, in fact doubling at each level as almost every new vertex is accompanied by two new edges. Before the reconciliation even reaches

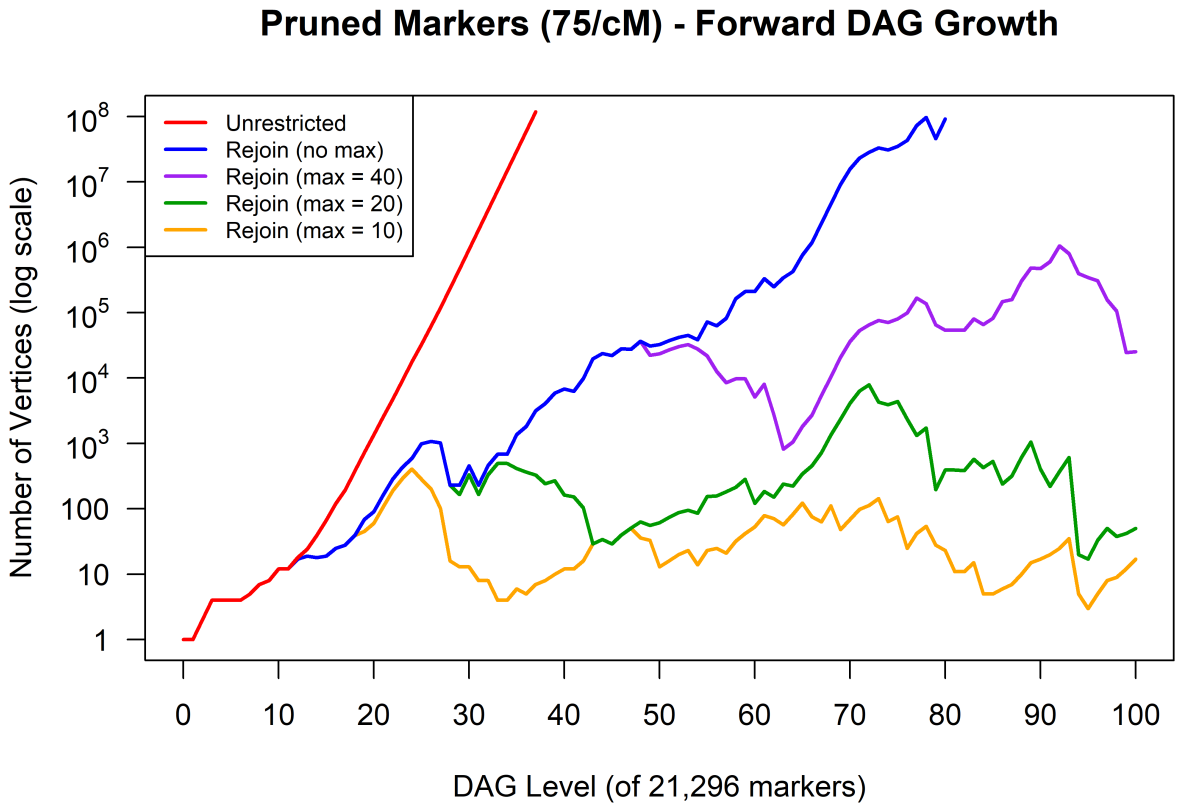

Figure 4.6: *The number of vertices at each level of the "forward" DAG during reconciliation of the first 100 markers in the 1000 Genomes data. The red line is for when new haplotype paths branch off without restriction. The blue line is for when new paths rejoin the DAG when memory is exhausted. The purple, green, and yellow lines are for when new paths rejoin within a maximum path length of 40, 20, and 10 markers, respectively.*

40 markers out of 21,296, there are nearly 100 million vertices, a computationally intractable number.

As an attempt to solve this problem, we considered a method that still allows new haplotype paths to be built out but forces them to eventually rejoin the preexisting DAG. It is important to note that any attempt to permit these paths to rejoin the DAG does effectively involve new vertex merges which induce new context-sensitive independencies that were not previously modeled by the DAG. However, we have no knowlege of the possible

DAG structure on these new paths and no evidence from the sample haplotypes to support these new independencies, so caution must be taken in deciding when and where these paths should rejoin. Our proposed method takes advantage of the fact that no matter what vertex we start in, if we follow the same sequence of alleles, all of those paths will eventually converge to a single vertex. That is, at some point the variable-order Markov chain (VOMC) represented by the DAG runs out of memory, and given enough distance between past and future, the distribution of future markers is independent from past markers. Our idea is that whenever a new path begins, we also start tracking paths from each preexisting DAG vertex at that level corresponding to the same allele sequence. When those paths converge, we allow the new path to rejoin the vertex they converged to. This ensures that even though we are creating new context-sensitive independencies, we are confident that they would have existed regardless, given that based on the existing DAG structure, the memory of the underlying Markov process had been exhausted.

The blue line in Figure 4.6 shows the rate of the "forward" DAG growth when allowing new haplotype paths to rejoin the preexisting DAG during reconciliation. While this growth is notably slower compared to the red line where paths were built without restriction, we are still at over 100 million vertices before reaching even 100 markers. We do believe that this line would eventually stabilize as the rate of rejoins approaches the rate of new path creation, but it is clearly taking too long to do so for this reconciliation to be computationally feasible. In addition, compared to the unpruned high-density marker panel, LD here is relatively low, so in that panel we would expect the VOMC to have even longer memory and new paths to take even longer to merge.

Our final modification was to add a maximum number of markers within which new haplotype paths are forced to rejoin the preexisting DAG. The method proceeds as before, tracking paths from each preexisting vertex until they converge, but in addition, we also track the probabilities of the partial haplotypes along these paths. These probabilities are computed as the product of the initial vertex weight and the edge weights along the path, and when two or more paths meet at a vertex, their haplotype probabilities are summed. If the

paths do not converge to a single vertex within a maximum length, then the new haplotype path rejoins the vertex containing the partial haplotype with the highest probability. Again, we are creating new context-sensitive independencies, but we hope to be creating the ones that most likely would have existed if we had observed haplotypes along those new paths. Note that when the maximum path length is equal to zero, this method amounts to drawing new edges to the preexisting vertex with the highest weight, the vertex containing the most observed haplotypes. In this case, no new vertices are created in the DAG.

The purple, green, and yellow lines in Figure 4.6 show rates of the "forward" DAG growth when new haplotype paths are forced to rejoin within a maximum length of 40, 20, and 10 markers, respectively. We now see that these lines are able to stabilize at a lower number of vertices, decreasing with the path length. Though even with a maximum path length of 10, there can over 100 vertices per marker, which is relatively high. At lower values of the path length, we are able to run the reconciliation method to completion, and some results are shown in Figure 4.7.

First, Figure 4.7a shows the log-likelihood of the reconciled bidirectional model at various values of the maximum path length, from zero to eight. In addition, it shows the average log-likelihood of the "forward" and "backward" DAG models and the log-likelihood of the independence model under complete linkage equilibrium. As expected, the reconciled model falls between the directional DAG models and independence because it inherits the context-sensitive independencies from both directions. It is a good deal closer to independence, however, which might indicate that too many new context-sensitive independencies were created by merging new haplotype paths during reconciliation. Also as expected, the log-likelihood increases with the maximum path length as new haplotype paths are allowed to be longer before merging. However, we note that over the range from zero to eight, the increase is very slight. This could be evidence that path length has little impact on the reconciled log-likelihood, but we simply do not know what would happen with a larger maximum length. Finally, Figure 4.7b shows the average number of vertices per marker between the reconciled "forward" and "backward" DAGs at the same values of maximum path length. We see

that the sizes of the DAGs increase quickly, appearing to approximately double with each increased marker added to the path length. Because of this, we are clearly limited to lower values of the maximum path length to maintain computational tractibility in reconciliation.

This example has illustrated that reconciliation between DAG models fit in opposite direction can be performed on a large number of markers. However, there are concessions that must be made when haplotype paths in one direction do not exist in the other, and at this juncture, it is difficult to measure the impact of those concessions. Further research is required into the effect of merging new haplotype paths back into the preexisting DAG and perhaps into alternative methods for handling this complication.

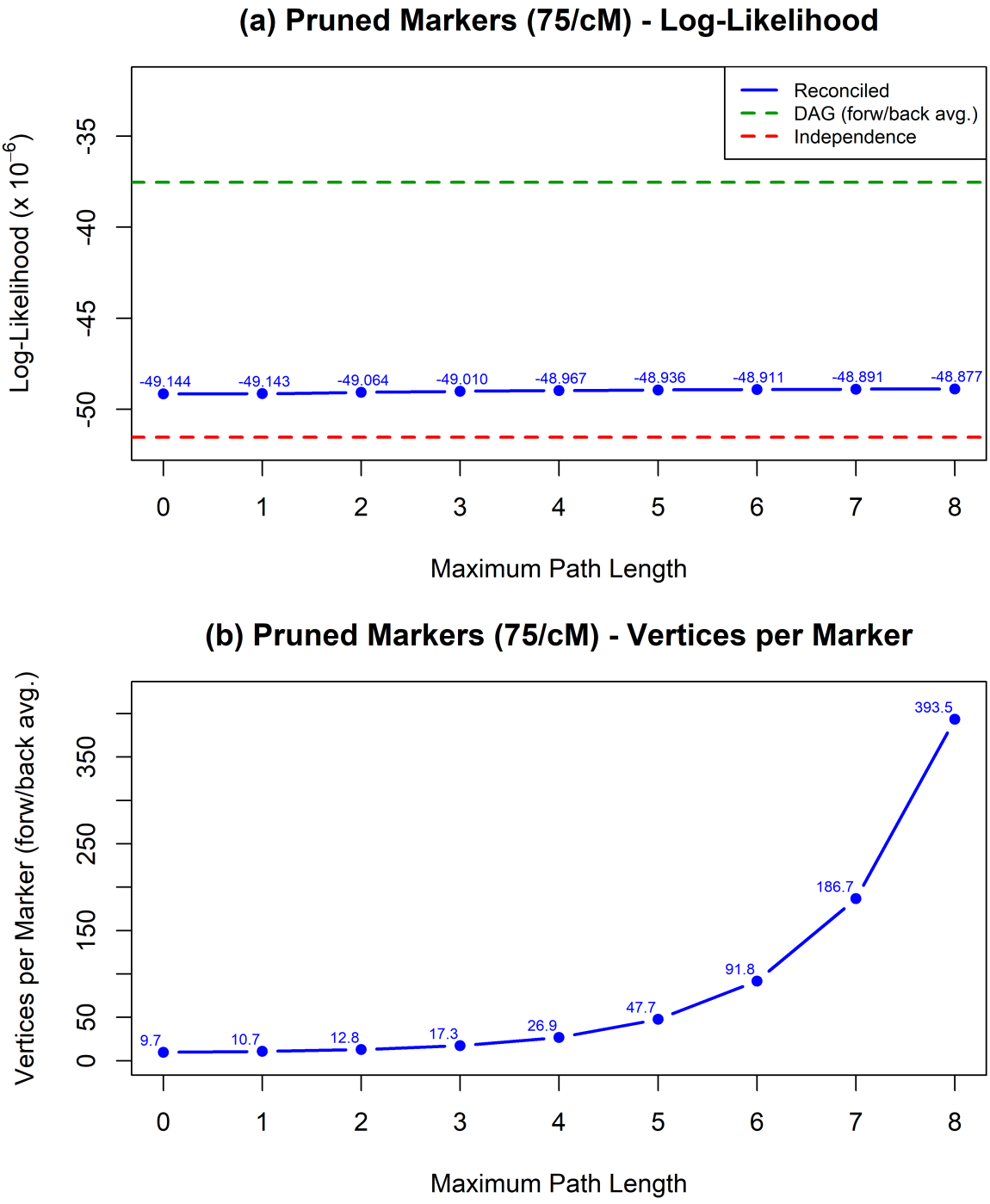

Figure 4.7: *The log-likelihood and average number of vertices per marker of the reconciled bidirectional model at various values of the maximum path length. The green dashed line shows the average log-likelihood of the "forward" and "backward" DAG models. The red dashed line shows the log-likelihood under independent markers.*

Chapter 5

IBD SEGMENT DETECTION

5.1 Previous methods for IBD segment detection

Two copies of DNA are said to be identical by descent (IBD) at a locus if they share genetic material inherited from the same ancestral copy. Because only a small number of recombination events occur between each pair of chromosomes during meiosis, IBD is not distributed randomly throughout the genome but instead appears in contiguous tracts or segments. Figure 5.1 shows an example of the transmission of IBD segments in first cousins through a pedigree modified from Wikimedia Commons (2013). IBD arises where the cousins share haplotypes from the same founder chromosomes, unbroken by recombination in prior meioses.

IBD does not have an absolute measure but is instead defined relative to a founder population. In the context of a pedigree, this will generally be the pedigree founders. However, in the context of population genetics, this will be some ancestral population, often delineated by a time point. Individuals in the same family sharing a recent common ancestor can have IBD segments that are quite long, 10 cM or more (Browning & Browning, 2011). As that common ancestor becomes less recent, more meioses separate the individuals, and more recombinations will break up any shared haplotypes, so IBD segments will become rarer and shorter. Under Haldane's model (Haldane, 1919), IBD segments resulting from a single chain of descent through m meioses will have lengths corresponding to an exponential distribution with rate m Morgans (Donnelly, 1983). Therefore, a pair of thirtieth cousins, separated by 62 meioses, could be expected to have IBD segments of average length 1/62 M, or 1.6 cM.

For a single pair of homologous chromosomes, IBD is binary. At each locus, the two copies of DNA are either IBD or not IBD. For more than two chromosomes, the IBD state

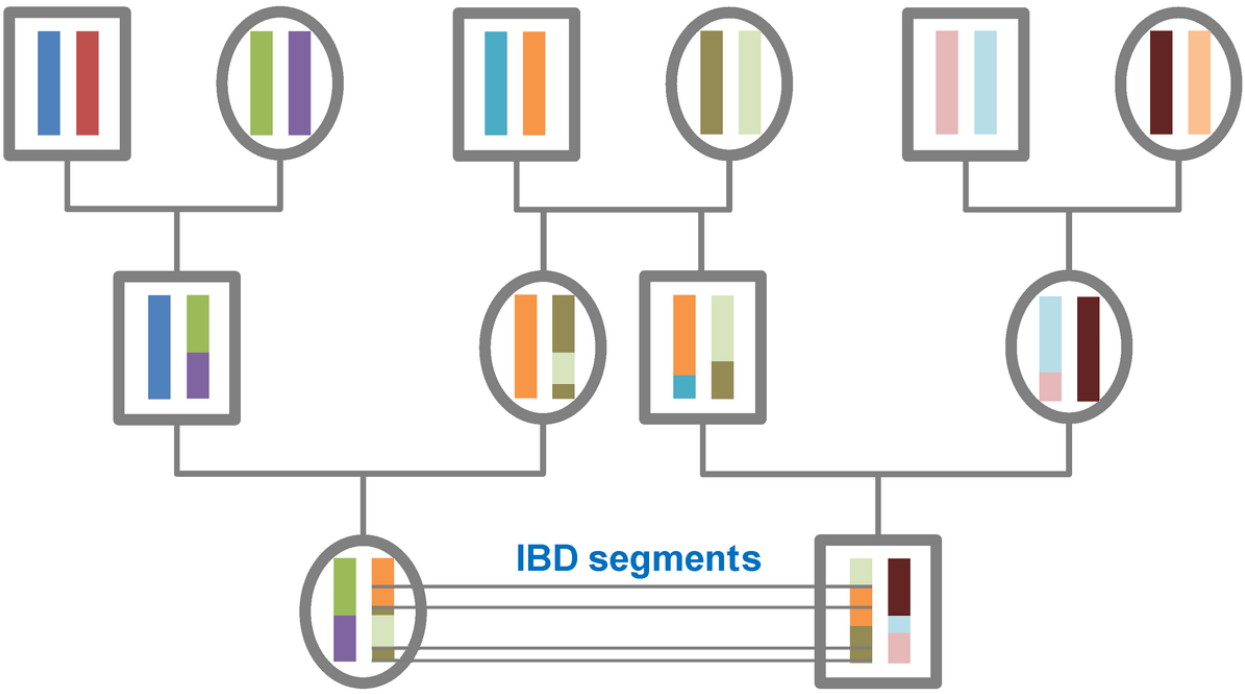

Figure 5.1: *Example transmission of IBD through a pedigree. Founder haplotypes are assigned twelve distinct colors. IBD segments exist in the first cousins at the bottom of the pedigree where they share the same founder haplotype.*

at a locus represents the partition of all DNA copies into equivalence classes for which any pair of copies in the same class are IBD, and any pair in different classes are not IBD. In particular, for a pair of individuals with two sets of chromosomes each, there exist 15 possible IBD states. Table 5.1 shows each of these possible partitions along with the state vector notation of Thompson (1974) and the condensed states of Jacquard (1970) which collapse IBD states that are not genetically distinct for unphased genotypes.

IBD is not observed directly but can instead be inferred from marker data. One of the earliest methods to detect segments of IBD in population data was developed by Leutenegger et al. (2003), who used a hidden Markov model (HMM) where the underlying hidden IBD states could be estimated from observed segments of genotypes which were identitical by state (IBS). This model considered only IBD between two chomosomes within an individual,

IBD State	Diagram	Partition	Thompson Vector	Jacquard Condensed	Kinship Coefficient
1		(abcd)	1111	1	1
2		(ab)(cd)	1122	2	0
3		(abc)(d)	1112	3	$\frac{1}{2}$
4		(abd)(c)	1121	3	$\frac{1}{2}$
5		(ab)(c)(d)	1123	4	0
6		(acd)(b)	1211	5	$\frac{1}{2}$
7		(a)(bcd)	1222	5	$\frac{1}{2}$
8		(a)(b)(cd)	1233	6	0
9		(ac)(bd)	1212	7	$\frac{1}{2}$
10		(ad)(bc)	1221	7	$\frac{1}{2}$
11		(ac)(b)(d)	1213	8	$\frac{1}{4}$
12		(ad)(b)(c)	1231	8	$\frac{1}{4}$
13		(a)(bc)(d)	1223	8	$\frac{1}{4}$
14		(a)(bd)(c)	1232	8	$\frac{1}{4}$
15		(a)(b)(c)(d)	1234	9	0

Table 5.1: *Various representations of the 15 IBD states at a single locus for the four chromosomes of two individuals. In the state diagrams, the top dots represent the DNA copies for individual 1, and the bottom represent those for individual 2, and colors represent different patterns of IBD, red for all four individuals, green for a group of three, yellow for two pairs, and blue for a single pair (Glazner, 2014). In the partition notation, a and b represent the DNA copies for individual 1, and c and d represent those for individual 2. This kinship coefficient is the probability that a pair of randomly selected DNA copies from each individual are IBD.*

often called homozygosity by descent (HBD); however, it did allow for genotyping error. That is, a segment of IBD would not necessarily be broken by a single deviation from IBS in observed genotypes. Purcell et al. (2007) also used an HMM to estimate IBD between two individuals but only modeled probabilities of sharing 0, 1, or 2 alleles IBD, not considering HBD within individuals. Thompson (2008) used Ewens (1972) sampling formula to construct a Markov model for all 15 IBD states among the four chromosomes of two individuals, and Thompson (2009) extended the HMM from Leutenegger et al. (2003) to infer this complete set of IBD states from observed marker data. In Brown et al. (2012) this Markov model was expanded to include additional transitions among the 15 IBD states and investigated the impact of linkage disequilibrium (LD) on IBD segment detection.

None of the above methods account for the presence of LD between markers. Similar to IBD, LD results in tracts of haplotypic similarity. However, the coancestry reflected by LD stems from the history of populations at a longer time frame than IBD. Because of this, segments of IBD tend to be longer than those resulting from LD. Accomodating LD in the inference model can potentially allow for IBD segments from recent coancestry to be differentiated from the background LD in the population.

Albrechtsen et al. (2009) accounts for the presence of pairwise LD by defining emission probabilities in the HMM that are conditional upon the previous marker. In Han and Abney (2011), the LD model is extended from only considering the immediately previous marker to include a fixed length of preceding markers with conditional probabilities approximated using a linear model that avoids the exponential growth of parameters as length increases. As discussed in previous chapters, fixed-order Markov chains are not appropriate for modeling LD due to differences in marker spacing and recombination rates across a chromosome. The Beagle version 3 software models LD with a variable-order Markov chain (VOMC), and Browning (2008) and Browning and Browning (2010) use this estimated directed acyclic graph (DAG) model as the basis for an HMM to detect IBD segments (Browning, 2006). This model reduces the IBD states between two individuals to an IBD or non-IBD binary, ignoring both the possibilities of HBD within an individual and bilinear relatedness through

more than one pair of parents.

Other nonprobabilistic methods allow for rapid IBD segment detection among many pairs of haplotypes. Gusev et al. (2009) creates a dictionary of potentially shared segments and uses genetic length as a criteria for detecting IBD. Browning and Browning (2011) incorporates this same dictionary approach but instead uses haplotype frequency estimates from a DAG model as the criteria. This method is included in Beagle version 3 as a faster alternative to their HMM that can be used to filter for pairs with potential IBD before applying the more computationally intensive method. Beagle version 4 includes a method that further iterates on these by using genetic length for the identification of prospective IBD segments before filtering them based on likelihood ratio scores computed from the DAG model (Browning & Browning, 2013). All IBD detection methods included within the Beagle software use a DAG to model LD. Brown et al. (2012) observed in their simulation study that not accounting for LD when it is present can result in high false positive rates of IBD detection but also observed that Beagle can fail to detect IBD segments in regions where LD is high enough for haplotypic similarity to be comparable in magnitude.

5.2 Hidden Markov model for 15 IBD states

Our method for IBD segment detection among the four chromosomes of two individuals is a further iteration on those discussed above, accommodating transitions among the full set of 15 IBD states as in Brown et al. (2012) yet modeling LD similar to Browning (2008) using the likelihood-based DAG model discussed in previous chapters. Consider four observed haplotypes $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}^{(4)}$ among which we want to find segments of IBD along with a DAG model $G = (V, E, w)$ for which the vertex set V, the edge set E, and the weight function w have been selected and estimated using the methods discussed in Chapter 3 from an independent set of haplotypes that may or may not include $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}^{(4)}$. The hidden state of our HMM jointly represents both LD and IBD. In particular, LD is represented by each haplotype's path through the DAG, corresponding to a sequence of edges in E. Combined with IBD, the joint state $\mathbf{s}_\ell$ at each marker level ℓ can be defined by the ordered

quintuplet

$$\mathbf{s}_\ell = (e_\ell^{(1)}, e_\ell^{(2)}, e_\ell^{(3)}, e_\ell^{(4)}, i_\ell) = (e_\ell^{(1:4)}, i_\ell)$$

where the edge state $e_\ell^{(h)} = e_{\ell,j^{(h)},k^{(h)}}$ represents a marker from haplotype h at locus ℓ having allele $k^{(h)}$ for $h = 1, 2, 3, 4$, $e_\ell^{(1:4)}$ represents the joint states of all four edges at locus ℓ, and $i_\ell = 1, 2, \ldots, 15$ represents the IBD state at locus ℓ.

An HMM requires specification of the transition probabilities among the hidden states and the emission probabilities expressing the likelihoods of observations being generated from those hidden states. Transition probabilities for the hidden joint states along the chromosome are defined as follows:

$$\begin{aligned}
\Pr(\mathbf{s}_\ell|\mathbf{s}_{\ell-1}) &= \Pr(e_\ell^{(1:4)}, i_\ell|e_{\ell-1}^{(1:4)}, i_{\ell-1}) \\
&= \Pr(e_\ell^{(1:4)}|e_{\ell-1}^{(1:4)}, i_\ell, i_{\ell-1}) \times \Pr(i_\ell|e_{\ell-1}^{(1:4)}, i_{\ell-1}) \qquad (5.1) \\
&= \Pr(e_\ell^{(1:4)}|e_{\ell-1}^{(1:4)}, i_\ell) \times \Pr(i_\ell|i_{\ell-1}), \qquad (5.2)
\end{aligned}$$

where the simplifications from lines Equation 5.1 to Equation 5.2 assume that the current edge states are independent from the prior IBD state given both the prior edge states and the current IBD state and that the current IBD state is independent from the prior edge states given the prior IBD state.

The first factor in Equation 5.2 concerns the probabilities of haplotype paths through the DAG and can be expressed using the weight function defined in Equation 2.2. The probability of a transition across edge states along haplotype h is

$$\Pr(e_\ell^{(h)}|e_{\ell-1}^{(h)}) = \begin{cases} w(e_\ell^{(h)}) & \text{if } d(e_{\ell-1}^{(h)}) = j^{(h)} \\ 0 & \text{otherwise} \end{cases},$$

where d is the destination function defined in Equation 2.1 and $j^{(h)}$ is the index for the source vertex of edge $e_\ell^{(h)}$. That is, the probability of transtioning to the current edge from a prior edge is equal to the weight of the current edge if they are adjacent in the DAG, if the destination vertex of the prior edge is the same as the source vertex of the current edge. Otherwise, it is equal to zero. To incorporate IBD state, we use the fact that chromosomes

within the same IBD partition at a locus have the same allele with a single probability while the alleles of chromosomes within different IBD partitions are independent. Thus, we have

$$\Pr(e_\ell^{(1:4)}|e_{\ell-1}^{(1:4)}, i_\ell) = \begin{cases} 0 \quad \text{if any of } e_{\ell-1}^{(h)} \text{ and } e_\ell^{(h)} \text{ are not adjacent} & \\ 0 \quad \text{if alleles } k_\ell^{(1:4)} \text{ do not conform to } i_\ell & \\ \min\left\{w(e_\ell^{(1)}), w(e_\ell^{(2)}), w(e_\ell^{(3)}), w(e_\ell^{(4)})\right\} & \text{if } i_\ell = 1 \\ \min\left\{w(e_\ell^{(1)}), w(e_\ell^{(2)})\right\} \min\left\{w(e_\ell^{(3)}), w(e_\ell^{(4)})\right\} & \text{if } i_\ell = 2 \\ \min\left\{w(e_\ell^{(1)}), w(e_\ell^{(2)}), w(e_\ell^{(3)})\right\} w(e_\ell^{(4)}) & \text{if } i_\ell = 3 \\ \min\left\{w(e_\ell^{(1)}), w(e_\ell^{(2)}), w(e_\ell^{(4)})\right\} w(e_\ell^{(3)}) & \text{if } i_\ell = 4 \\ \min\left\{w(e_\ell^{(1)}), w(e_\ell^{(2)})\right\} w(e_\ell^{(3)}) w(e_\ell^{(4)}) & \text{if } i_\ell = 5 \\ \min\left\{w(e_\ell^{(1)}), w(e_\ell^{(3)}), w(e_\ell^{(4)})\right\} w(e_\ell^{(2)}) & \text{if } i_\ell = 6 \\ w(e_\ell^{(1)}) \min\left\{w(e_\ell^{(2)}), w(e_\ell^{(3)}), w(e_\ell^{(4)})\right\} & \text{if } i_\ell = 7 \\ w(e_\ell^{(1)}) w(e_\ell^{(2)}) \min\left\{w(e_\ell^{(3)}), w(e_\ell^{(4)})\right\} & \text{if } i_\ell = 8 \\ \min\left\{w(e_\ell^{(1)}), w(e_\ell^{(3)})\right\} \min\left\{w(e_\ell^{(2)}), w(e_\ell^{(4)})\right\} & \text{if } i_\ell = 9 \\ \min\left\{w(e_\ell^{(1)}), w(e_\ell^{(4)})\right\} \min\left\{w(e_\ell^{(2)}), w(e_\ell^{(3)})\right\} & \text{if } i_\ell = 10 \\ \min\left\{w(e_\ell^{(1)}), w(e_\ell^{(3)})\right\} w(e_\ell^{(2)}) w(e_\ell^{(4)}) & \text{if } i_\ell = 11 \\ \min\left\{w(e_\ell^{(1)}), w(e_\ell^{(4)})\right\} w(e_\ell^{(2)}) w(e_\ell^{(3)}) & \text{if } i_\ell = 12 \\ w(e_\ell^{(1)}) \min\left\{w(e_\ell^{(2)}), w(e_\ell^{(3)})\right\} w(e_\ell^{(4)}) & \text{if } i_\ell = 13 \\ w(e_\ell^{(1)}) \min\left\{w(e_\ell^{(2)}), w(e_\ell^{(4)})\right\} w(e_\ell^{(3)}) & \text{if } i_\ell = 14 \\ w(e_\ell^{(1)}) w(e_\ell^{(2)}) w(e_\ell^{(3)}) w(e_\ell^{(4)}) & \text{if } i_\ell = 15 \end{cases}, \quad (5.3)$$

where we assume without loss of generality that the observed haplotypes $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ are from one individual and $\mathbf{x}^{(3)}$ and $\mathbf{x}^{(4)}$ are from the other. Note that the alleles $k_\ell^{(1:4)}$ corresponding to the edges $e_\ell^{(1:4)}$ must conform to the IBD state i_ℓ in order to have nonzero probability.

In an earlier version of our HMM, we had tried requring the edges of all haplotype paths within the same IBD partition at a locus to be equal. This allowed for the use of a single edge weight instead of taking the minimum of multiple edge weights for the transition probabilities in Equation 5.3. However, we found in our simulations that IBD segments were often estimated to be slightly shorter than the truth because depending on the amount of LD,

it can take different haplotypes paths quite a few markers of equivalent alleles before they converge into the same vertex. Also, as discussed in Browning and Browning (2010), taking the minimum edge weight has the benefit of downweighting transition probabilities when IBD haplotype paths are not traveling through the same edges in the DAG. One alternative solution could be to require edges within the same IBD partition to be equal but allow transitions to have nonzero probabilities when edges are not adjacent. This would allow the kind of jumps from unconnected vertices that we might expect when recombinations break up regions of LD, but if the nonzero probabilities are not properly tuned, the impact of the entire LD model could be muted. This is an issue that we would like to investigate further in the future.

The second factor in Equation 5.2 concerns transitions of the IBD states along the genome. At a single locus, the state space IBD among multiple chromosomes can be represented by the set of possible partitions of those chromosomes. As described in Thompson (2008), one probability model for these partitions can be derived from Ewens's sampling formula where the single parameter β is the probability that two of the chromosomes are IBD (Balding & Nichols, 1994; Ewens, 1972). To construct the transition matrix of probabilities for IBD state changes along the genome for our HMM, we follow Brown et al. (2012) and consider a modified Chinese restaurant process (Aldous, 1985). In this process, we allow a new chromosome to enter the set and either join any of the existing partitions of size j at rate $j\beta$ or create its own new partition with rate $\eta = 1 - \beta$. The modification then simultaneously requires one of the chromosomes to be randomly removed from the set with equal probability in order to maintain the same number of total chromosomes. If the newly added chromosome is not the one removed, then it assumes the identity of the removed one.

Table 5.2 shows the transition rate matrix resulting from this process on four chromosomes. As an example, consider transitions from the state (abc)(d). The new chromosome enters the (abc) partition at rate 3β, and removal of (d) results in state (abcd). It enters the (d) partition at rate β, and removal of any of (abc) results in one of the three states with two IBD pairs such as (ab)(cd). Finally, it creates its own new partition at rate η, and

removal of any of (abc) results in one of the six states with one IBD pair such as (ab)(c)(d). Note that as a transition rate matrix, the diagonal entries are defined such that the sum of each row equals zero. This stochastic process can be shown to have stationary distribution equal to the probability distribution derived from Ewens's sampling formula (Brown et al., 2012).

The transition matrix in Table 5.2 is unscaled with regard to the rate at which IBD state transitions occur along the chromosome. This is controlled by the overall rate parameter α, which is multiplied by the unscaled transition matrix along with marker distances to obtain the scaled rates of transtions per centimorgan. As shown in (Brown et al., 2012), the rate of transitions given by the Chinese restaurant process for four chromosomes is $\alpha[4\beta + (1-\beta)]$ per centimorgan, which is approximately equal to α per centimorgan when the pairwise IBD probability β is small. This results in IBD segments between two chromosomes of approximately length $(2\alpha)^{-1}$ cM.

Finally, emission probabilities of observed haplotypes $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}^{(4)}$ from the hidden states $\mathbf{s} = (\mathbf{s}_1, \ldots, \mathbf{s}_M)$ are given by

$$\begin{aligned}
\Pr(x_\ell^{(1:4)}|\mathbf{s}_\ell) &= \Pr(x_\ell^{(1:4)}|e_\ell^{(1:4)}, i_\ell) \\
&= \Pr(x_\ell^{(1)}, x_\ell^{(2)}, x_\ell^{(3)}, x_\ell^{(4)}|e_{\ell,j^{(1)},k^{(1)}}, e_{\ell,j^{(2)},k^{(2)}}, e_{\ell,j^{(3)},k^{(3)}}, e_{\ell,j^{(4)},k^{(4)}}) \\
&= \prod_{h=1}^{4} \left[(1-\epsilon) I(x_\ell^{(h)} = k^{(h)}) + \epsilon I(x_\ell^{(h)} \neq k^{(h)}) \right],
\end{aligned}$$

where ϵ is the parameter representing the rate of genotyping error. That is, each edge in the hidden state emits its corresponding allele with probability $(1-\epsilon)$ and emits the alternative allele with probability ϵ. Note that these emission probabilities do not depend on the IBD state, so the observed alleles do not need to conform to the IBD state at any given level. This allows inferred segments of IBD to remain unbroken even when IBS is occasionally violated at individual markers.

To detect IBD segments, we used the Viterbi algorithm, a dynamic programming algorithm that computes the most probable sequence of hidden states, called a Viterbi path,

	IBD State	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
1	(abcd)	—	0	η	η	0	η	η	0	0	0	0	0	0	0	0
2	(ab)(cd)	0	—	2β	2β	2η	2β	2β	2η	0	0	0	0	0	0	0
3	(abc)(d)	3β	β	—	0	η	0	0	0	β	β	η	0	η	0	0
4	(abd)(c)	3β	β	0	—	η	0	0	0	β	β	0	η	0	η	0
5	(ab)(c)(d)	0	2β	2β	2β	—	0	0	0	0	0	β	β	β	β	2η
6	(acd)(b)	3β	β	0	0	0	—	0	η	β	β	η	η	0	0	0
7	(a)(bcd)	3β	β	0	0	0	0	—	η	β	β	0	0	η	η	0
8	(a)(b)(cd)	0	2β	0	0	0	2β	2β	—	0	0	β	β	β	β	2η
9	(ac)(bd)	0	0	2β	2β	0	2β	2β	0	—	0	2η	0	0	2η	0
10	(ad)(bc)	0	0	2β	2β	0	2β	2β	0	0	—	0	2η	2η	0	0
11	(ac)(b)(d)	0	0	2β	0	β	2β	0	β	2β	0	—	β	β	0	2η
12	(ad)(b)(c)	0	0	0	2β	β	2β	0	β	0	2β	β	—	0	β	2η
13	(a)(bc)(d)	0	0	2β	0	β	0	2β	β	0	2β	β	0	—	β	2η
14	(a)(bd)(c)	0	0	0	2β	β	0	2β	β	2β	0	0	β	β	—	2η
15	(a)(b)(c)(d)	0	0	0	0	2β	0	0	2β	0	0	2β	2β	2β	2β	—

Table 5.2: *Unscaled IBD state transition rate matrix resulting from the modified Chinese restaurant process on four chromosomes. The parameter β is the pairwise IBD probability, and $\eta = 1 - \beta$. The diagonal entries x_{jj} are not shown, but are defined such that the sums of the rows equal zero. That is, $x_{jj} = -\sum_{i \neq j} x_{ji}$ for all $j = 1, \ldots, 15$.*

given an HMM and a sequence of observations (Forney, 1973). In our specific case, this Viterbi path is the most probable sequence of IBD states given the haplotype marker data. The Viterbi algorithm operates by maximizing functions of the transition and emission probabilities at each level of the HMM over all possible values of the hidden state at that level. Since our HMM has 15 IBD states and $2^4 = 16$ possible edge traversals at each level, this would result in $15 \times 16 = 240$ such maximizations, each across all 240 states, for a total of 57,600 probability computations at each level. For chromosomes with hundreds of thousands of markers, this was computationally infeasible, so we developed a simple method for speeding up the algorithm.

Instead of maximizing over all 240 possible hidden states, we considered only a subset of the states by restricting the possibilities for more than a single edge to emit its alternative allele at each marker and for any edges to emit their alternative alleles when switching IBD state. That is, in performing the Viterbi maximizations, we allowed for an edge to switch to the other allele at a vertex only when doing so to maintain an IBD segment and, even then, only if it was the only edge required to switch in order to maintain a configuration of alleles that conformed to that IBD state. For example, in computing probabilities for transitions from and to the IBD state (abcd), if the observed alleles at marker ℓ had the relationship $x_\ell^{(1)} = x_\ell^{(2)} = x_\ell^{(3)} \neq x_\ell^{(4)}$, then the edge $e_\ell^{(4)}$ was allowed to switch alleles to bring it into parity with the others. Under some IBD states, more than one switch is possible, so in these cases both were computed and tracked for the Viterbi algorithm. For example, if the observed alleles again had the relationship $x_\ell^{(1)} = x_\ell^{(2)} = x_\ell^{(3)} \neq x_\ell^{(4)}$ but the IBD state was instead (ab)(cd), switching either edge $e_\ell^{(3)}$ or $e_\ell^{(4)}$ would achieve a correct configuration. Table 5.3 shows all of the possible edge switches for transitions maintaining each of the 15 IBD states.

Under this edge switching method, each IBD state now has at most three possible combinations of edge traversals: 1) make no edge switches; 2) make one possible edge switch; or 3) make the other possible edge switch, if available. Most of the IBD states allow for two possible edge switches, but five of them allow for only one, and state (a)(b)(c)(d) allows for

IBD State	Partition	Allowed Switches
1	(abcd)	$a = b = c \neq \mathbf{d}$ $a = b = d \neq \mathbf{c}$ $a = c = d \neq \mathbf{b}$ $b = c = d \neq \mathbf{a}$
2	(ab)(cd)	$a = b = \mathbf{c} \neq \mathbf{d}$ $a = b = \mathbf{d} \neq \mathbf{c}$ $\mathbf{a} = c = d \neq \mathbf{b}$ $\mathbf{b} = c = d \neq \mathbf{a}$
3	(abc)(d)	$a = b \neq \mathbf{c}$ $a = c \neq \mathbf{b}$ $\mathbf{a} \neq b = c$
4	(abd)(c)	$a = b \neq \mathbf{d}$ $\mathbf{a} \neq b = d$ $a = d \neq \mathbf{b}$
5	(ab)(c)(d)	$\mathbf{a} \neq \mathbf{b}$
6	(acd)(b)	$\mathbf{a} \neq c = d$ $a = c \neq \mathbf{d}$ $a = d \neq \mathbf{c}$
7	(a)(bcd)	$a = \mathbf{b} \neq c = d$ $a = \mathbf{c} \neq b = d$ $a = \mathbf{d} \neq b = c$
8	(a)(b)(cd)	$b = \mathbf{c} \neq \mathbf{d}$ $b = \mathbf{d} \neq \mathbf{c}$ $\mathbf{c} \neq b = \mathbf{d}$ $\mathbf{d} \neq b = \mathbf{c}$
9	(ac)(bd)	$a = \mathbf{b} = c \neq \mathbf{d}$ $\mathbf{a} = b = d \neq \mathbf{c}$ $a = c = \mathbf{d} \neq \mathbf{b}$ $b = \mathbf{c} = d \neq \mathbf{a}$
10	(ad)(bc)	$\mathbf{a} = b = c \neq \mathbf{d}$ $a = \mathbf{b} = d \neq \mathbf{c}$ $a = \mathbf{c} = d \neq \mathbf{b}$ $b = c = \mathbf{d} \neq \mathbf{a}$
11	(ac)(b)(d)	$\mathbf{a} \neq \mathbf{c}$
12	(ad)(b)(c)	$\mathbf{a} \neq \mathbf{d}$
13	(a)(bc)(d)	$\mathbf{b} \neq \mathbf{c}$
14	(a)(bd)(c)	$\mathbf{b} \neq \mathbf{d}$
15	(a)(b)(c)(d)	—

Table 5.3: *All possible edge switches that allow for maintenance of each of the 15 IBD states. For each set of equivalences, the bolded haplotype can have its allele switched in order to conform to the corresponding IBD state. If two haplotypes are bolded, then either allele could be switched. For example, to achieve the IBD state (ab)(cd) if the alleles are such that* $a = b = \mathbf{c} \neq \mathbf{d}$*, then either allele* **c** *can be switched to be equal to* **d***, or allele* **d** *can be switched to be equal to* **c***; hence they are both bolded. Where an allele is missing from a set of equivalencies, its relationship to the other alleles is irrelevant.*

none. This results in only 38 total hidden states, down from 240, and because edge switches are allowed only during transitions that maintained IBD states, not for transitions to new IBD states, the total number of probability computations at each level is reduced to 634, down from 57,600, speeding up the algorithm by a factor of nearly 100.

For the most part, this reduction in hidden states mostly eliminates transitions that are either impossible, such as edges not conforming to the IBD state, or improbable, such as more than one genotype error at the same locus. However, in the less restrictive IBD states like (a)(b)(c)(d), the inability to account for genotyping error by switching edges could result in poor estimation of the LD structure in regions near errors. One possible way to solve this is to preprocess each haplotype using a simple Viterbi algorithm that attempts to use the DAG model to correct these genotyping errors. This algorithm is based upon an HMM with edges as hidden states that transition along the chromosome with respect to the DAG and emit the observed alleles. We explore this as potential solution in our example.

5.3 1000 Genomes IBD segment detection example

To test our method for IBD segment detection, we applied it to the SNP marker data derived from phase 3 of the 1000 Genomes Project described previously in Chapter 3.4 (The 1000 Genomes Project Consortium, 2015). Low coverage whole-genome sequence data were available for 2,504 diploid individuals, and SNP marker selection mirroring that from Wang et al. (2017) resulted in 902,606 high quality biallelic SNPs from chromosome 1. To generate related individuals from these data, the R package `rref` was used to simulate inheritance patterns for pairs of individuals given a pedigree and founder haplotypes from the 1000 Genomes sample (Wang, 2018). Any DAGs used to model the LD present in the population were created from the likelihood-based method defined in Chapter 3 with the regularization parameter $\lambda = 0.5$, the value found to minimize validation loss by cross-validation.

For our first examples, we simulated inheritance patterns according to the pedigree in Figure 5.2. The founders labeled 11 and 12 were selected randomly from the 1000 Genomes subjects, and the haplotypes from individuals labeled 31 and 32 were output for IBD segment

detection. This pedigree, in which the siblings of interest 31 and 32 were themselves the children of siblings 21 and 22, was chosen in order to allow for the possibility of all 15 IBD states to arise in the simulation. This resulted in a kinship coefficient of $5/16 = 0.3125$, elevated from the kinship of $1/4 = 0.25$ for siblings from unrelated parents. The realized kinship for our particular simulated example was 0.2905 and included some regions in IBD state (abcd), where kinship is equal to 1.

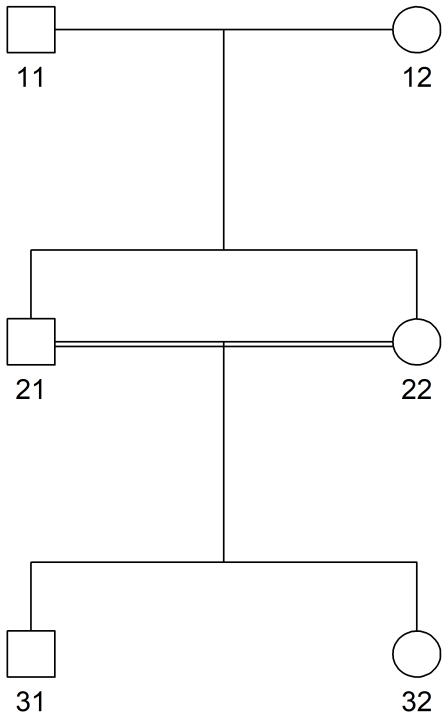

Figure 5.2: *Pedigree used for the simulated example to test IBD segment detection methods. Siblings 31 and 32 from sibling parents 21 and 22 were used to obtain a greater variety of possible IBD states. The kinship coefficient between these two individuals is 5/16 = 0.3125.*

Figure 5.3 shows the results from the Viterbi algorithm on our HMM using DAGs to control for LD when no genotyping error was introduced into the simulations. For the sake of simplicity in displaying the results, the 15 IBD states were collapsed to their corresponding kinship coefficients as given in Table 5.1. The black line in each plot represents the

realized kinship between the simulated individuals at each marker position. Since IBD here is relative to founders only two generations removed, segments of IBD can be quite long, some extending nearly 100 cM. The red lines represent the kinship inferred from the haplotype data under different models and methods. The first two plots give the results from our method using parameter values $\alpha = 10^{-4}$ for the overall transition rate, $\beta = 0.05$ for the pairwise IBD probability, and $\epsilon = 10^{-5}$ for the genotyping error rate. Although there was no genotyping error in these haplotypes, a positive value of ϵ was used to avoid zero probability paths through the HMM. As discussed in Chapter 4, DAG models differ depending on the direction they were fit along a chromosome, so we performed the method in both directions, labeled "forward" and "backward" based on their directions relative to the reference position numbers in the 1000 Genomes variant call format (VCF) file.

As a basis for comparison, we also used the Beagle version 4.1 software to detect IBD segments in the same simulated individuals. Recall that Beagle version 4 combines a non-probabilistic method for initial rapid detection of IBD segments before filtering them using likelihood ratio scores computed from the DAG as a way to control for LD (Browning & Browning, 2013). The HMM method from Beagle version 3 is much more analogous to our method, and we would have preferred to include it as a comparator; however, we were unable to successfully run the software on our large set of 902,606 markers. Even allocating the process as much as 10 gigabytes of RAM on a high performance computing cluster resulted in errors pertaining to lack of memory. Since Beagle performs IBD segment detection on pairs of chromosomes independently from one another, joint IBD states among all four chromosomes were inferred by simple transitivity. That is, if Beagle detects IBD equivalences a = b and b = c among all four chromosomes at a locus, then we will infer the joint IBD state (abc)(d), assuming the IBD equivalence a = b = c by transitivity even if a = c was not detected.

What we see in Figure 5.3 is that our HMM method does a nearly perfect job detecting the IBD segments in the simulated individuals when there is no genotyping error. In the "forward" direction, the inferred kinship averaged across markers is 0.2905, which is slightly

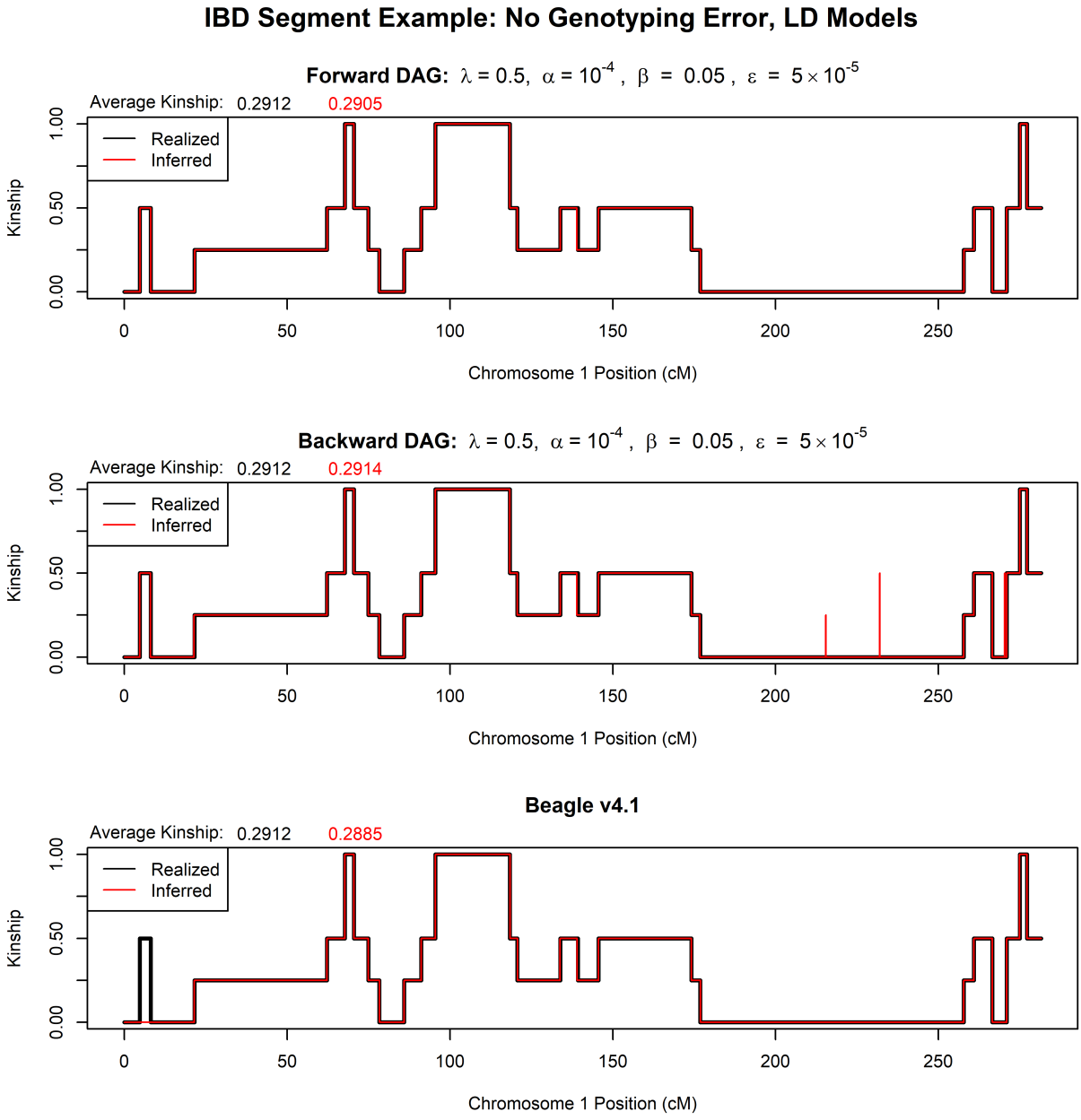

Figure 5.3: *IBD segment detection results from our Viterbi algorithm run in both directions compared to results from Beagle v4.1. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method. Parameters for the Viterbi algorithm are displayed for each result.*

below the average realized kinship of 0.2912 due to the detected IBD segments being a bit shorter than the true realized segments. This is likely an artifact of the downweighting of transition probabilities when haplotype paths are not traveling through the same edges in the DAG after a recombination that resulted in an IBD state change. The "backward" direction looks much the same except for two small blips in the 220-230 cM range where IBD was falsely detected. Closer examination of these blips showed that they occurred in areas with large tracts of IBS despite there being no IBD, likely due to LD. We believe that the backward DAG may fail to model the LD in those regions as well as the forward DAG. Even when the α parameter for the transition rate is decreased down to 10^{-7}, those blips persist (Supplemental Figure B.12). The Beagle IBD segment detection method also performs similarly but misses an IBD segment at around 10 cM. This could possibly be fixed by tuning the IBD detection arguments beyond the recommended defaults, but we did not explore this.

Figure 5.4 shows the results from the Viterbi algorithm on our HMM using the DAG model obtained by setting the regularization parameter at $\lambda = \infty$, which causes every level to collapse to a single vertex as in the independence model at the top of Figure 2.2. This collapsed DAG model represents independent alleles with no LD, so the HMM is analogous to that from Brown et al. (2012), although it is not executed in exactly the same way. As in the last example, we use parameter values $\beta = 0.05$ for the pairwise IBD probability and $\epsilon = 10^{-5}$ for the genotyping error rate, but the overall transition rate parameter is set much lower, to $\alpha = 10^{-8}$ in the first plot and $\alpha = 10^{-14}$ in the second. We see that when LD is not accounted for by the DAG, we get many false positive detections of IBD segments from tracts of IBS that are the actually the result of LD, even when the α parameter is very low. In fact, when $\alpha = 10^{-14}$, true IBD starts to go undetected around 25 cM even though a fair amount of false detection remains. If α is set not much lower, most of the true IBD is lost (Supplemental Figure B.18). These results show the importance of accounting for LD in IBD segement detection, particularly with high-density SNP data.

Figure 5.5 shows the results when genotyping error is introduced into the simulations.

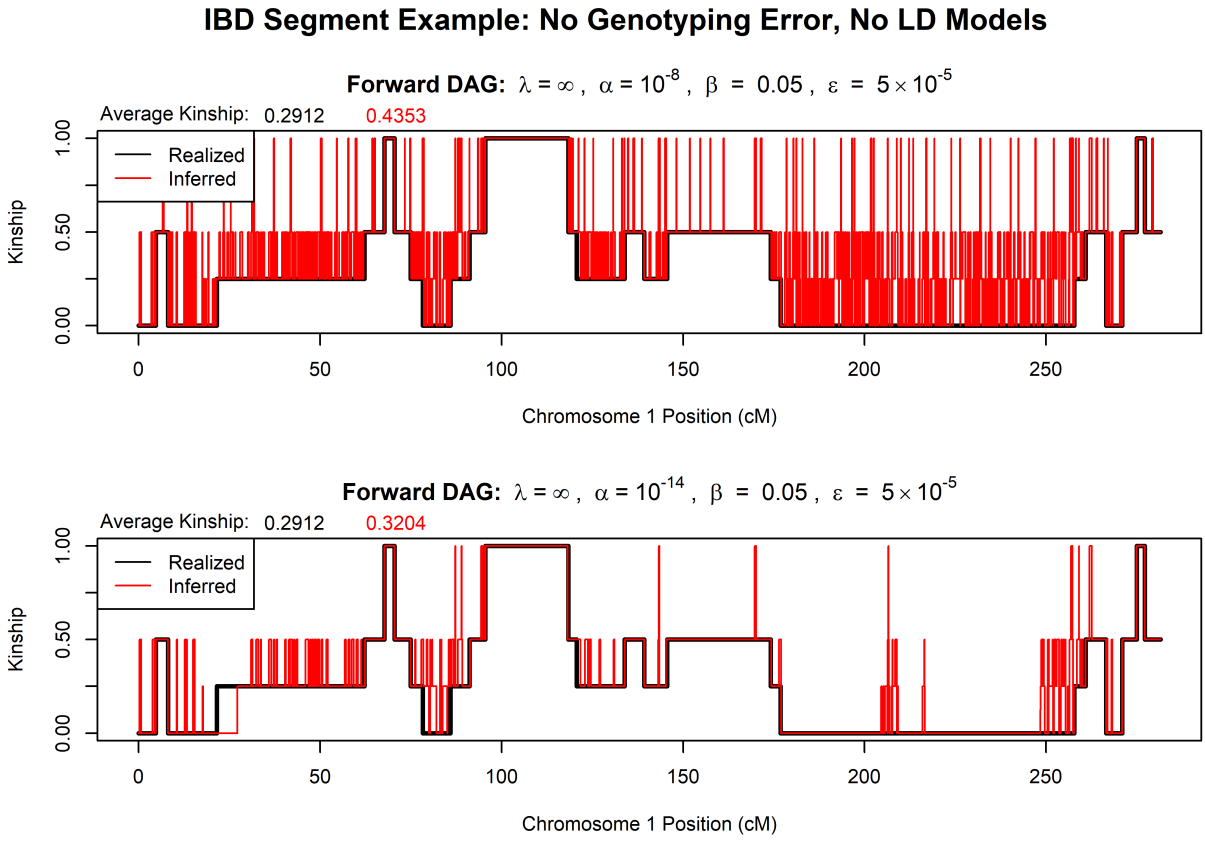

Figure 5.4: *IBD segment detection results from our Viterbi algorithm when using the collapsed DAG with no LD structure ($\lambda = \infty$). Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method. Parameters for the Viterbi algorithm are displayed for each result.*

Each simulated SNP was randomly switched to the alternate allele at a rate of 0.001 per marker, independently across all loci and chromosomes. DAG models controlling for LD in both directions using the optimal regularization parameter $\lambda = 0.5$ were again used, along with HMM parameter values $\alpha = 10^{-7}$ for the overall transition rate, $\beta = 0.05$ for the pairwise IBD probability, and $\epsilon = 10^{-5}$ for the genotyping error rate. In the "forward" direction, the inferred kinship averaged across markers is 0.2952, which is above the average realized kinship of 0.2912. We see that genotyping error induces small segments of both false positive and false negative IBD detection even though the HMM attempts to account for

this error. Recall that the edge switching method in the HMM only handles errors when maintaining an IBD state, and in the state (a)(b)(c)(d) with no IBD, no edge switching is permitted. This explains the numerous false positive segments in the 170-260 cM range where there is no realized IBD; however, controlling for LD still results in many fewer false positives than we saw in Figure 5.4. False negative segments likely occur when multiple genotyping errors occur in close proximity, so even though edge switches are allowed to correct them, transitioning to a lower IBD state produces a more likely path through the HMM. In the "backward" direction, the average inferred kinship is 0.2895, which is below the average realized kinship of 0.2912 due to kinship being underestimated in the segement around 70 cM. This segment can be recaptured at higher values of the transition rate parameter α but at the cost of more false positive segments (Supplemental Figure B.26). We see here that a number of the false negative segments occur at the same locations as in the "forward" direction, bolstering the conjecture that this is the result of clusters of genotyping errors. Beagle was no longer able to detect any meaningful IBD segments amidst the genotyping error. We attempted feeding both phased and unphased data to Beagle, toggled imputation options, and tested a variety of minimum LOD scores but were not able to achieve good results (Supplemental Figure B.32). The IBD segment detection method of Beagle version 4.1 was designed for SNP array data, not sequence data, so it is possible that better results could be obtained by removing low frequency variants and thinning the markers to provide a larger minimum spacing between them (Browning et al., 2021).

It is notable that the values for the overall transition rate parameter α that we have been using for these examples is much lower than we might expect. Recall that based upon the Chinese restaurant process underlying the HMM, we expect to obtain IBD segments between two chromosomes of approximately length $(2\alpha)^{-1}$ centimorgans. Using the parameter value $\alpha = 10^{-7}$ from the previous example should therefore result in IBD segments on the order of 5,000,000 cM, over ten thousand times longer than chromosome 1. Brown et al. (2012) found that using values of α corresponding to the lengths of the segments they expected to detect led to many false positives and poor results overall. Aligning with what we see here in

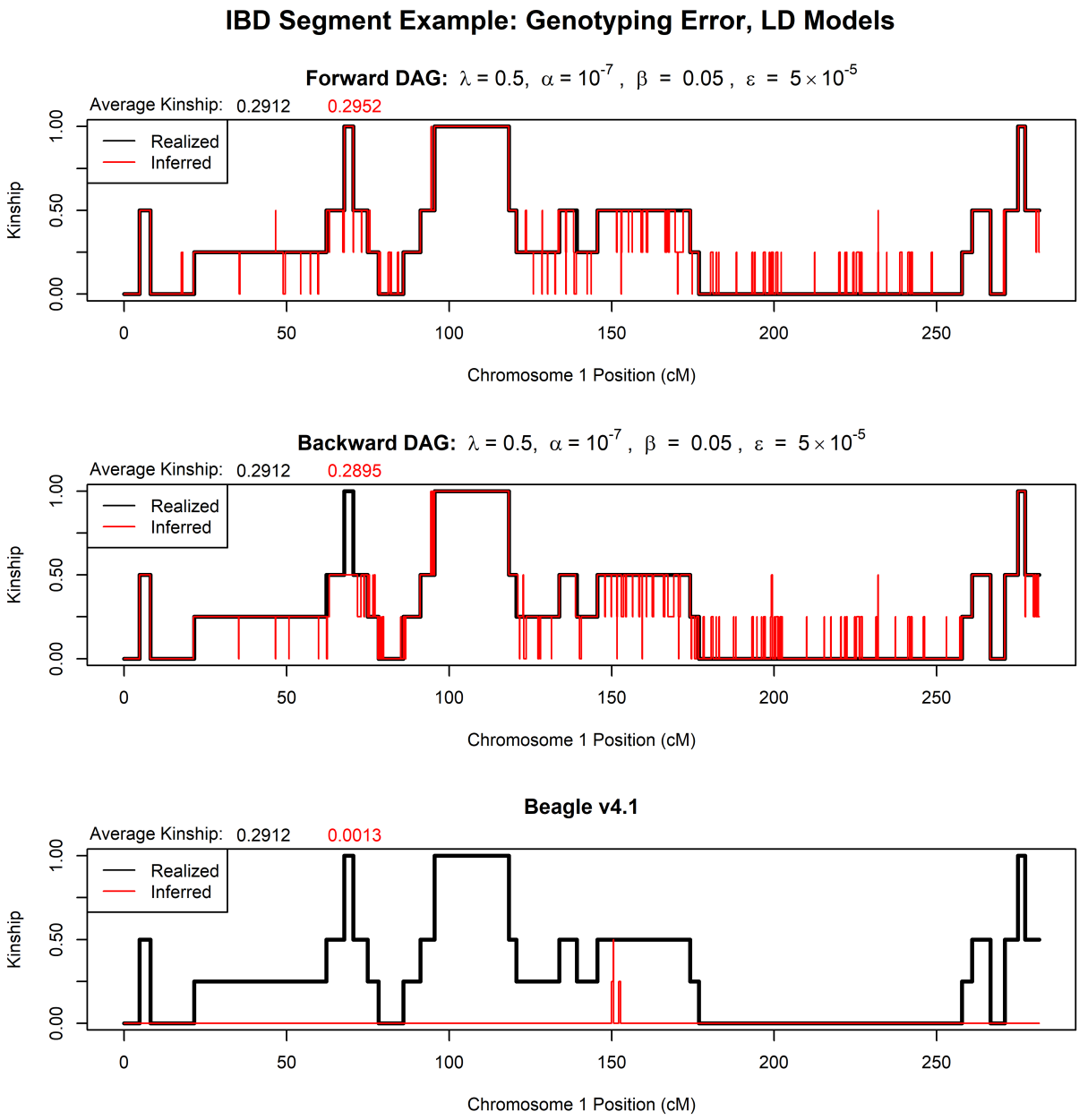

Figure 5.5: *IBD segment detection results from our Viterbi algorithm run in both directions compared to results from Beagle v4.1. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method. Parameters for the Viterbi algorithm are displayed for each result.*

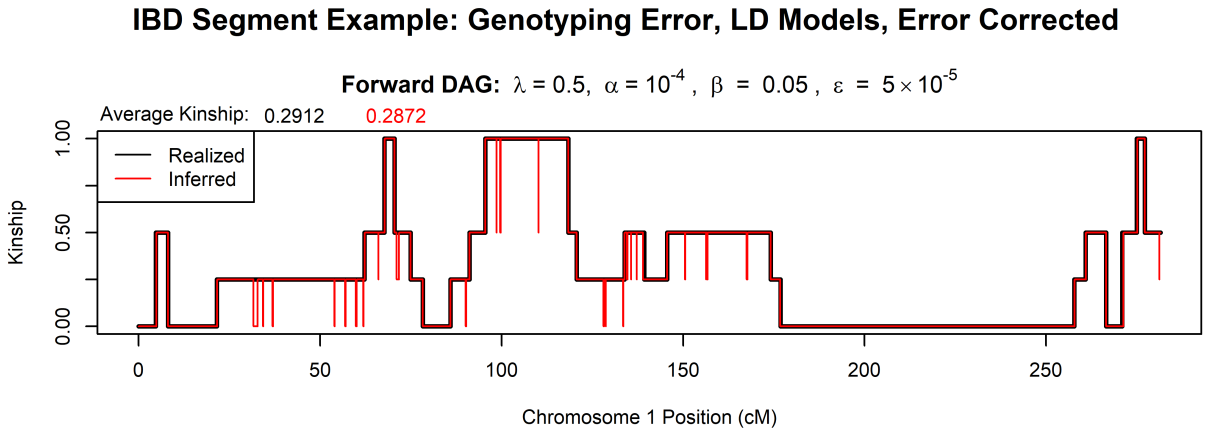

Figure 5.6: *IBD segment detection results from our Viterbi algorithm run after preprocessing by error correction. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method. Parameters for the Viterbi algorithm are displayed for each result.*

these examples, they found that their results were improved by using markedly lower values for α.

Figure 5.6 shows the results when each haplotype is preprocessed using a Viterbi algorithm to correct genotyping errors based on the DAG model. This method of preprocessing manages to completely eliminate the false positive detection of IBD segments that we saw in Figure 5.5, most notably in the region with no realized IBD in the 170-260 cM range. Due to this, the value for the α parameter was able to be increased back to the $\alpha = 10^{-4}$ value from the example with no genotyping error, allowing for more sensitive detection of IBD segments. However, there are still a number of small false negative segments. In some regions, such as from 150-175 cM, the preprocessing appears to have reduced the rate of false negatives compared to Figure 5.5, but in other regions, such as from 100-120 cM, new false negative segments have materialized. Overall, the preprocessing step improves IBD segment detection, but with the caveat that it seems greatly biased toward cleaning up false positives over false negatives.

Our next example aims to measure the performance of our IBD segment detection method by way of kinship estimation in a large sample of related individuals. As before, we used the R package `rref` to simulate inheritance patterns for pairs of individuals given founder haplotypes from the 1000 Genomes sample (Wang, 2018). Instead of a single pair, we simulated $N = 1,000$ independent pairs each of siblings and third cousins from random founders. Sibling pairs have a kinship coefficient of $1/4 = 0.25$ while third cousin pairs have a kinship coefficient of $1/256 \approx 0.0039$. Note that the realized kinship for each pair will vary about these expected values.

For each simulated sibling and third cousin pair, we estimated kinship coefficients using our HMM by collapsing the 15 IBD states as in the previous examples. To compare the performance of our method, we also computed the kinship coefficients for these same pairs using a variety of other methods. The classic genomic relationship matrix (GRM) estimator simply estimates the kinship matrix of a sample directly as the empirical correlation of genotypes. Modifications of the GRM estimator include a robust version that is less sensitive to rare variants and two-step estimator that computes weights to minimize variance (VanRaden, 2008; Wang, 2018). Also, the PLINK version 1.07 software package includes a method of moments approach to compute the probabilities of sharing 0, 1, or 2 alleles IBD by conditioning the probability of observed IBS on those IBD states (Purcell et al., 2007).

Aside from ours, none of the above methods consider marker position along the chromosome or account for LD. Since they treat markers as independent, there will be a limit to the amount of information they can extract from increasingly dense SNP panels as markers become increasingly dependent. Sverdlov (2014) introduced another GRM-based method which computes weights that minimize variance under the assumption of the presence of LD. These LD weights were computed from the 5,008 founder haplotypes and subsequently applied to the GRM estimates on the simulated sibling and cousin pairs. We also included estimates from Beagle version 4.1, which incorporates a DAG model to account for LD in its detection of IBD segments. Since Beagle creates its DAG model using the same haplotypes in which it detects IBD segments, we added only 50 of the simulated individuals to

the 2,504 founders in 20 batches so as not to allow the simulated haplotypes to overwhelm the estimation of LD structure. Finally, we also estimated kinship using our HMM but with the collapsed DAG model obtained by setting the regularization parameter at $\lambda = \infty$ that represents independent alleles with no LD. Recall that this method is analogous to that from Brown et al. (2012) and does not account for LD but does use the genetic position of markers for additional information.

To investigate the impact of LD on these methods, we estimated kinship using not only the full set of 902,606 SNP markers, but also a pruned set obtained from performing LD pruning using PLINK version 1.07. This pruning employs a sliding window in which markers are recursively discarded on the basis of having genotypic correlation $r^2 > 0.2$ (Purcell et al., 2007). This process resulted in a pruned set of 21,296 markers, a reduction of more than 97.5%. The full marker panel has a density of 3,200 SNPs per cM while the pruned panel has only 75 SNPs per cM.

Table 5.4 shows the results from the various kinship estimators. We measure performance using the root mean squared error between estimated and realized kinship normalized by mean realized kinship in a manner similar to Wang et al. (2017). That is, we define normalized root mean squared error (NRMSE) by

$$\text{NRMSE} = \frac{\sqrt{\frac{1}{N}\sum_{j=1}^{N}\left(\hat{\Phi}_j - \Phi_j\right)^2}}{\frac{1}{N}\sum_{j=1}^{N}\Phi_j},$$

where $\hat{\Phi}_j$ is the estimated kinship from each method and Φ_j is the realized kinship for each simulated pair $j = 1, \ldots, N$. All of the methods compared use marker allele frequency for estimation, but some use additional information in the form of genetic position or LD structure to improve estimation by considering inheritance jointly across the chromosome. The types of information used are marked in the table. The final line of the table gives values based on pedigree kinship, where the estimator is the constant kinship coefficient for each relationship, $1/4 = 0.25$ for siblings and $1/256 \approx 0.0039$ for third cousins. The GRM estimate with LD weights was unable to be computed on the denser 3,200 SNPs per cM

data because the windows used for calculating the LD weights were either too large to be tractable or too small to have a nonsingular correlation matrix.

	Information			Sibling Pairs		3rd Cousin Pairs	
Estimator	Allele Freq.	Genetic Position	LD	3,200/cM	75/cM	3,200/cM	75/cM
PLINK	✓			12.36	6.33	850.6	448.2
Classic GRM	✓			34.78	12.10	437.9	356.4
Robust GRM	✓			17.21	9.89	513.7	363.0
Two-Step GRM	✓			12.78	9.14	1,692.7	354.7
LD-Weighted GRM	✓		✓	—	6.74	—	173.5
No LD HMM	✓	✓		8.64	1.21	126.0	15.3
DAG HMM	✓	✓	✓	0.68	0.86	11.0	11.5
Beagle v4.1	✓	✓	✓	10.81	4.76	46.7	45.5
Pedigree				29.84	28.84	251.1	244.3

Table 5.4: *Performance of various kinship estimators as measured by 100 times the NRMSE on both the full 3,200/cM and the pruned 75/cM marker panels. The types of genetic information used for each estimator are given (Wang, 2019). The transition rate parameter $\alpha = 10^{-4}$ was used for the DAG HMM model, and for the no LD HMM model, $\alpha = 10^{-14}$ was used. The final line gives values based on pedigree kinship, 0.25 for siblings and 0.0039 for third cousins.*

The first notable result that we see from Table 5.4 is that every estimator performs substantially better on the sibling pairs than on the third cousin pairs. This is expected since our computed ratio is relative to average realized kinship, which is much lower for distant third cousins than for siblings. The PLINK and GRM estimators that use only allele frequency information generally perform notably worse than the estimates that use additional information regardless of marker density, and in the case of third cousins, they perform even worse than simply using the constant expected kinship from the pedigree as an estimate. The GRM with LD weights works better than the other GRM estimators, but still lacks compared to the HMM methods that leverage information from the genetic position

of markers. Of those methods, our DAG HMM performs best in all situations. Nearly all estimates were better on the pruned 75 per cM marker panel than on the full 3,200 per cM panel, indicating that the correlations from the high LD in the dense markers had a deleterious effect on estimator accuracy and precision. The only exception was our DAG HMM, which had slightly increased performance on the dense marker panel. As expected, when the collapsed DAG representing no LD is used in our HMM, the impact of LD is much greater. The HMM with no LD performs about an order of magnitude worse than our DAG HMM on the dense markers, but on the pruned markers, its performance is only marginally worse. Based on these results, our method appears to be the only one tested that is not only able to properly control for LD but also leverage increased information from it.

Finally, Figure 5.7 shows the distributions of estimator biases $\hat{\Phi}_j - \Phi_j$ that went into the calculations for the normalized RMSE reported in Table 5.4. The red boxplots represent the full 3,200 per cM panel while the blue ones represent the pruned 75 per cM marker panel. GRM estimators appear to be largely unbiased on the pruned markers where LD is low, but the high LD of the dense markers can introduce bias and decrease precision, particularly for sibling pairs. The PLINK estimator has a positive bias toward higher than realized kinship, and this bias seems to be exacerbated by LD. As expected from 5.4, when our HMM uses the collapsed DAG that does not control for LD, we see a positive bias from false positive detections of IBD segments. It is notable that the Beagle estimator appears to be able to leverage the higher LD in the dense marker panel to increase precision, especially in the sibling pairs, but a substantial number of outlying estimates show a strong negative bias, which could be inflating the NRMSE. This is likely due to missed detection of some IBD segments as also observed in Brown et al. (2012) and could possibly be remedied with proper tuning of some parameters beyond the recommended defaults. Our DAG HMM method shows small bias and high precision. However, in the dense marker panel, outlying estimates tend toward a positive bias, indicating false positive detection of IBD segments. This could be an artifact of our algorithm not permitting edge switching in states with no IBD, as discussed earlier.

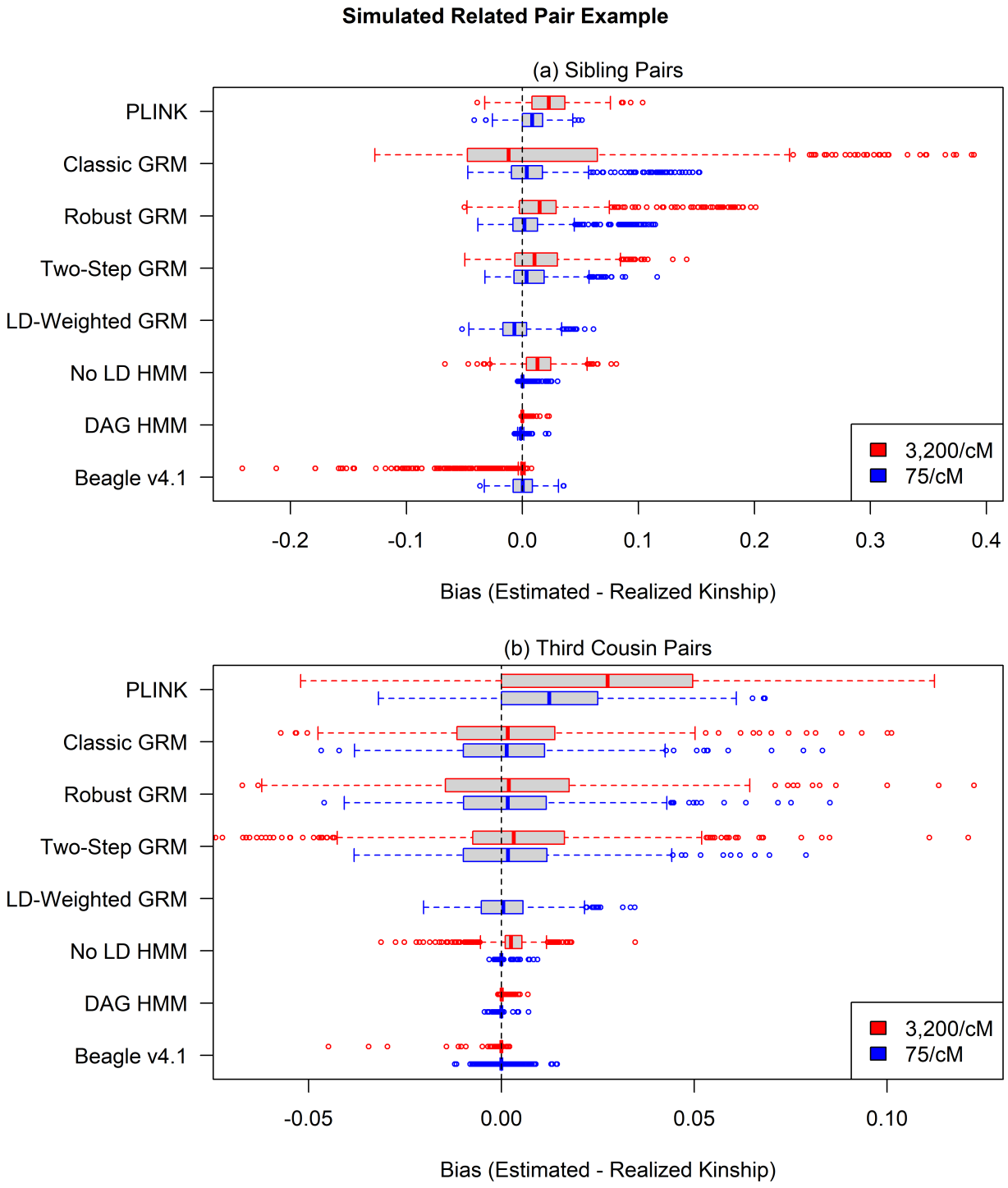

Figure 5.7: *The distributions of estimator biases $\hat{\Phi}_j - \Phi_j$ used to calculate the coefficients of variance of RMSE reported in Table 5.4. The red boxplots represent the full 3,200/cM panel while the blue ones represent the pruned 75/cM marker panel. Outliers are plotted as circles outside the whiskers of the boxplots.*

Overall, we see that our DAG HMM method for IBD segment detection performed better for estimating relatedness than not only methods based on the GRM but also other HMM methods that account for LD. In particular, our HMM was more precise than Beagle version 4.1 and did not show the same tendency for underestimating IBD. Most importantly, our method was the only one tested that was able to leverage the additional information from the dense 3,200 per cM marker panel to improve performance instead of allowing the high LD to obfuscate IBD detection.

Chapter 6

CONCLUSION

6.1 Contributions

The primary objective of this thesis is to research the use of variable-order Markov chain (VOMC) models as represented by directed acyclic graphs (DAGs) for the purpose of haplotype frequency estimation. These models have been shown to characterize the dynamic nature of linkage disequilibrium (LD) across the genome quite well, providing enough structure to capture the compex dependencies within haplotype markers while remaining parsimonious enough to be computationally tractable in even the highest density marker panels. While the team behind the Beagle software has done a tremendous amount of work in developing and applying these DAG models, we believe that there is fertile ground here for more research. With this thesis, we aspire to add statistical rigor by incorporating a likelihood-based method for DAG model selection, to increase understanding of the mathematical constraints underpinning the model and how they relate to the direction of a DAG, and to improve existing methods for detection of genome segments that are identical by descent (IBD).

In Chapters 2 and 3, we introduced the concept of DAG models for haplotype frequency estimation before developing likelihood functions and maximum likelihood estimates (MLEs) for model parameters. Our major contribution was an algorithm for DAG model selection through the use of regularized likelihood along with a technique for choosing the regularization parameter by cross-validation. We tested our method using high-density marker data from the 1000 Genomes Project and observed that it compared favorably to results from the Beagle software in terms of both model parsimony and goodness of model fit.

In Chapter 4, we conducted an investigation into how the arbitarily chosen direction in which marker data are ordered along a chromosome can affect the structure of DAG models

fit to those data. DAGs are inherently directional, and we started by characterizing the types of context-sensitive conditional independencies that can be imposed by models from opposite directions. By defining the concepts of vertex completeness and reversibility, we were able to formalize the differences in DAG directionality. We used these ideas to introduce a novel method for reconciling two DAGs fit in opposite directions into a single bidirectional model that maintains the full properties of both.

In Chapter 5, we developed a method for IBD segment detection using a hidden Markov model (HMM) across the full set of 15 IBD states for the four chromosomes of two individuals that also accounts for LD by using a DAG model in the hidden layer. Methods admitting all 15 IBD states and methods accommodating LD all previously existed, but ours is the first to do both. Using data simulated from the 1000 Genomes Project, we showed that our method provided more accurate IBD segment detection than Beagle and a host of other methods. Moreover, we observed that despite LD effectively acting as noise that obfuscates IBD, our method was able to improve upon performance when using a higher density marker panel, something no other method we tested was able to accomplish.

6.2 Future work

One major area for future work would be in improving the algorithm for model selection as described in Section 3.2. This is currently a greedy algorithm which traverses a DAG by level and attempts to minimize an l_0 regulated objective function by merging the best pairs of vertices at each level. As we discussed in that section, if the DAG model selection problem could be properly convexified and expressed in terms of an l_1 regulated objective function, then optimization methods exist that would allow us to merge vertices simultaneously across the entire genome, likely finding better optima than the greedy merge selections. It is quite possible that this would improve both the performance and the computational speed of the algorithm.

More research could also be done regarding the local performance of DAG models in potentially difficult regions such as chromosome telomeres and centromeres. We observed in

Section 3.4 that both our likelihood-based method and the Beagle software resulted in poorer model fits within these regions. However, the complexity of DAG models on a large number of markers makes it hard to diagnose any possible causes for these issues. DAG models have the capability to provide not just haplotype frequency estimates but also information about the structure of LD throughout the genome, but only if we can better understand their local behavior.

While studying the directionality and reversibility issues inherent in DAG models as discussed in Chapter 4, we also considered eschewing DAG models altogether in lieu of more general undirected graphical models. We toyed with this in Section 4.2 when we explored the parameterization of haplotype frequencies in the fashion of a Markov random field (MRF) for only four markers. Estimating graphical models on a genomic scale may not be possible using standard methods, but alternative methods have been proposed which could handle a large number of markers (Edwards, 2013). We are also aware of the concept of conditional random fields and their applications in predictive text and natural language processing (Sutton et al., 2012). These types of models may be capable of representing a larger set of context-sensitive conditional dependencies than DAG models without having to adhere to their directional constraints.

Finally, as observed in Section 4.4, our DAG reconciliation algorithm still has scalability issues that require further research and improvement. We settled on the solution of allowing new haplotype paths that must be created in one direction to accomodate haplotypes unique to the other direction to rejoin the DAG during reconciliation. However, we noted that this rejoining was likely imposing new context-sensitive conditional independencies that were not previously present in DAGs of either direction, especially when the new haplotype paths were not allowed to be long enough to provide sufficient contextual memory in the model. In particular, we were only able to extend these new haplotype paths for a small number of markers before they became computationally infeasible. An alternative method might circumvent this issue and allow for faster and more valid reconciliation.

REFERENCES

Akaike, H. (1998). Information theory and an extension of the maximum likelihood principle. In *Selected papers of Hirotugu Akaike* (pp. 199–213). Springer.

Akey, J., Jin, L., & Xiong, M. (2001). Haplotypes vs single marker linkage disequilibrium tests: what do we gain? *European Journal of Human Genetics*, *9*(4), 291–300.

Albrechtsen, A., Sand Korneliussen, T., Moltke, I., van Overseem Hansen, T., Nielsen, F. C., & Nielsen, R. (2009). Relatedness mapping and tracts of relatedness for genome-wide data in the presence of linkage disequilibrium. *Genetic epidemiology*, *33*(3), 266–274.

Aldous, D. J. (1985). Exchangeability and related topics. In *École d'été de probabilités de Saint-Flour XIII—1983* (pp. 1–198). Springer.

Balding, D. J., & Nichols, R. A. (1994). DNA profile match probability calculation: how to allow for population stratification, relatedness, database selection and single bands. *Forensic science international*, *64*(2-3), 125–140.

Baran, Y., Pasaniuc, B., Sankararaman, S., Torgerson, D. G., Gignoux, C., Eng, C., ... others (2012). Fast and accurate inference of local ancestry in Latino populations. *Bioinformatics*, *28*(10), 1359–1367.

Boutilier, C., Friedman, N., Goldszmidt, M., & Koller, D. (1996). Context-specific independence in Bayesian networks. In *Proceedings of the twelfth international conference on uncertainty in artificial intelligence* (pp. 115–123).

Broman, K. W., Murray, J. C., Sheffield, V. C., White, R. L., & Weber, J. L. (1998). Comprehensive human genetic maps: individual and sex-specific variation in recombination. *The American Journal of Human Genetics*, *63*(3), 861–869.

Brown, M. D., Glazner, C. G., Zheng, C., & Thompson, E. A. (2012). Inferring coancestry in population samples in the presence of linkage disequilibrium. *Genetics*, *190*(4),

1447–1460.

Browning, B. L., & Browning, S. R. (2011). A fast, powerful method for detecting identity by descent. *The American Journal of Human Genetics*, *88*(2), 173–182.

Browning, B. L., & Browning, S. R. (2013). Improving the accuracy and efficiency of identity-by-descent detection in population data. *Genetics*, *194*(2), 459–471.

Browning, B. L., Tian, X., Zhou, Y., & Browning, S. R. (2021). Fast two-stage phasing of large-scale sequence data. *The American Journal of Human Genetics*, *108*(10), 1880–1890.

Browning, S. R. (2006). Multilocus association mapping using variable-length Markov chains. *The American Journal of Human Genetics*, *78*(6), 903–913.

Browning, S. R. (2008). Estimation of pairwise identity by descent from dense genetic marker data in a population sample of haplotypes. *Genetics*, *178*(4), 2123–2132.

Browning, S. R., & Browning, B. L. (2007). Rapid and accurate haplotype phasing and missing-data inference for whole-genome association studies by use of localized haplotype clustering. *The American Journal of Human Genetics*, *81*(5), 1084–1097.

Browning, S. R., & Browning, B. L. (2010). High-resolution detection of identity by descent in unrelated individuals. *The American Journal of Human Genetics*, *86*(4), 526–539.

Bühlmann, P., & Wyner, A. J. (1999). Variable length Markov chains. *The Annals of Statistics*, *27*(2), 480–513.

Carroll, D. (2013). Genetic recombination. In S. Maloy & K. Hughes (Eds.), *Brenner's encyclopedia of genetics (second edition)* (Second Edition ed., p. 277-280). San Diego: Academic Press. Retrieved from `https://www.sciencedirect.com/science/article/pii/B9780123749840006276` doi: https://doi.org/10.1016/B978-0-12-374984-0.00627-6

Clifford, P., & Hammersley, J. (1971). Markov fields on finite graphs and lattices.

Donnelly, K. P. (1983). The probability that related individuals share some section of genome identical by descent. *Theoretical population biology*, *23*(1), 34–63.

Edwards, D. (2013). Modelling and visualizing fine-scale linkage disequilibrium structure.

BMC bioinformatics, *14*(1), 1.

Ewens, W. J. (1972). The sampling theory of selectively neutral alleles. *Theoretical population biology*, *3*(1), 87–112.

Excoffier, L., & Slatkin, M. (1995). Maximum-likelihood estimation of molecular haplotype frequencies in a diploid population. *Molecular biology and evolution*, *12*(5), 921–927.

Forney, G. D. (1973). The Viterbi algorithm. *Proceedings of the IEEE*, *61*(3), 268–278.

Glazner, C. (2014). *Monte carlo estimation of identity by descent in populations* (Unpublished doctoral dissertation).

Greenspan, G., & Geiger, D. (2004). High density linkage disequilibrium mapping using models of haplotype block variation. *Bioinformatics*, *20*(suppl_1), i137–i144.

Gusev, A., Lowe, J. K., Stoffel, M., Daly, M. J., Altshuler, D., Breslow, J. L., ... Pe'er, I. (2009). Whole population, genome-wide mapping of hidden relatedness. *Genome research*, *19*(2), 318–326.

Haldane, J. (1919). The combination of linkage values and the calculation of distances between the loci of linked factors. *J Genet*, *8*(29), 299–309.

Han, L., & Abney, M. (2011). Identity by descent estimation with dense genome-wide genotype data. *Genetic epidemiology*, *35*(6), 557–567.

Jacquard, A. (1970). *Structures génétiques des populations*. Institut National d'Etudes Démographiques.

Koller, D., & Friedman, N. (2009). *Probabilistic graphical models: principles and techniques*. MIT press.

Leutenegger, A.-L., Prum, B., Génin, E., Verny, C., Lemainque, A., Clerget-Darpoux, F., & Thompson, E. A. (2003). Estimation of the inbreeding coefficient through use of genomic data. *The American Journal of Human Genetics*, *73*(3), 516–523.

Liu, Z., & Lin, S. (2005). Multilocus LD measure and tagging SNP selection with generalized mutual information. *Genetic Epidemiology: The Official Publication of the International Genetic Epidemiology Society*, *29*(4), 353–364.

Matise, T. C., Chen, F., Chen, W., Francisco, M., Hansen, M., He, C., ... others (2007).

A second-generation combined linkage–physical map of the human genome. *Genome research*, *17*(12), 1783–1786.

Morris, R. W., & Kaplan, N. L. (2002). On the advantage of haplotype analysis in the presence of multiple disease susceptibility alleles. *Genetic epidemiology*, *23*(3), 221–233.

Price, A. L., Tandon, A., Patterson, N., Barnes, K. C., Rafaels, N., Ruczinski, I., ... Myers, S. (2009). Sensitive detection of chromosomal segments of distinct ancestry in admixed populations. *PLoS Genet*, *5*(6), e1000519.

Purcell, S., Neale, B., Todd-Brown, K., Thomas, L., Ferreira, M. A., Bender, D., ... others (2007). PLINK: a tool set for whole-genome association and population-based linkage analyses. *The American journal of human genetics*, *81*(3), 559–575.

Slatkin, M. (2008). Linkage disequilibrium—understanding the evolutionary past and mapping the medical future. *Nature Reviews Genetics*, *9*(6), 477–485.

Smith, A. V., Thomas, D. J., Munro, H. M., & Abecasis, G. R. (2005). Sequence features in regions of weak and strong linkage disequilibrium. *Genome research*, *15*(11), 1519–1534.

Sutton, C., McCallum, A., et al. (2012). An introduction to conditional random fields. *Foundations and Trends® in Machine Learning*, *4*(4), 267–373.

Sverdlov, S. (2014). *Functional quantitative genetics and the missing heritability problem* (Unpublished doctoral dissertation). University of Washington.

The 1000 Genomes Project Consortium. (2015). A global reference for human genetic variation. *Nature*, *526*(7571), 68–74.

Thompson, E. A. (1974). Gene identities and multiple relationships. *Biometrics*, 667–680.

Thompson, E. A. (2008). The IBD process along four chromosomes. *Theoretical population biology*, *73*(3), 369–373.

Thompson, E. A. (2009). Inferring coancestry of genome segments in populations. *Invited Proceedings of the 57th Session of the International Statistical Institute; Durban, South Africa*.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 267–288.

Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., & Knight, K. (2005). Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *67*(1), 91–108.

Tibshirani, R., Wainwright, M., & Hastie, T. (2015). *Statistical learning with sparsity: the lasso and generalizations*. Chapman and Hall/CRC.

Tibshirani, R. J., & Taylor, J. (2011). The solution path of the generalized lasso. *The annals of statistics*, *39*(3), 1335–1371.

Tishkoff, S. A., Reed, F. A., Friedlaender, F. R., Ehret, C., Ranciaro, A., Froment, A., ... others (2009). The genetic structure and history of Africans and African americans. *science*, *324*(5930), 1035–1044.

VanRaden, P. M. (2008). Efficient methods to compute genomic predictions. *Journal of dairy science*, *91*(11), 4414–4423.

Wang, B. (2018). rres: Realized relatedness estimation and simulation [Computer software manual]. Retrieved from `https://cran.r-project.org/web/packages/rres/index.html` (R package version 1.1)

Wang, B. (2019). *Realized genome sharing in random effects models for quantitative genetic traits* (Unpublished doctoral dissertation).

Wang, B., Sverdlov, S., & Thompson, E. (2017). Efficient estimation of realized kinship from SNP genotypes. *Genetics*, genetics–116.

Wikimedia Commons. (2013). *Pedigree, recombination and resulting IBD segments, schematic representation.* Retrieved from `https://commons.wikimedia.org/wiki/File:Pedigree,_recombination_and_resulting_IBD_segments,_schematic_representation.png`

Zhang, K., Akey, J. M., Wang, N., Xiong, M., Chakraborty, R., & Jin, L. (2003). Randomly distributed crossovers may generate block-like patterns of linkage disequilibrium: an

act of genetic drift. *Human genetics*, *113*(1), 51–59.

Appendix A

MRF PARAMETER CALCULATIONS

This appendix contains the calculations for the Markov random field parameter restrictions imposed by the vertex merges displayed in Table 4.1. These are computed by equating the odds ratios of the conditional distributions represented by each vertex. When vertex merges impose further downstream merges, those imposed parameter restrictions are also assumed.

$\underline{v_{3,1}^{(f)} = \{111, 211\}}$:

$$\frac{\Pr(\mathbf{X}_{1:4} = 1112)}{\Pr(\mathbf{X}_{1:4} = 1111)} = \frac{\Pr(\mathbf{X}_{1:4} = 2112)}{\Pr(\mathbf{X}_{1:4} = 2111)} \Longrightarrow \frac{\exp(\beta_4)}{1} = \frac{\exp(\beta_1 + \beta_4 + \beta_{14})}{\exp(\beta_1)}$$
$$\Longrightarrow \exp(\beta_4) = \exp(\beta_4 + \beta_{14})$$
$$\Longrightarrow \beta_{14} = 0.$$

$\underline{v_{3,2}^{(f)} = \{112, 212\}}$:

$$\frac{\Pr(\mathbf{X}_{1:4} = 1122)}{\Pr(\mathbf{X}_{1:4} = 1121)} = \frac{\Pr(\mathbf{X}_{1:4} = 2122)}{\Pr(\mathbf{X}_{1:4} = 2121)}$$
$$\Longrightarrow \frac{\exp(\beta_3 + \beta_4 + \beta_{34})}{\exp(\beta_3)} = \frac{\exp(\beta_1 + \beta_3 + \beta_4 + \beta_{13} + \beta_{14} + \beta_{34} + \beta_{134})}{\exp(\beta_1 + \beta_3 + \beta_{13})}$$
$$\Longrightarrow \exp(\beta_4 + \beta_{34}) = \exp(\beta_4 + \beta_{14} + \beta_{34} + \beta_{134})$$
$$\Longrightarrow \beta_{14} + \beta_{134} = 0.$$

$\underline{v_{3,3}^{(f)} = \{121, 122, 221, 222\}}$:

$$\frac{\Pr(\mathbf{X}_{1:4} = 1212)}{\Pr(\mathbf{X}_{1:4} = 1211)} = \frac{\Pr(\mathbf{X}_{1:4} = 1222)}{\Pr(\mathbf{X}_{1:4} = 1221)}$$

$$\implies \frac{\exp(\beta_2 + \beta_4 + \beta_{24})}{\exp(\beta_2)} = \frac{\exp(\beta_2 + \beta_3 + \beta_4 + \beta_{23} + \beta_{24} + \beta_{34} + \beta_{234})}{\exp(\beta_2 + \beta_3 + \beta_{23})}$$

$$\implies \exp(\beta_4 + \beta_{24}) = \exp(\beta_4 + \beta_{24} + \beta_{34} + \beta_{234})$$

$$\implies \beta_{34} + \beta_{234} = 0.$$

$$\frac{\Pr(\mathbf{X}_{1:4} = 1212)}{\Pr(\mathbf{X}_{1:4} = 1211)} = \frac{\Pr(\mathbf{X}_{1:4} = 2212)}{\Pr(\mathbf{X}_{1:4} = 2211)}$$

$$\implies \frac{\exp(\beta_2 + \beta_4 + \beta_{24})}{\exp(\beta_2)} = \frac{\exp(\beta_1 + \beta_2 + \beta_4 + \beta_{12} + \beta_{14} + \beta_{24} + \beta_{124})}{\exp(\beta_1 + \beta_2 + \beta_{12})}$$

$$\implies \exp(\beta_4 + \beta_{24}) = \exp(\beta_4 + \beta_{14} + \beta_{24} + \beta_{124})$$

$$\implies \beta_{14} + \beta_{124} = 0.$$

$$\frac{\Pr(\mathbf{X}_{1:4} = 2212)}{\Pr(\mathbf{X}_{1:4} = 2211)} = \frac{\Pr(\mathbf{X}_{1:4} = 2222)}{\Pr(\mathbf{X}_{1:4} = 2221)}$$

$$\implies \frac{\exp(\beta_1 + \beta_2 + \beta_4 + \beta_{12} + \beta_{14} + \beta_{24} + \beta_{124})}{\exp(\beta_1 + \beta_2 + \beta_{12})}$$

$$= \frac{\exp(\beta_1 + \beta_2 + \beta_3 + \beta_4 + \beta_{12} + \beta_{13} + \beta_{14} + \beta_{23} + \beta_{24} + \beta_{34} + \beta_{123} + \beta_{124} + \beta_{134} + \beta_{234} + \beta_{1234})}{\exp(\beta_1 + \beta_2 + \beta_3 + \beta_{12} + \beta_{13} + \beta_{23} + \beta_{123})}$$

$$\implies \exp(\beta_4 + \beta_{14} + \beta_{24} + \beta_{124}) = \exp(\beta_4 + \beta_{14} + \beta_{24} + \beta_{34} + \beta_{124} + \beta_{134} + \beta_{234} + \beta_{1234})$$

$$\implies \beta_{34} + \beta_{134} + \beta_{234} + \beta_{1234} = 0$$

$$\implies \beta_{134} + \beta_{1234} = 0, \qquad \text{since } \beta_{34} + \beta_{234} = 0.$$

$\underline{v_{2,1}^{(f)} = \{11, 21\}}$: imposes $\{111, 211\}$ and $\{112, 212\} \Longrightarrow \beta_{14} = \beta_{134} = 0$

$$\begin{aligned}
&\frac{\Pr(\mathbf{X}_{1:4} = 1121) + \Pr(\mathbf{X}_{1:4} = 1122)}{\Pr(\mathbf{X}_{1:4} = 1111) + \Pr(\mathbf{X}_{1:4} = 1112)} = \frac{\Pr(\mathbf{X}_{1:4} = 2121) + \Pr(\mathbf{X}_{1:4} = 2122)}{\Pr(\mathbf{X}_{1:4} = 2111) + \Pr(\mathbf{X}_{1:4} = 2112)} \\
&\quad \Longrightarrow \frac{\exp(\beta_3)[1 + \exp(\beta_4 + \beta_{34})]}{1 + \exp(\beta_4)} = \frac{\exp(\beta_1 + \beta_3 + \beta_{13})[1 + \exp(\beta_4 + \beta_{34})]}{\exp(\beta_1)[1 + \exp(\beta_4)]} \\
&\quad \Longrightarrow \frac{\exp(\beta_3)[1 + \exp(\beta_4 + \beta_{34})]}{1 + \exp(\beta_4)} = \frac{\exp(\beta_3 + \beta_{13})[1 + \exp(\beta_4 + \beta_{34})]}{1 + \exp(\beta_4)} \\
&\quad \Longrightarrow \beta_{13} = 0.
\end{aligned}$$

$\underline{v_{2,2}^{(f)} = \{12, 22\}}$: imposes $\{121, 221\}$ and $\{122, 222\} \Longrightarrow \beta_{14} + \beta_{124} = \beta_{134} + \beta_{1234} = 0$

$$\begin{aligned}
&\frac{\Pr(\mathbf{X}_{1:4} = 1221) + \Pr(\mathbf{X}_{1:4} = 1222)}{\Pr(\mathbf{X}_{1:4} = 1211) + \Pr(\mathbf{X}_{1:4} = 1212)} = \frac{\Pr(\mathbf{X}_{1:4} = 2221) + \Pr(\mathbf{X}_{1:4} = 2222)}{\Pr(\mathbf{X}_{1:4} = 2211) + \Pr(\mathbf{X}_{1:4} = 2212)} \\
&\quad \Longrightarrow \frac{\exp(\beta_3 + \beta_{23})[1 + \exp(\beta_4 + \beta_{24} + \beta_{34} + \beta_{234})]}{1 + \exp(\beta_4 + \beta_{24})} \\
&\qquad = \frac{\exp(\beta_1 + \beta_2 + \beta_3 + \beta_{12} + \beta_{13} + \beta_{23} + \beta_{123})[1 + \exp(\beta_4 + \beta_{24} + \beta_{34} + \beta_{234})]}{\exp(\beta_1 + \beta_2 + \beta_{12})[1 + \exp(\beta_4 + \beta_{24})]} \\
&\quad \Longrightarrow \frac{\exp(\beta_3 + \beta_{23})[1 + \exp(\beta_4 + \beta_{24} + \beta_{34} + \beta_{234})]}{1 + \exp(\beta_4 + \beta_{24})} \\
&\qquad = \frac{\exp(\beta_3 + \beta_{13} + \beta_{23} + \beta_{123})[1 + \exp(\beta_4 + \beta_{24} + \beta_{34} + \beta_{234})]}{1 + \exp(\beta_4 + \beta_{24})} \\
&\quad \Longrightarrow \beta_{13} + \beta_{123} = 0.
\end{aligned}$$

$\underline{v_{3,1}^{(b)} = \{111, 112, 121, 122\}}$:

$$\begin{aligned}
\frac{\Pr(\mathbf{X}_{1:4} = 2111)}{\Pr(\mathbf{X}_{1:4} = 1111)} = \frac{\Pr(\mathbf{X}_{1:4} = 2112)}{\Pr(\mathbf{X}_{1:4} = 1112)} &\Longrightarrow \frac{\exp(\beta_1)}{1} = \frac{\exp(\beta_1 + \beta_4 + \beta_{14})}{\exp(\beta_4)} \\
&\Longrightarrow \exp(\beta_1) = \exp(\beta_1 + \beta_{14}) \\
&\Longrightarrow \beta_{14} = 0.
\end{aligned}$$

$$\frac{\Pr(\mathbf{X}_{1:4}=2111)}{\Pr(\mathbf{X}_{1:4}=1111)} = \frac{\Pr(\mathbf{X}_{1:4}=2121)}{\Pr(\mathbf{X}_{1:4}=1121)} \Longrightarrow \frac{\exp(\beta_1)}{1} = \frac{\exp(\beta_1+\beta_3+\beta_{13})}{\exp(\beta_3)}$$
$$\Longrightarrow \exp(\beta_1) = \exp(\beta_1+\beta_{13})$$
$$\Longrightarrow \beta_{13} = 0.$$

$$\frac{\Pr(\mathbf{X}_{1:4}=2121)}{\Pr(\mathbf{X}_{1:4}=1121)} = \frac{\Pr(\mathbf{X}_{1:4}=2122)}{\Pr(\mathbf{X}_{1:4}=1122)}$$
$$\Longrightarrow \frac{\exp(\beta_1+\beta_3+\beta_{13})}{\exp(\beta_3)} = \frac{\exp(\beta_1+\beta_3+\beta_4+\beta_{13}+\beta_{14}+\beta_{34}+\beta_{134})}{\exp(\beta_3+\beta_4+\beta_{34})}$$
$$\Longrightarrow \exp(\beta_1+\beta_{13}) = \exp(\beta_1+\beta_{13}+\beta_{134})$$
$$\Longrightarrow \beta_{134} = 0, \qquad \text{since } \beta_{14}=0.$$

$\underline{v_{3,2}^{(b)} = \{211, 221\}}$:

$$\frac{\Pr(\mathbf{X}_{1:4}=2211)}{\Pr(\mathbf{X}_{1:4}=1211)} = \frac{\Pr(\mathbf{X}_{1:4}=2221)}{\Pr(\mathbf{X}_{1:4}=1221)}$$
$$\Longrightarrow \frac{\exp(\beta_1+\beta_2+\beta_{12})}{\exp(\beta_2)} = \frac{\exp(\beta_1+\beta_2+\beta_3+\beta_{12}+\beta_{13}+\beta_{23}+\beta_{123})}{\exp(\beta_2+\beta_3+\beta_{23})}$$
$$\Longrightarrow \exp(\beta_1+\beta_{12}) = \exp(\beta_1+\beta_{12}+\beta_{13}+\beta_{123})$$
$$\Longrightarrow \beta_{13}+\beta_{123} = 0.$$

$\underline{v_{3,3}^{(b)} = \{212, 222\}}$:

$$\frac{\Pr(\mathbf{X}_{1:4} = 2212)}{\Pr(\mathbf{X}_{1:4} = 1212)} = \frac{\Pr(\mathbf{X}_{1:4} = 2222)}{\Pr(\mathbf{X}_{1:4} = 1222)}$$

$$\implies \frac{\exp(\beta_1 + \beta_2 + \beta_4 + \beta_{12} + \beta_{14} + \beta_{24} + \beta_{124})}{\exp(\beta_2 + \beta_4 + \beta_{24})}$$

$$= \frac{\exp(\beta_1 + \beta_2 + \beta_3 + \beta_4 + \beta_{12} + \beta_{13} + \beta_{14} + \beta_{23} + \beta_{24} + \beta_{34} + \beta_{123} + \beta_{124} + \beta_{134} + \beta_{234} + \beta_{1234})}{\exp(\beta_2 + \beta_3 + \beta_4 + \beta_{23} + \beta_{24} + \beta_{34} + \beta_{234})}$$

$$\implies \exp(\beta_1 + \beta_{12} + \beta_{14} + \beta_{124}) = \exp(\beta_1 + \beta_{12} + \beta_{13} + \beta_{14} + \beta_{123} + \beta_{124} + \beta_{134} + \beta_{1234})$$

$$\implies \beta_{13} + \beta_{123} + \beta_{134} + \beta_{1234} = 0.$$

$\underline{v_{2,1}^{(b)} = \{11, 21\}}$: imposes $\{111, 121\}$ and $\{211, 221\} \implies \beta_{13} = \beta_{123} = 0$

$$\frac{\Pr(\mathbf{X}_{1:4} = 2111) + \Pr(\mathbf{X}_{1:4} = 2211)}{\Pr(\mathbf{X}_{1:4} = 1111) + \Pr(\mathbf{X}_{1:4} = 1211)} = \frac{\Pr(\mathbf{X}_{1:4} = 2121) + \Pr(\mathbf{X}_{1:4} = 2221)}{\Pr(\mathbf{X}_{1:4} = 1121) + \Pr(\mathbf{X}_{1:4} = 1221)}$$

$$\implies \frac{\exp(\beta_1)[1 + \exp(\beta_2 + \beta_{12})]}{1 + \exp(\beta_2)} = \frac{\exp(\beta_1 + \beta_3)[1 + \exp(\beta_2 + \beta_{12} + \beta_{23})]}{\exp(\beta_3)[1 + \exp(\beta_2 + \beta_{23})]}$$

$$\implies \frac{\exp(\beta_1)[1 + \exp(\beta_2 + \beta_{12})]}{1 + \exp(\beta_2)} = \frac{\exp(\beta_1)[1 + \exp(\beta_2 + \beta_{12} + \beta_{23})]}{1 + \exp(\beta_2 + \beta_{23})}$$

$$\implies \beta_{23} = 0.$$

$\underline{v_{2,2}^{(b)} = \{12, 22\}}$: imposes $\{112, 122\}$ and $\{212, 222\} \Longrightarrow \beta_{13} + \beta_{134} = \beta_{123} + \beta_{1234} = 0$

$$\frac{\Pr(\mathbf{X}_{1:4} = 2112) + \Pr(\mathbf{X}_{1:4} = 2212)}{\Pr(\mathbf{X}_{1:4} = 1112) + \Pr(\mathbf{X}_{1:4} = 1212)} = \frac{\Pr(\mathbf{X}_{1:4} = 2122) + \Pr(\mathbf{X}_{1:4} = 2222)}{\Pr(\mathbf{X}_{1:4} = 1122) + \Pr(\mathbf{X}_{1:4} = 1222)}$$

$$\Longrightarrow \frac{\exp(\beta_1 + \beta_4 + \beta_{14})[1 + \exp(\beta_2 + \beta_{12} + \beta_{24} + \beta_{124})]}{\exp(\beta_4)[1 + \exp(\beta_2 + \beta_{24})]}$$

$$= \frac{\exp(\beta_1 + \beta_3 + \beta_4 + \beta_{14} + \beta_{34})[1 + \exp(\beta_2 + \beta_{12} + \beta_{23} + \beta_{24} + \beta_{124} + \beta_{234})]}{\exp(\beta_3 + \beta_4 + \beta_{34})[1 + \exp(\beta_2 + \beta_{23} + \beta_{24} + \beta_{234})]}$$

$$\Longrightarrow \frac{\exp(\beta_1 + \beta_{14})[1 + \exp(\beta_2 + \beta_{12} + \beta_{24} + \beta_{124})]}{1 + \exp(\beta_2 + \beta_{24})}$$

$$= \frac{\exp(\beta_1 + \beta_{14})[1 + \exp(\beta_2 + \beta_{12} + \beta_{23} + \beta_{24} + \beta_{124} + \beta_{234})]}{1 + \exp(\beta_2 + \beta_{23} + \beta_{24} + \beta_{234})}$$

$$\Longrightarrow \beta_{23} + \beta_{234} = 0.$$

Appendix B

IBD FIGURES

This appendix provides additional IBD segment detection results from our likelihood-based Viterbi algorithm as well as from the Beagle version 4.1 software using haplotypes simulated from the siblings in Figure 5.2. Supplemental Figures B.1-B.12 give results from our method run in the "forward" and "backward" directions at various values of α, β, and ϵ when no genotyping error was introduced. Supplemental Figures B.13-B.18 give results from our method run using the collapsed DAG with no LD structure ($\lambda = \infty$) at various values of α, β, and ϵ when no genotyping error was introduced. Supplemental Figures B.19-B.30 give results from our method run in the "forward" and "backward" directions at various values of α, β, and ϵ when genotyping error was introduced at a rate of 0.001 per marker. Finally, Supplemental Figures B.31-B.32 give results from Beagle v4.1 at various values of minimum LOD score when genotyping error was introduced at a rate of 0.001 per marker.

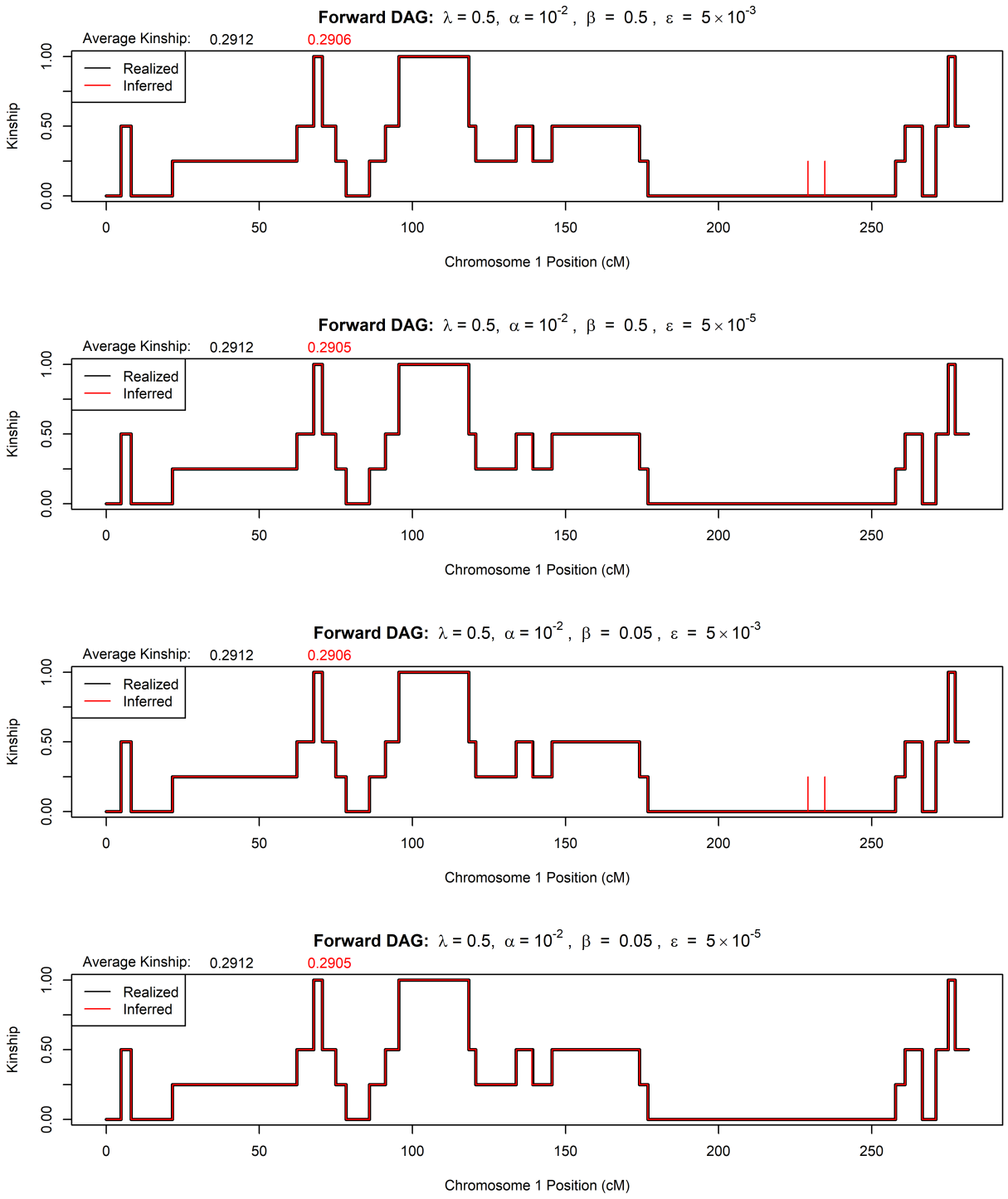

Figure B.1: *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at $\alpha = 10^{-2}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

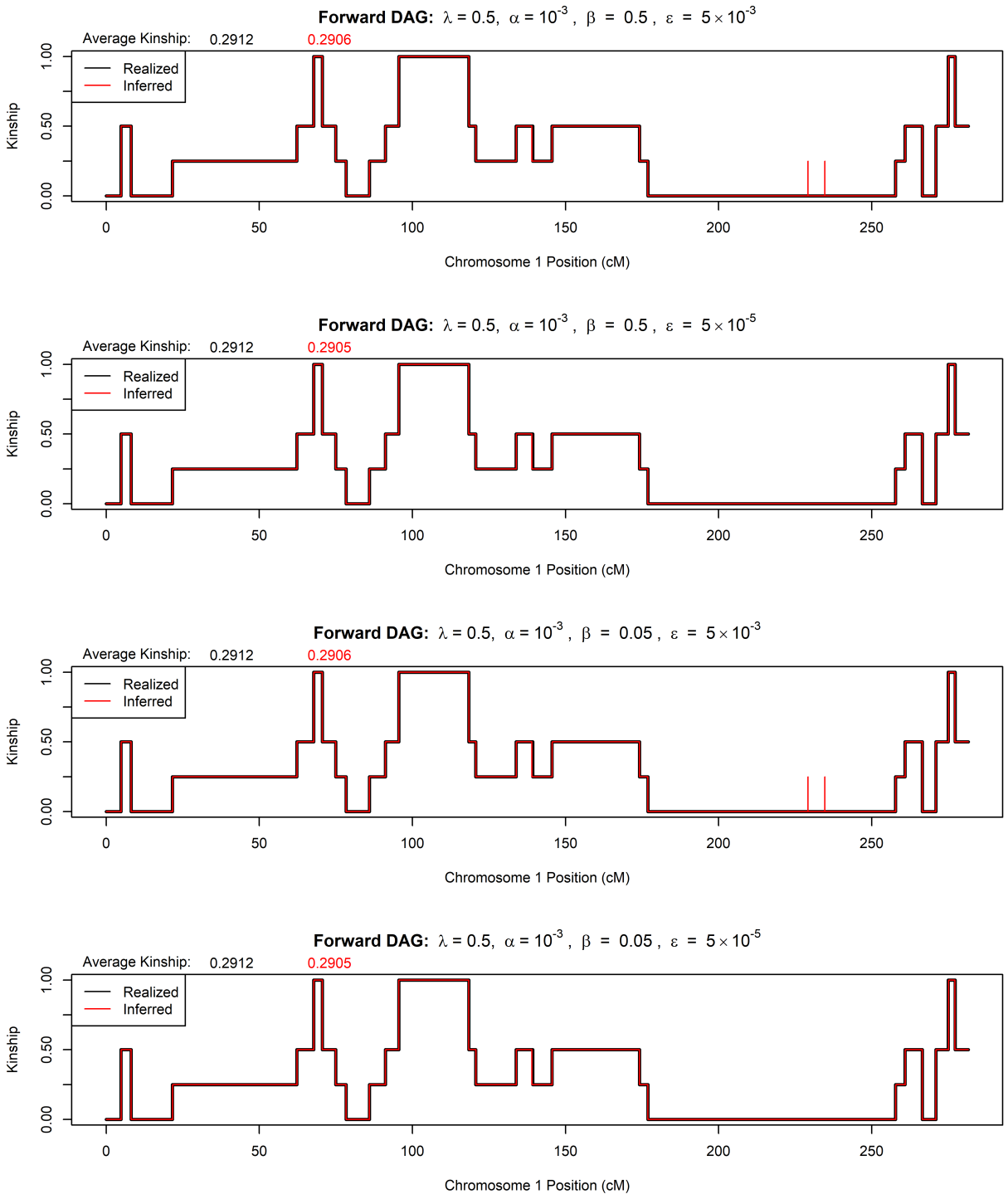

Figure B.2: *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at* $\alpha = 10^{-3}$ *and various values of* β *and* ϵ*. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal* λ *regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

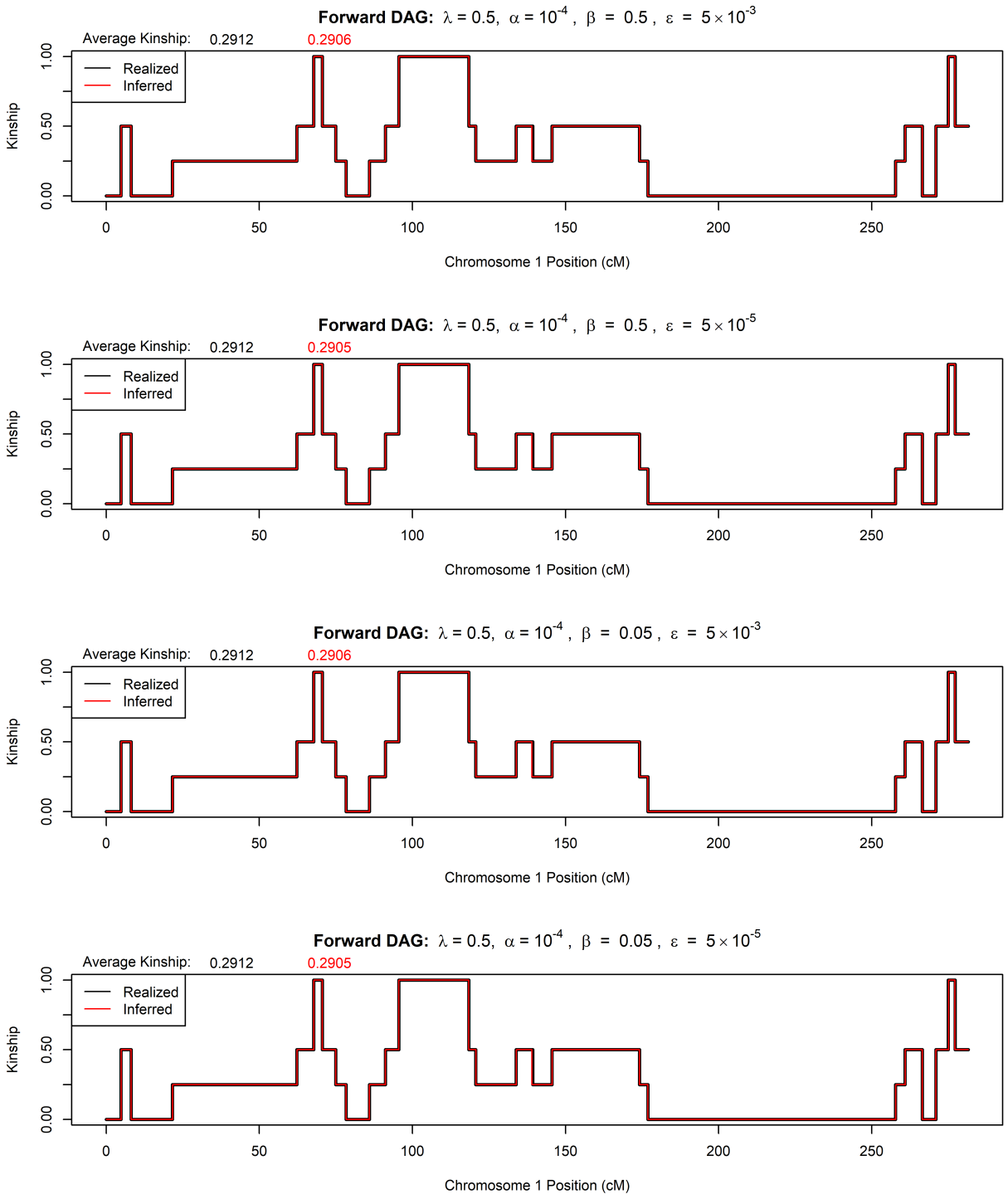

Figure B.3: *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at* $\alpha = 10^{-4}$ *and various values of* β *and* ϵ*. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal* λ *regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

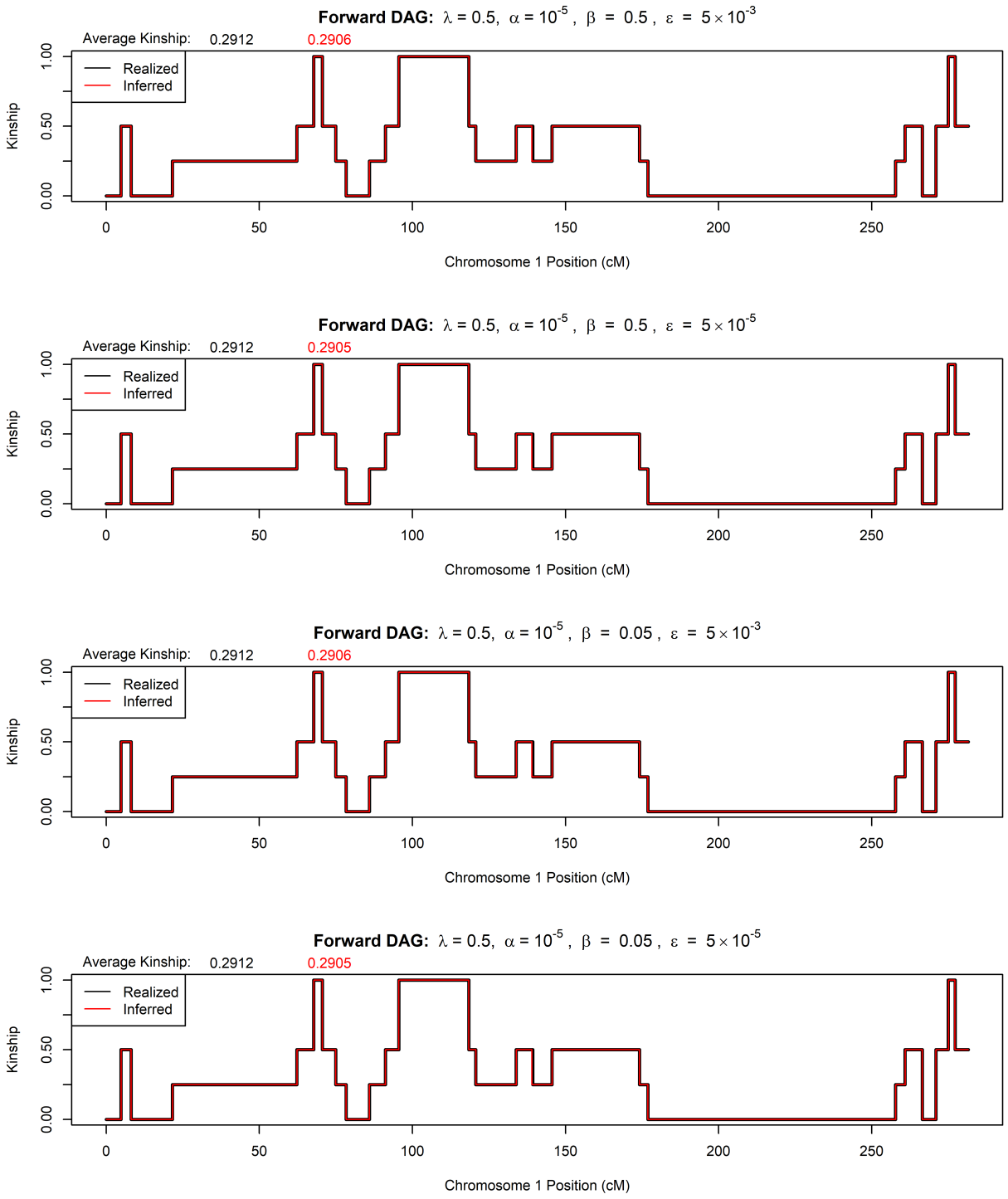

Figure B.4: *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at* $\alpha = 10^{-5}$ *and various values of* β *and* ϵ*. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal* λ *regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

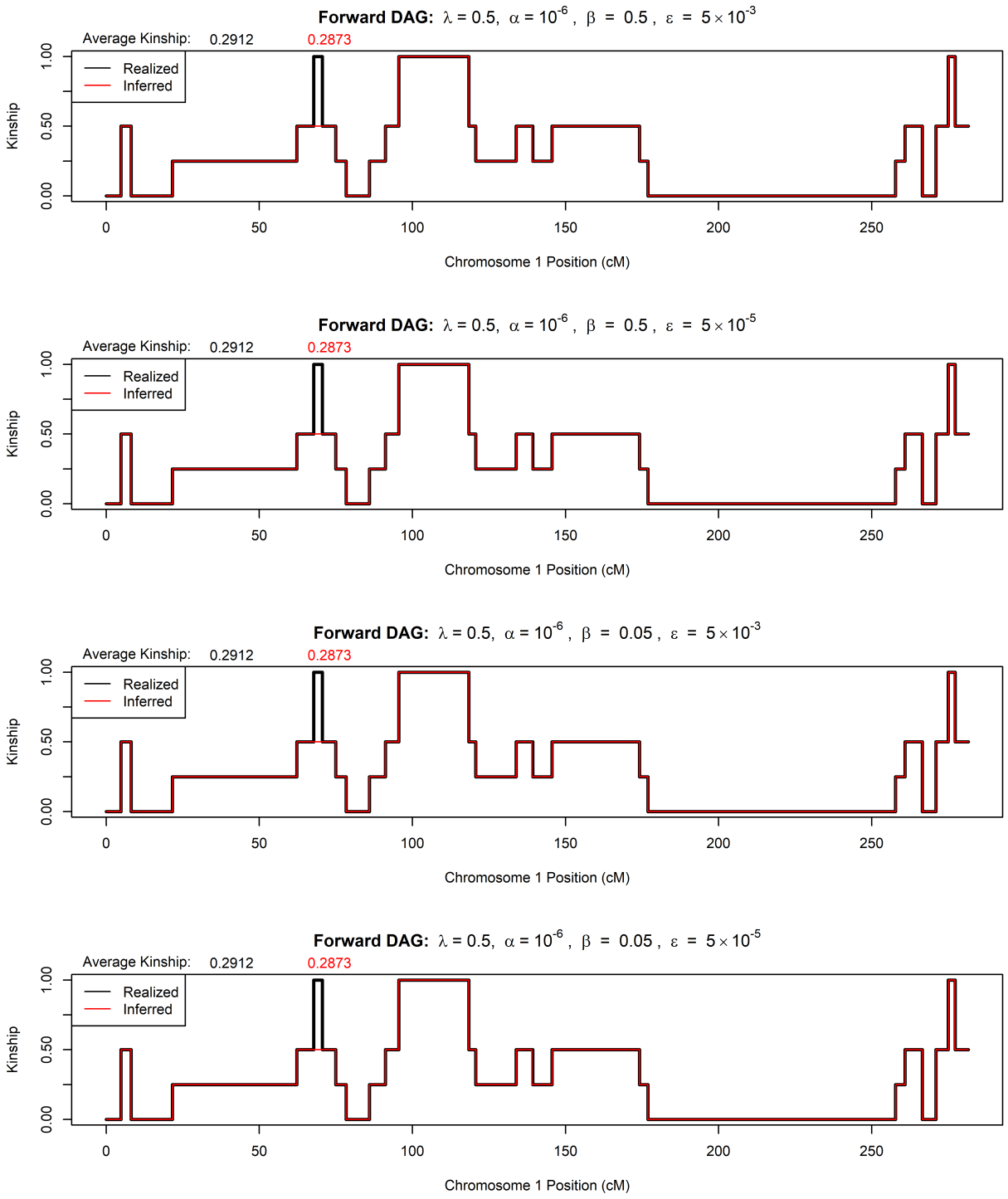

Figure B.5: *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at* $\alpha = 10^{-6}$ *and various values of* β *and* ϵ*. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal* λ *regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

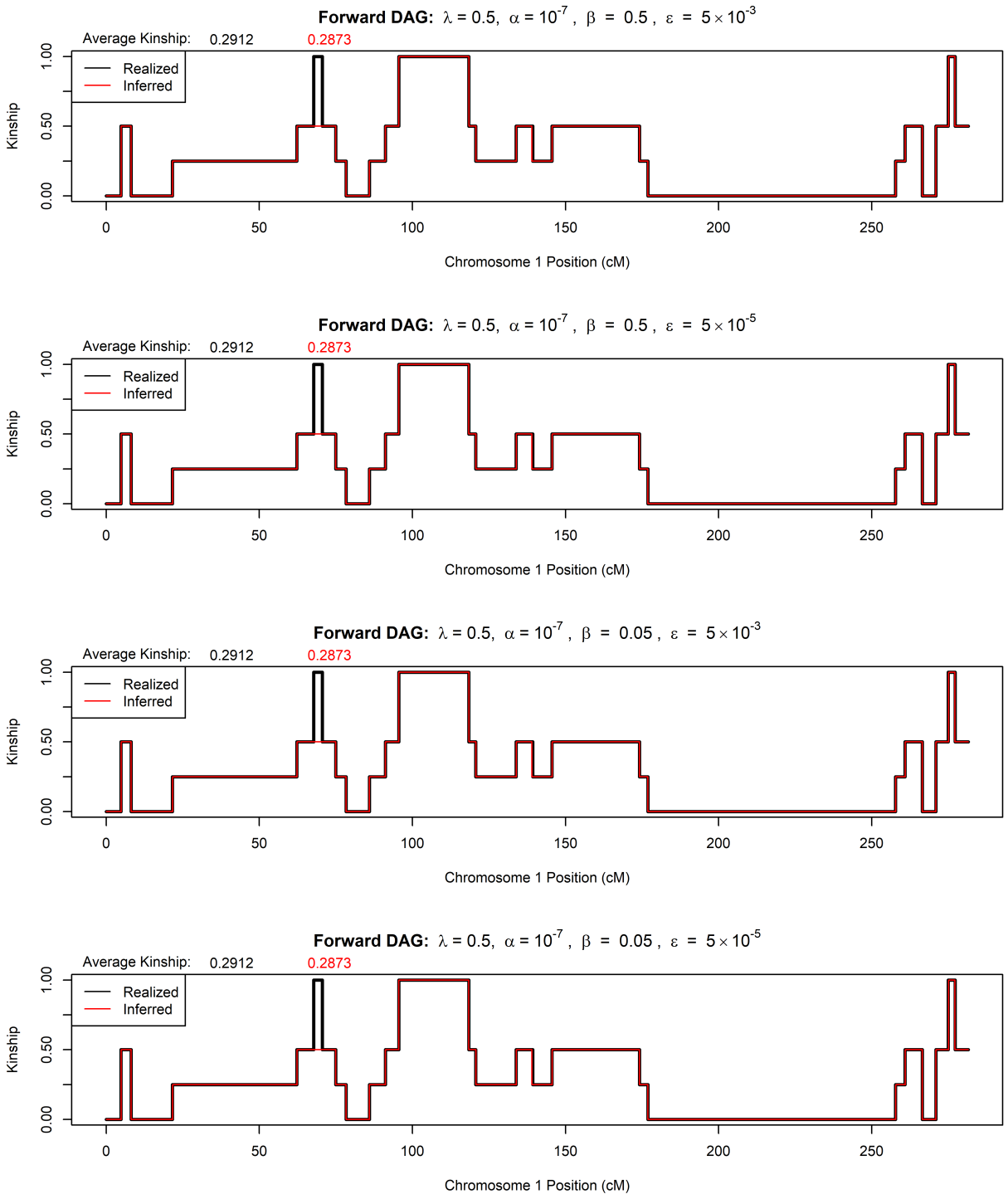

Figure B.6: *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at* $\alpha = 10^{-7}$ *and various values of* β *and* ϵ*. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal* λ *regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

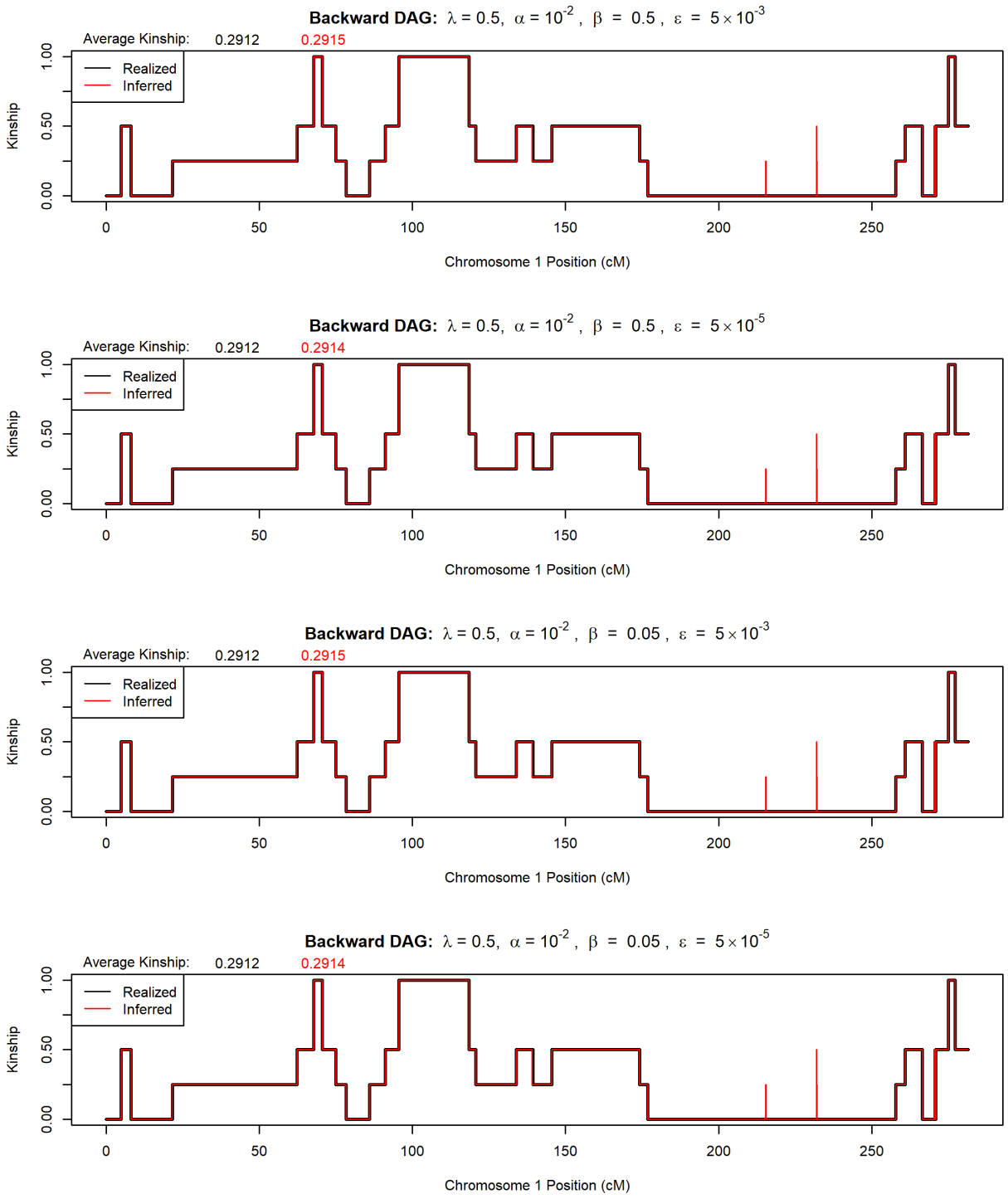

Figure B.7: *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at* $\alpha = 10^{-2}$ *and various values of* β *and* ϵ*. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal* λ *regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

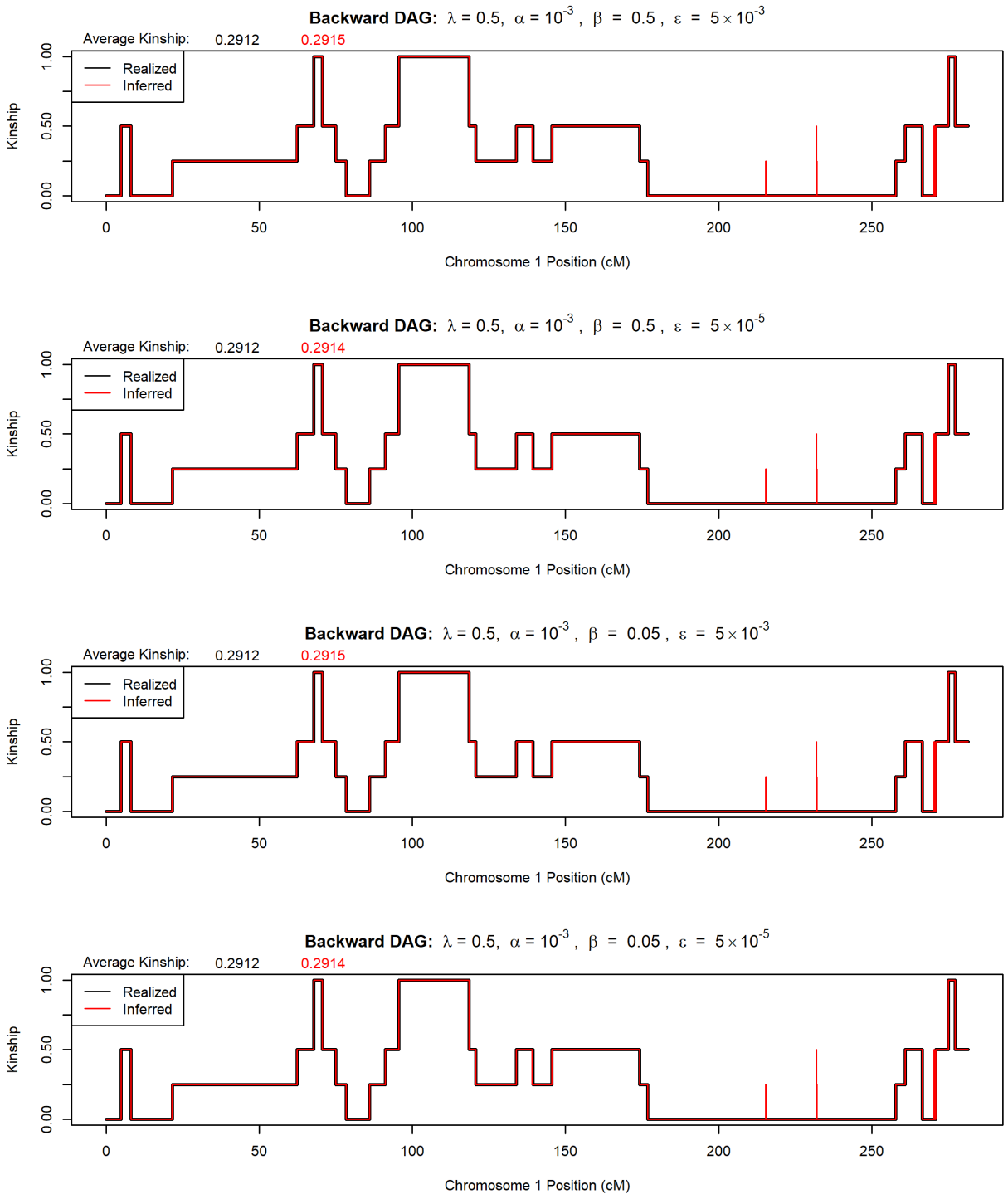

Figure B.8: *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-3}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

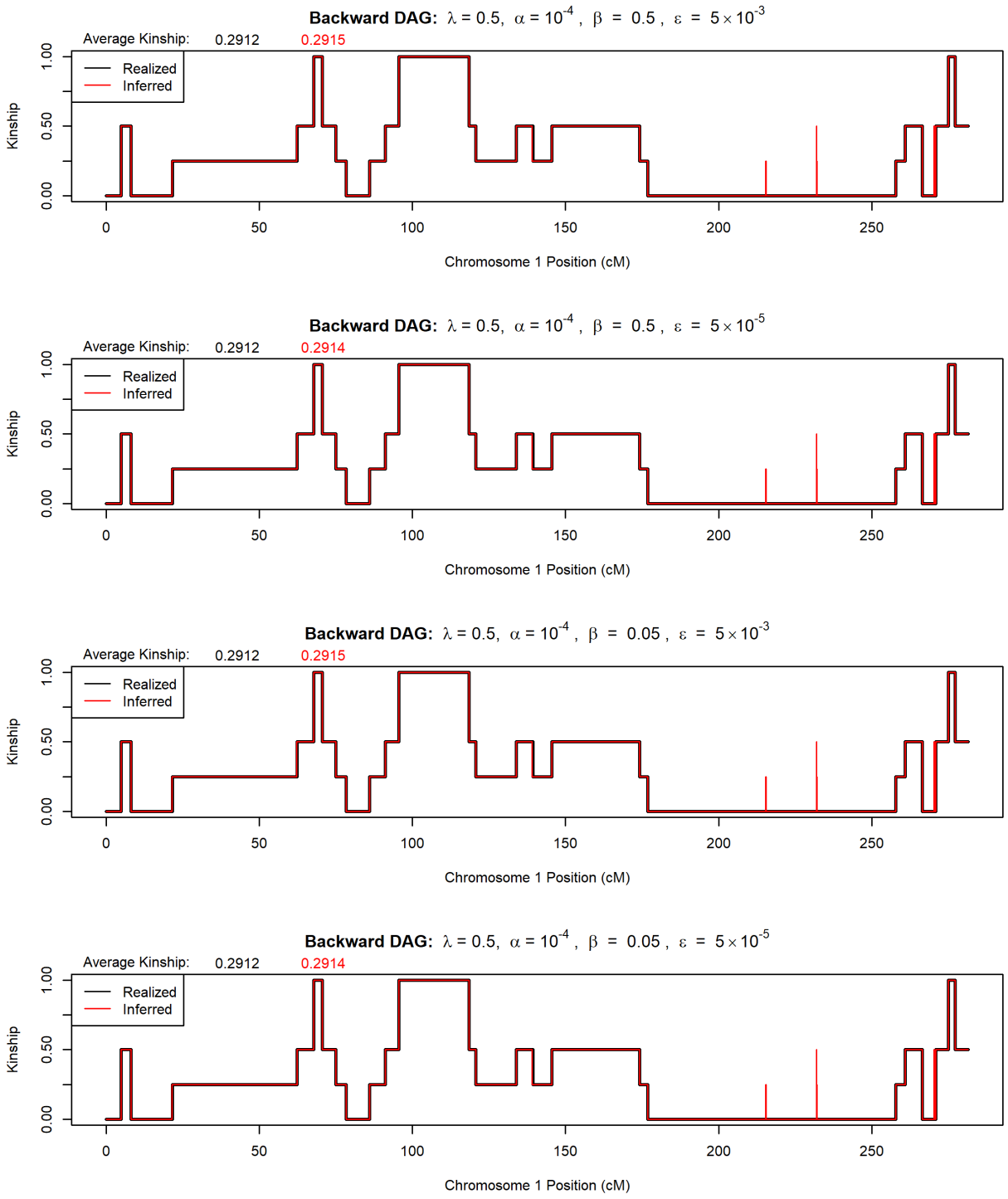

Figure B.9: *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at* $\alpha = 10^{-4}$ *and various values of* β *and* ϵ*. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal* λ *regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

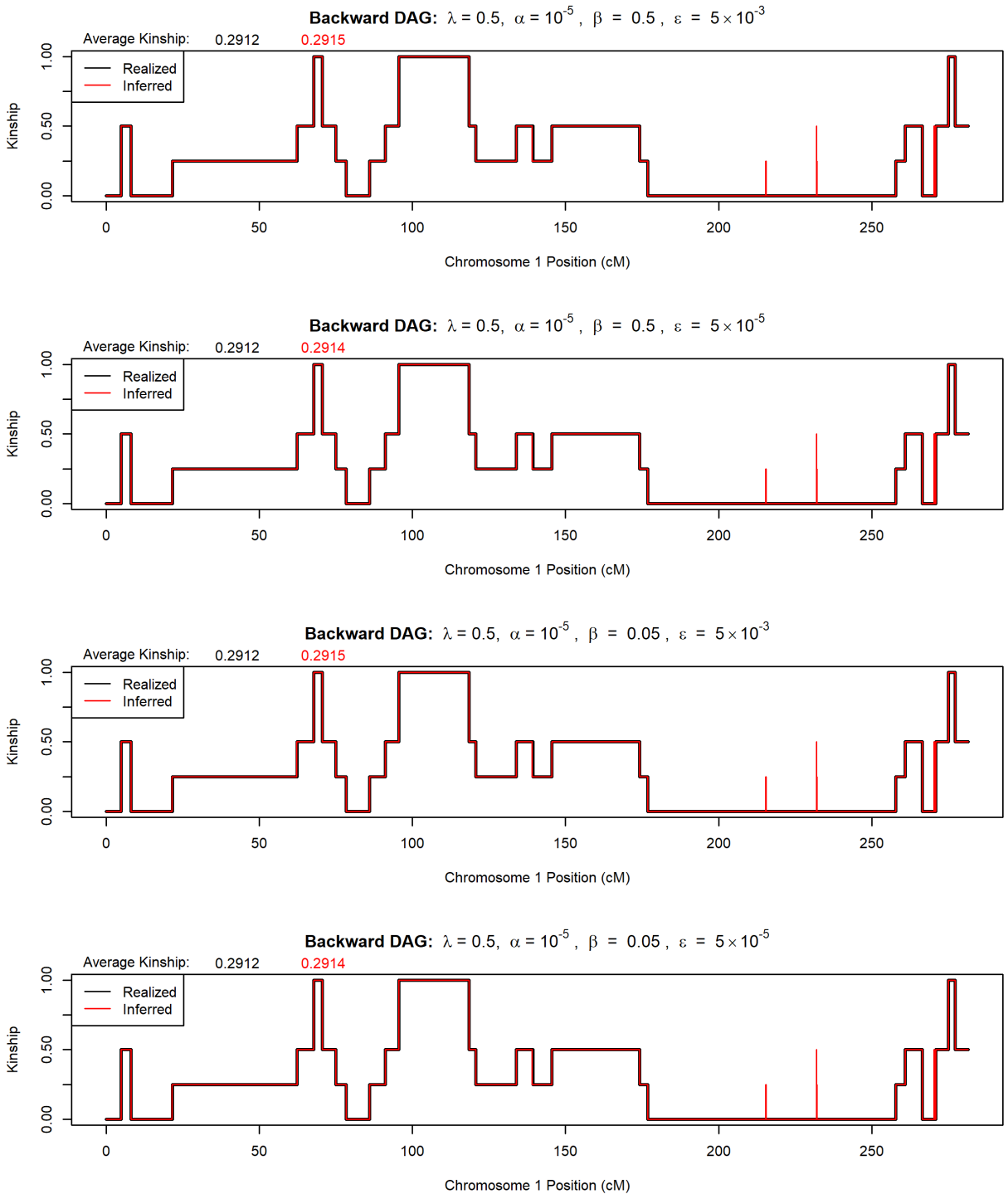

Figure B.10: *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at* $\alpha = 10^{-5}$ *and various values of* β *and* ϵ*. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal* λ *regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

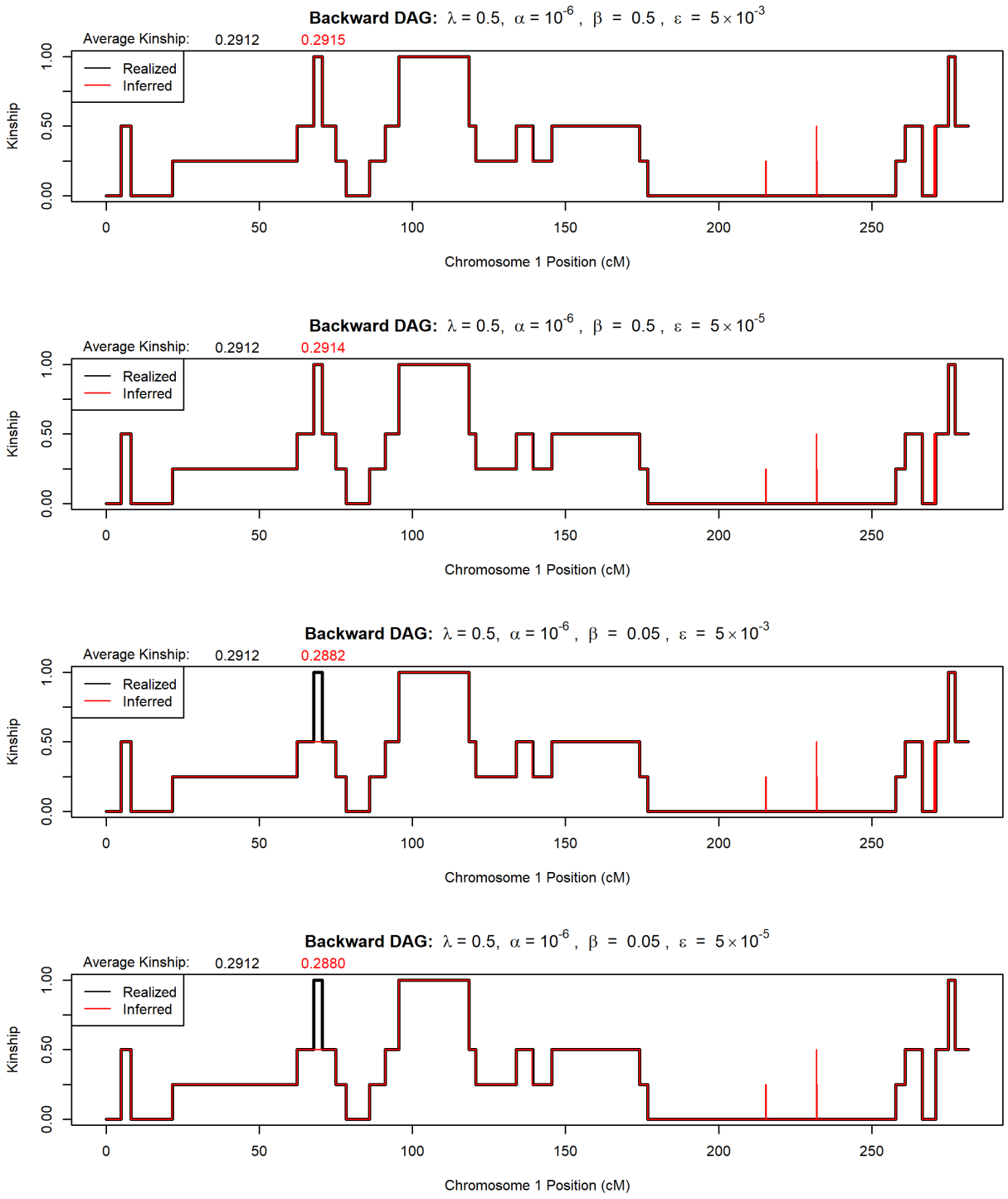

Figure B.11: *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-6}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

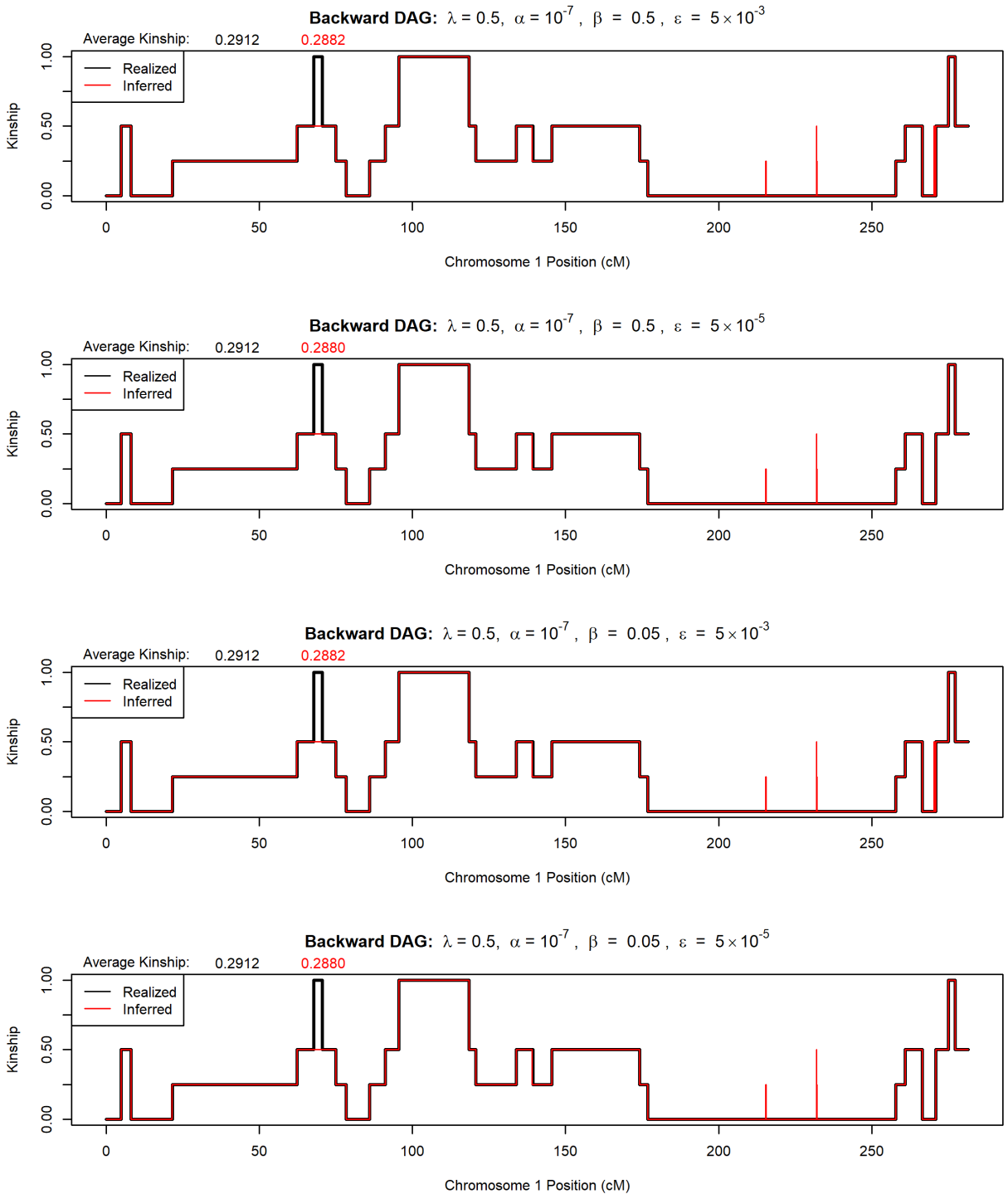

Figure B.12: *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-7}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

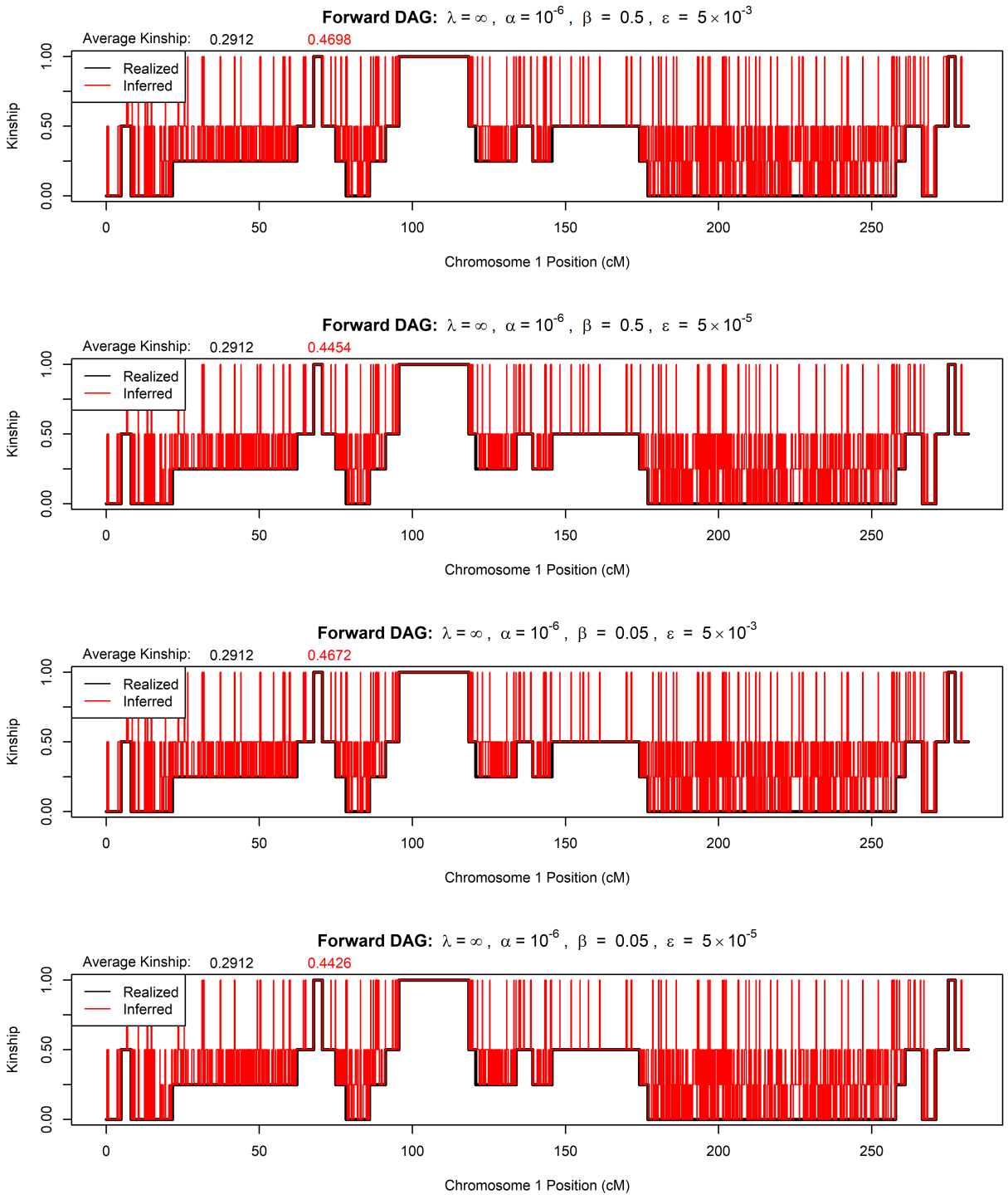

Figure B.13: *IBD segment detection results from our Viterbi algorithm when using a completely collapsed DAG with no LD structure ($\lambda = \infty$) at $\alpha = 10^{-6}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

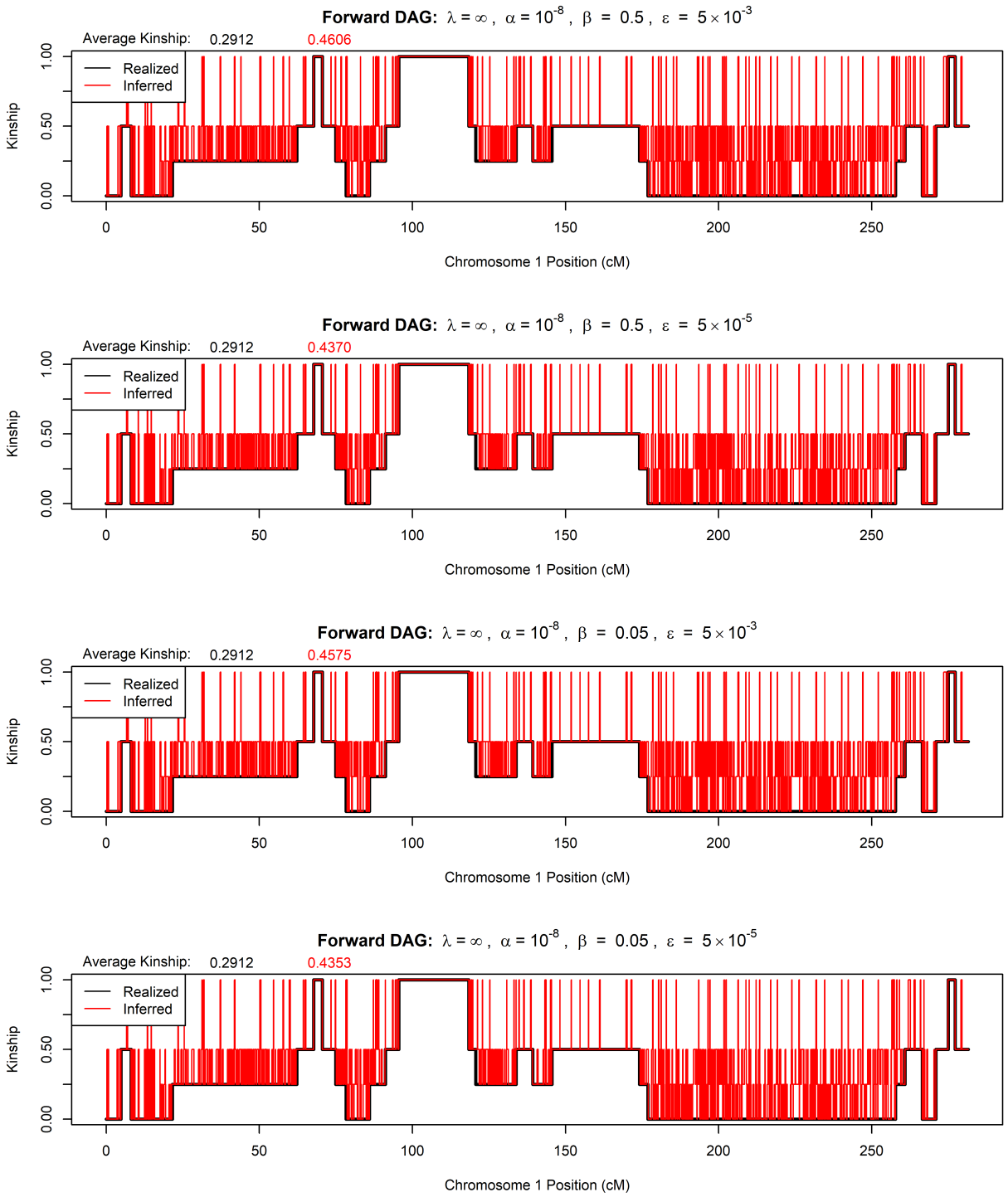

Figure B.14: *IBD segment detection results from our Viterbi algorithm when using a completely collapsed DAG with no LD structure ($\lambda = \infty$) at $\alpha = 10^{-8}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

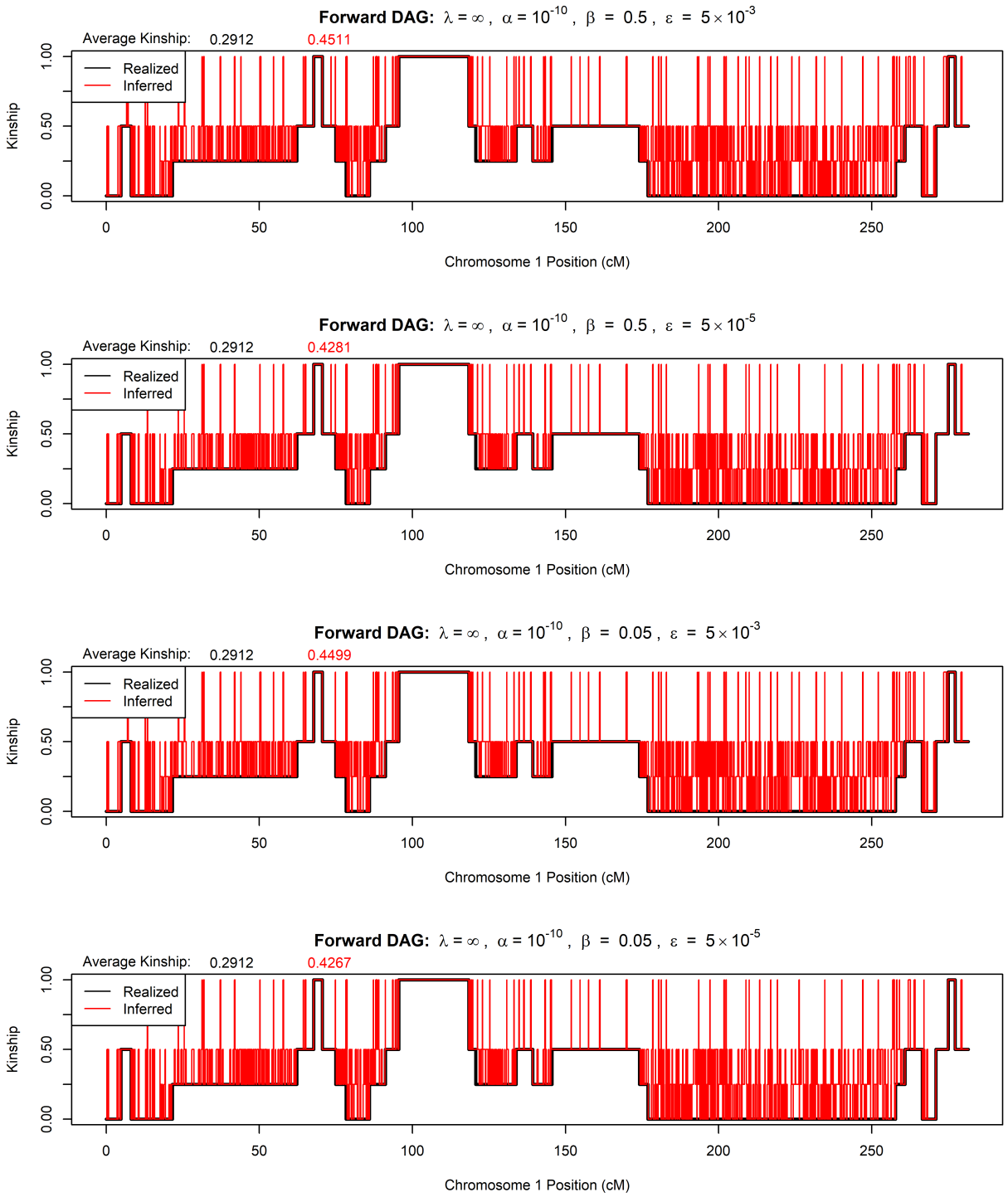

Figure B.15: *IBD segment detection results from our Viterbi algorithm when using a completely collapsed DAG with no LD structure ($\lambda = \infty$) at $\alpha = 10^{-10}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

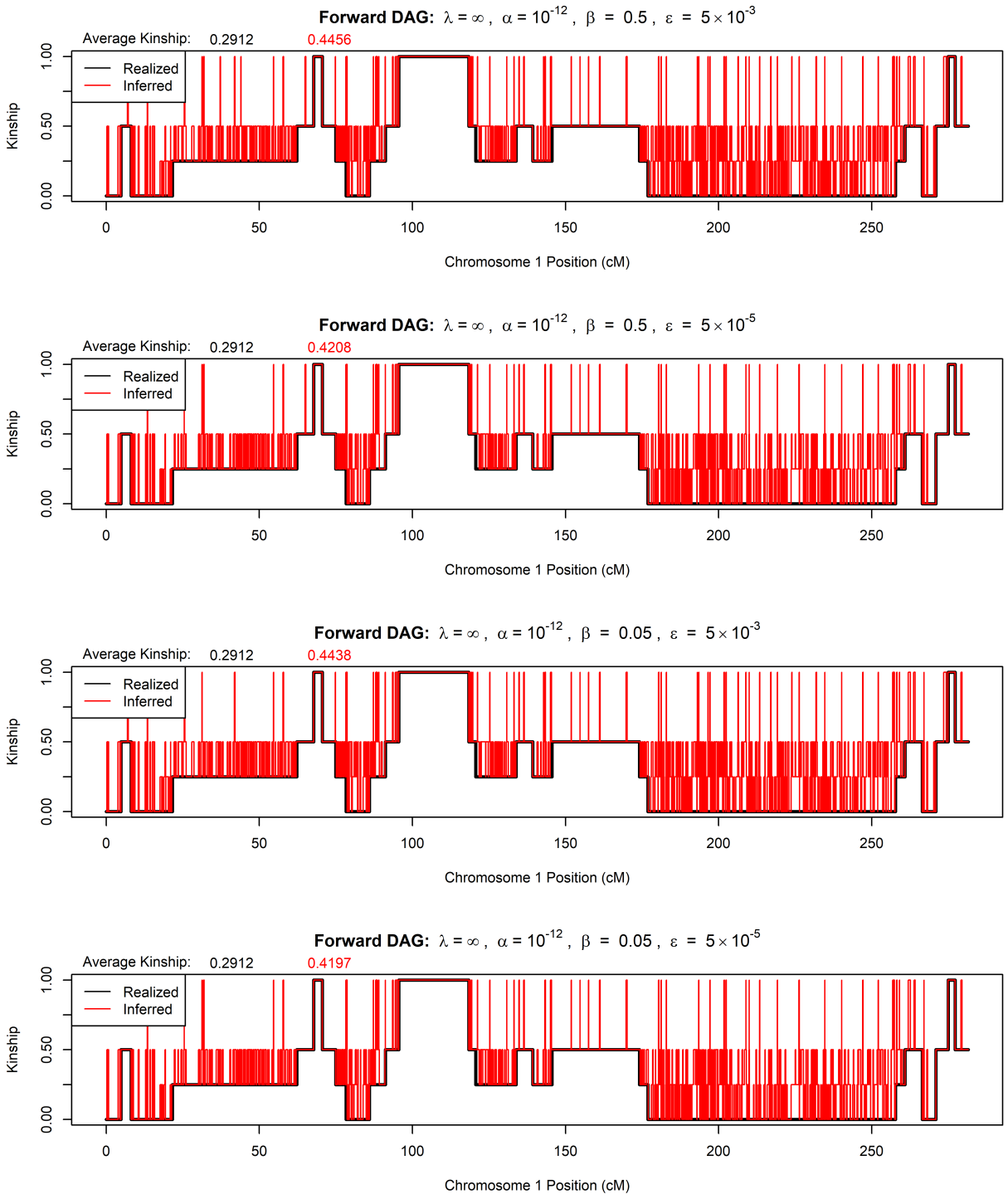

Figure B.16: *IBD segment detection results from our Viterbi algorithm when using a completely collapsed DAG with no LD structure ($\lambda = \infty$) at $\alpha = 10^{-12}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

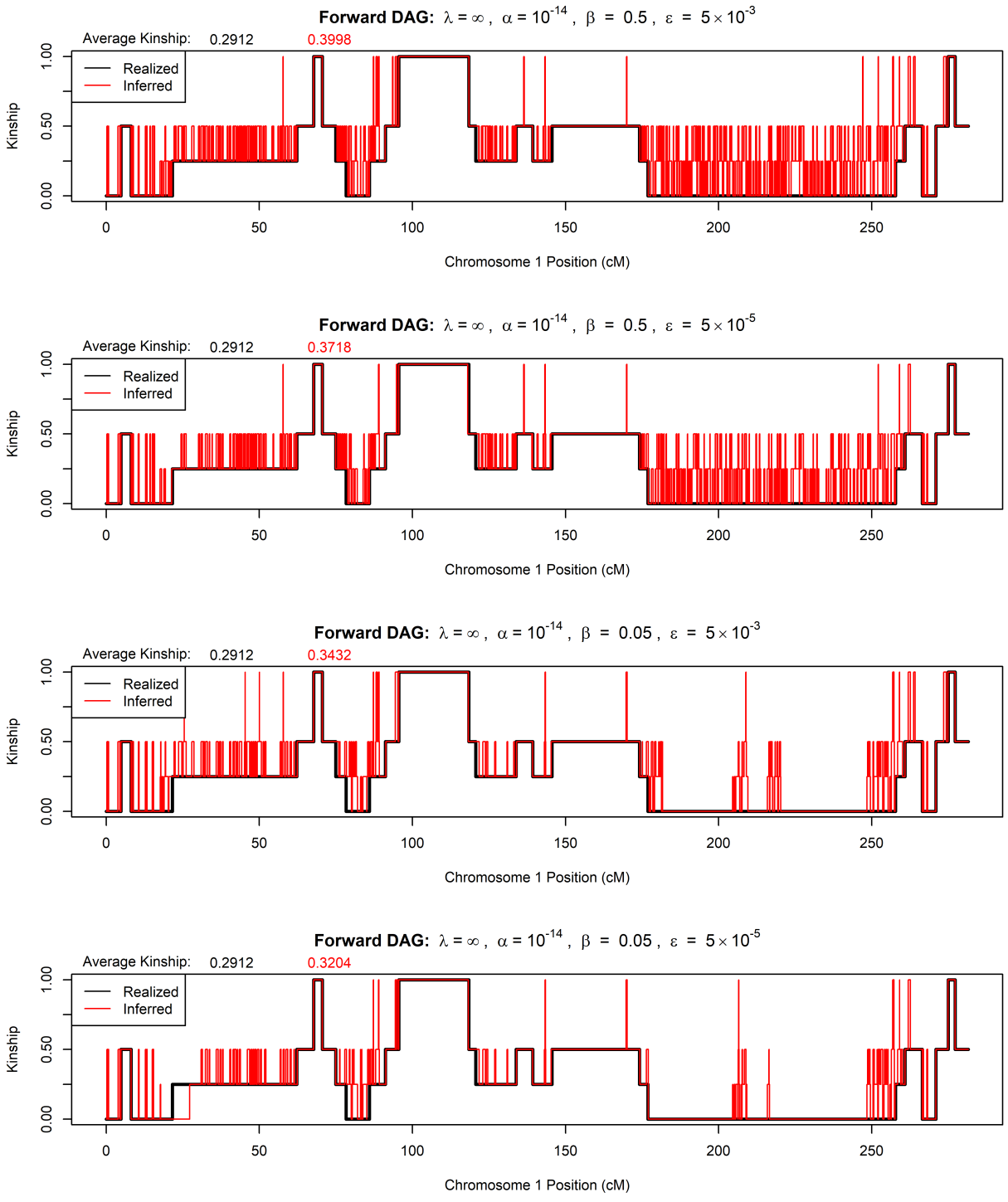

Figure B.17: *IBD segment detection results from our Viterbi algorithm when using a completely collapsed DAG with no LD structure ($\lambda = \infty$) at $\alpha = 10^{-14}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

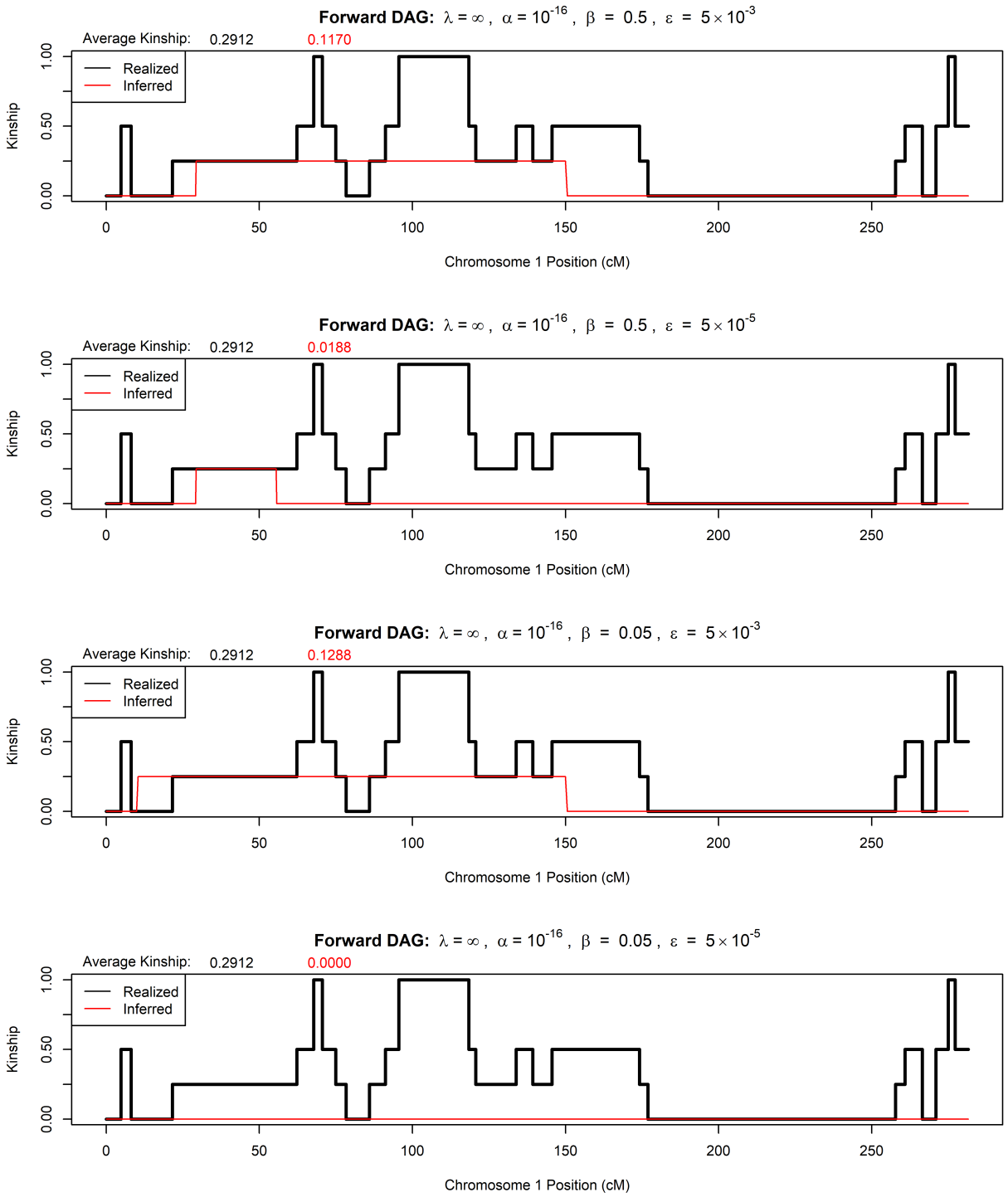

Figure B.18: *IBD segment detection results from our Viterbi algorithm when using a completely collapsed DAG with no LD structure ($\lambda = \infty$) at $\alpha = 10^{-16}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

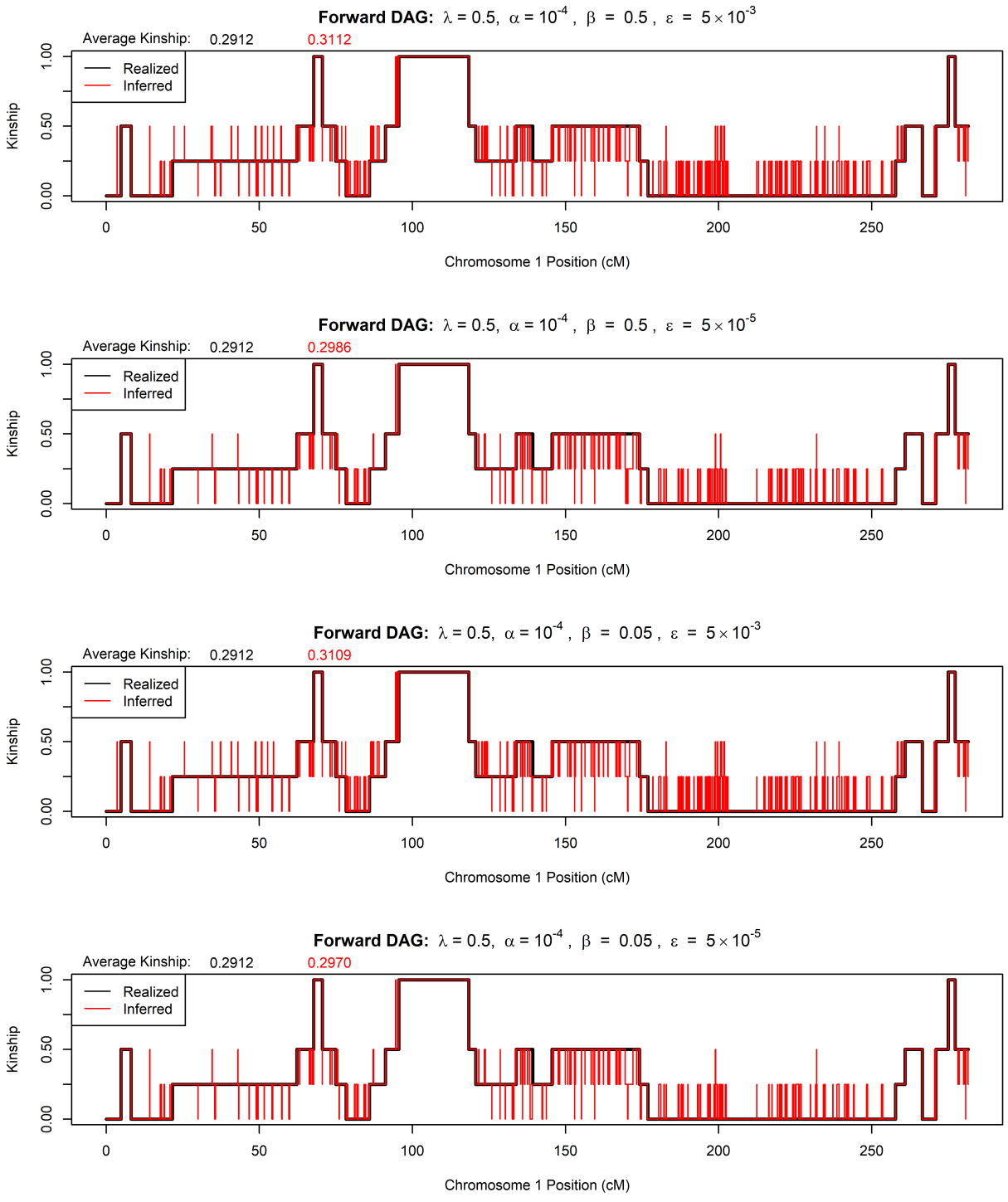

Figure B.19: *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at $\alpha = 10^{-4}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

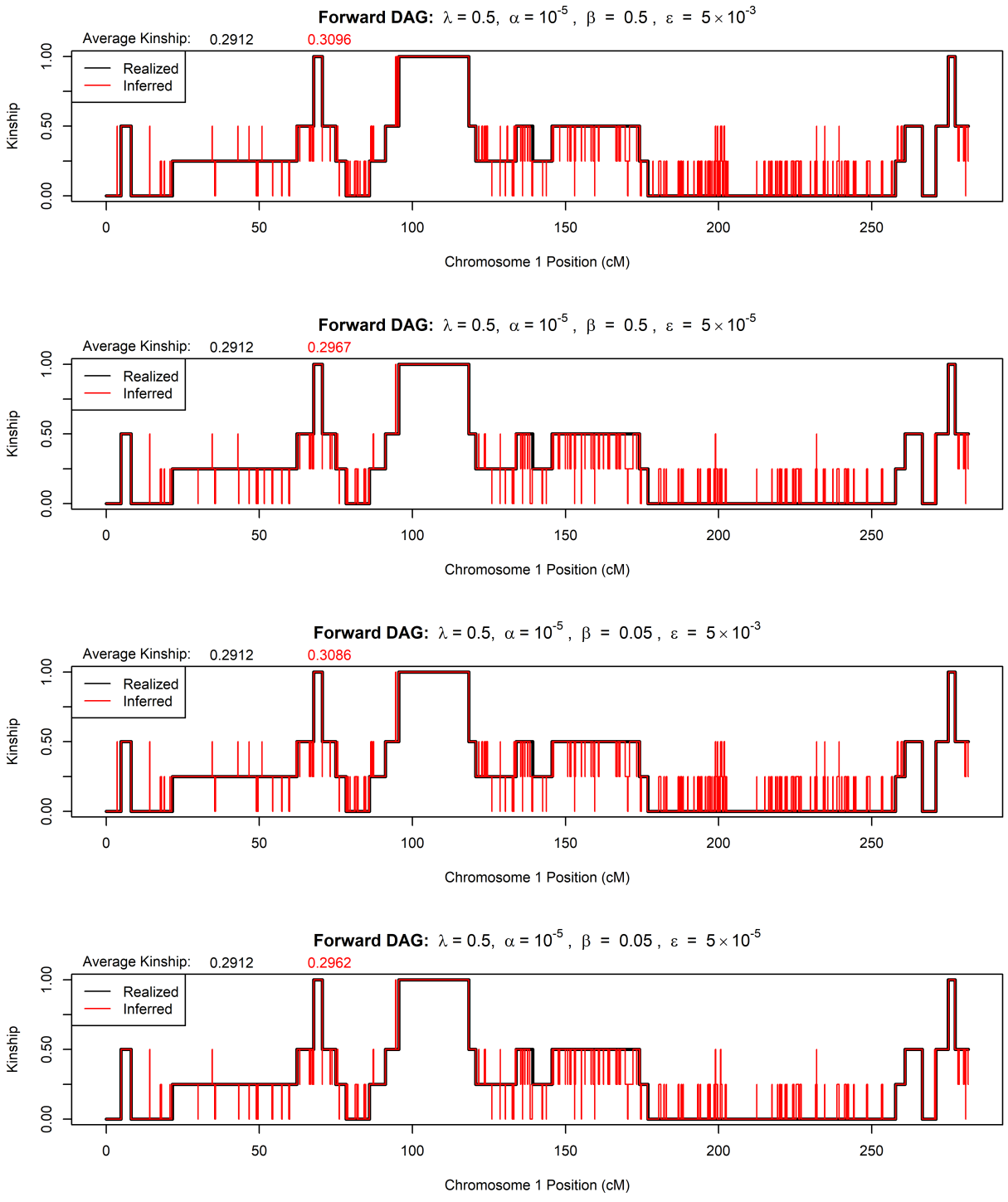

Figure B.20: *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at $\alpha = 10^{-5}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

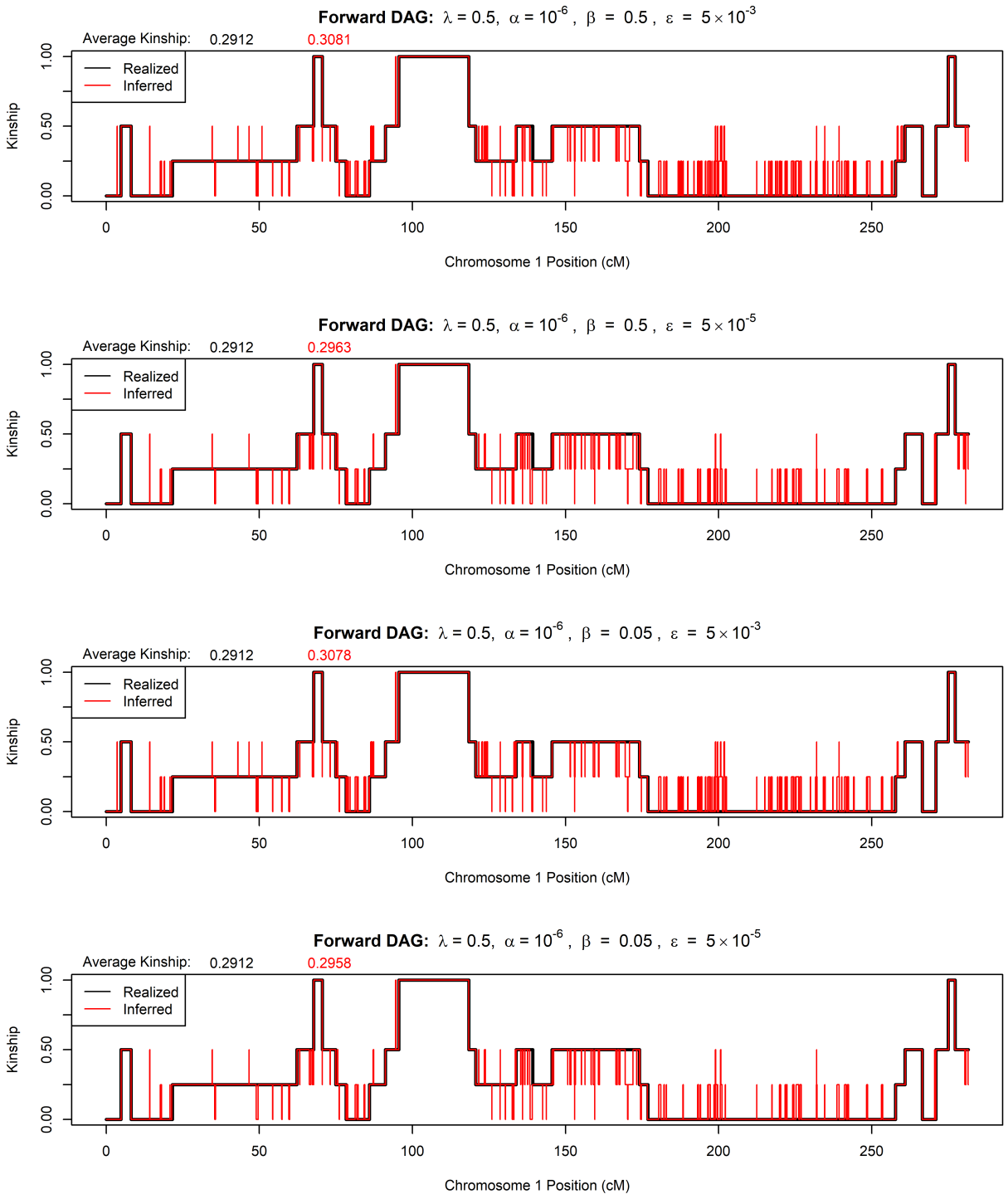

Figure B.21: *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at $\alpha = 10^{-6}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

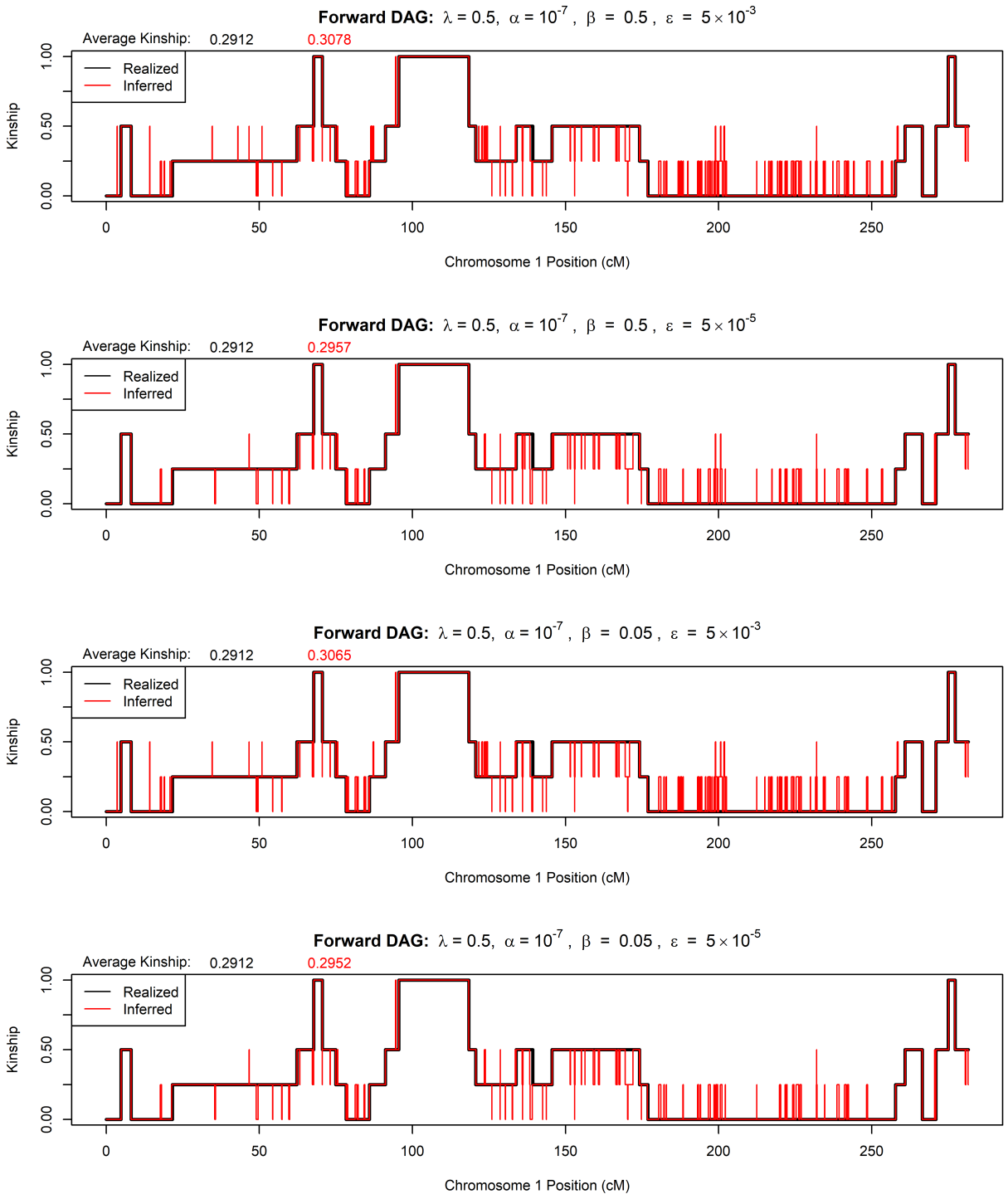

Figure B.22: *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at* $\alpha = 10^{-7}$ *and various values of* β *and* ϵ*. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal* λ *regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

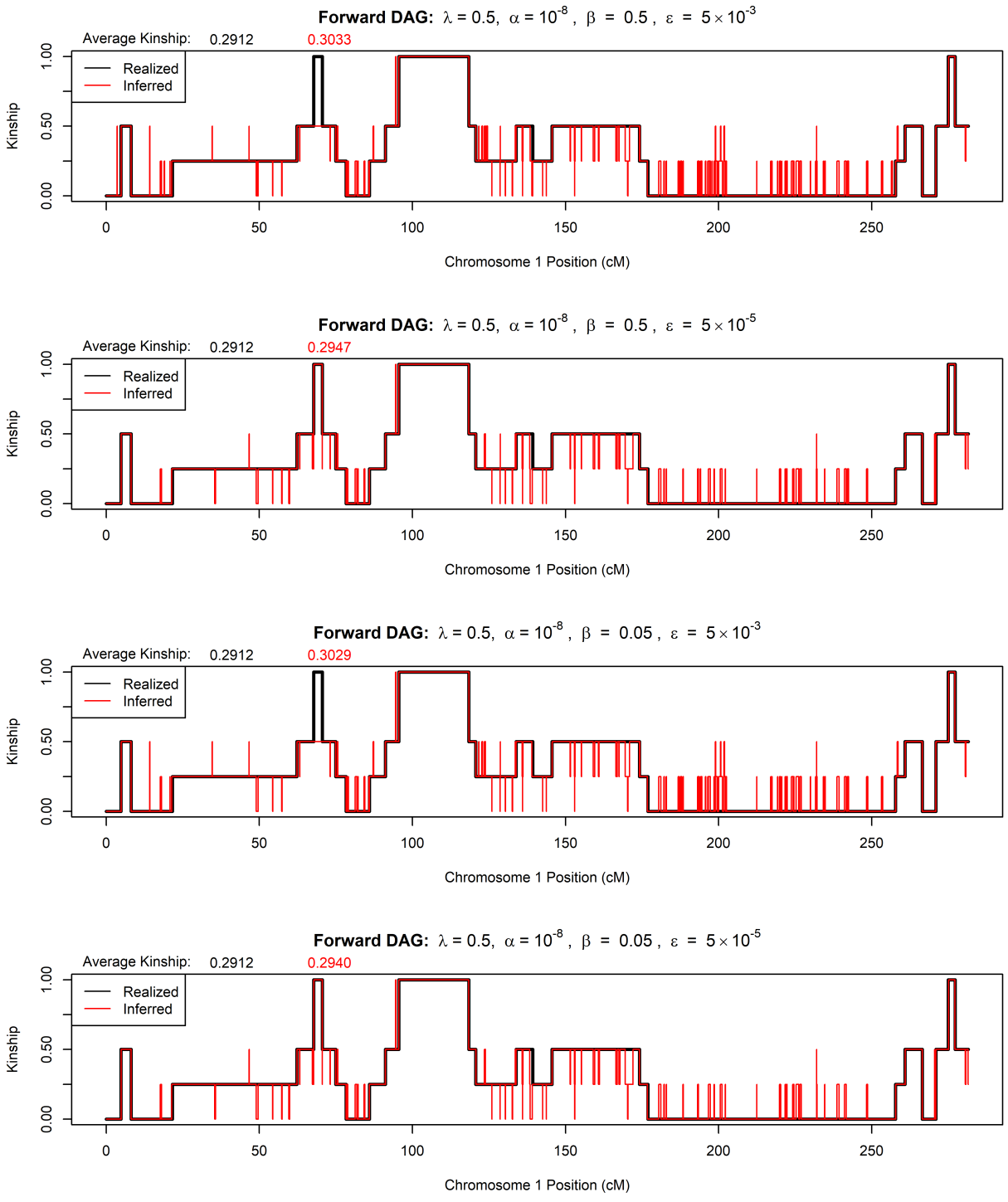

Figure B.23: *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at* $\alpha = 10^{-8}$ *and various values of* β *and* ϵ*. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal* λ *regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

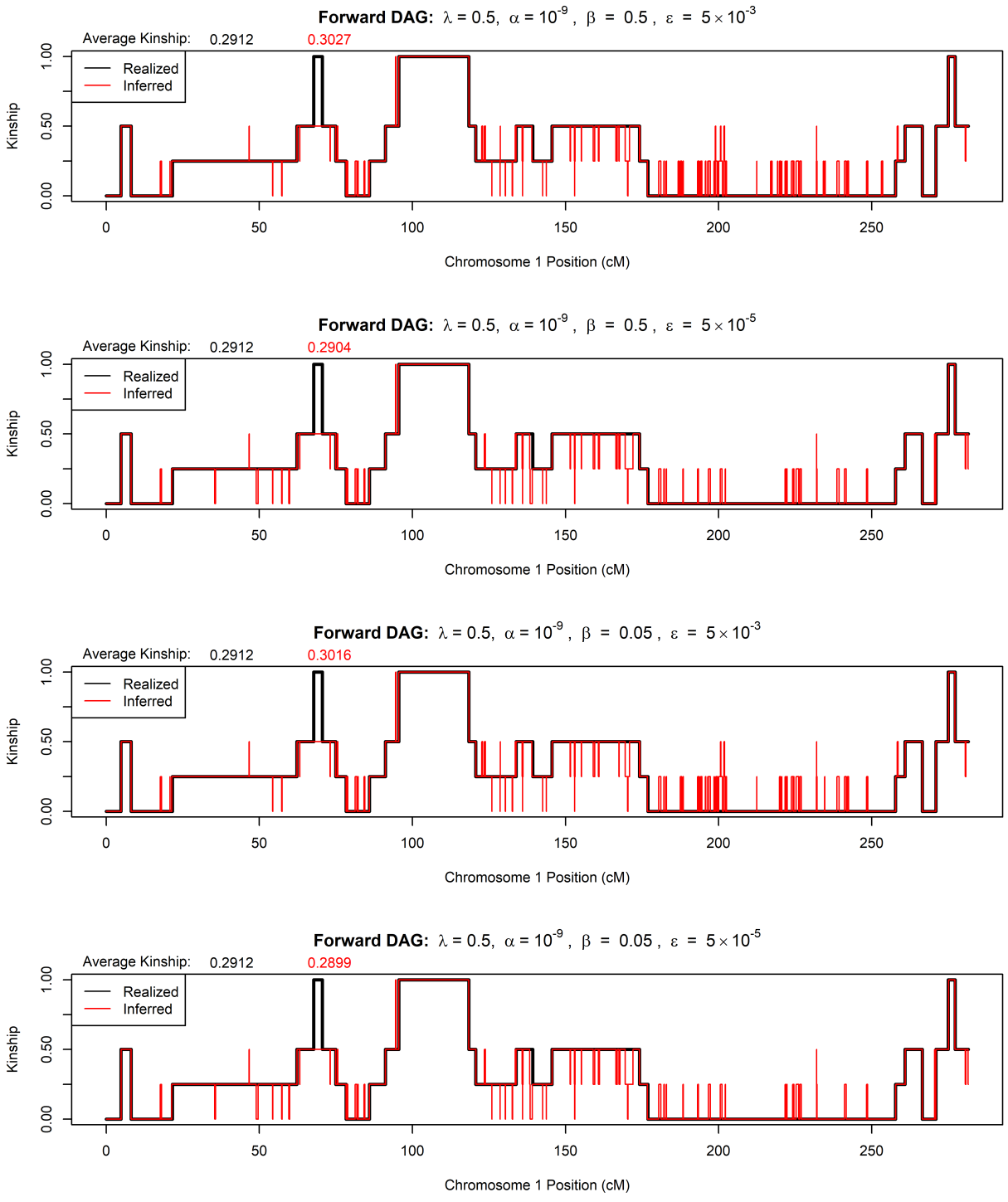

Figure B.24: *IBD segment detection results from our Viterbi algorithm run in the "forward" direction at $\alpha = 10^{-9}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

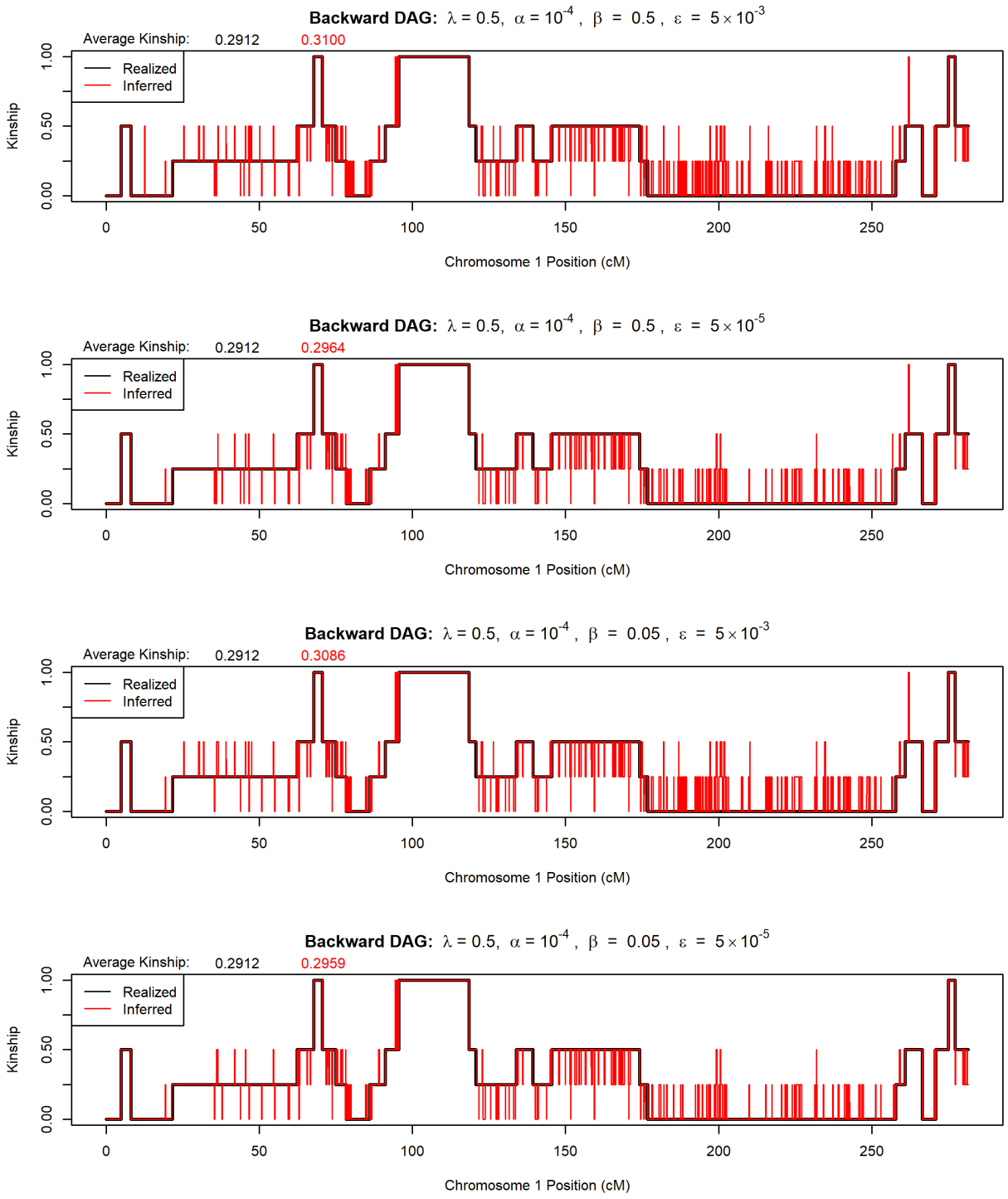

Figure B.25: *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-4}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

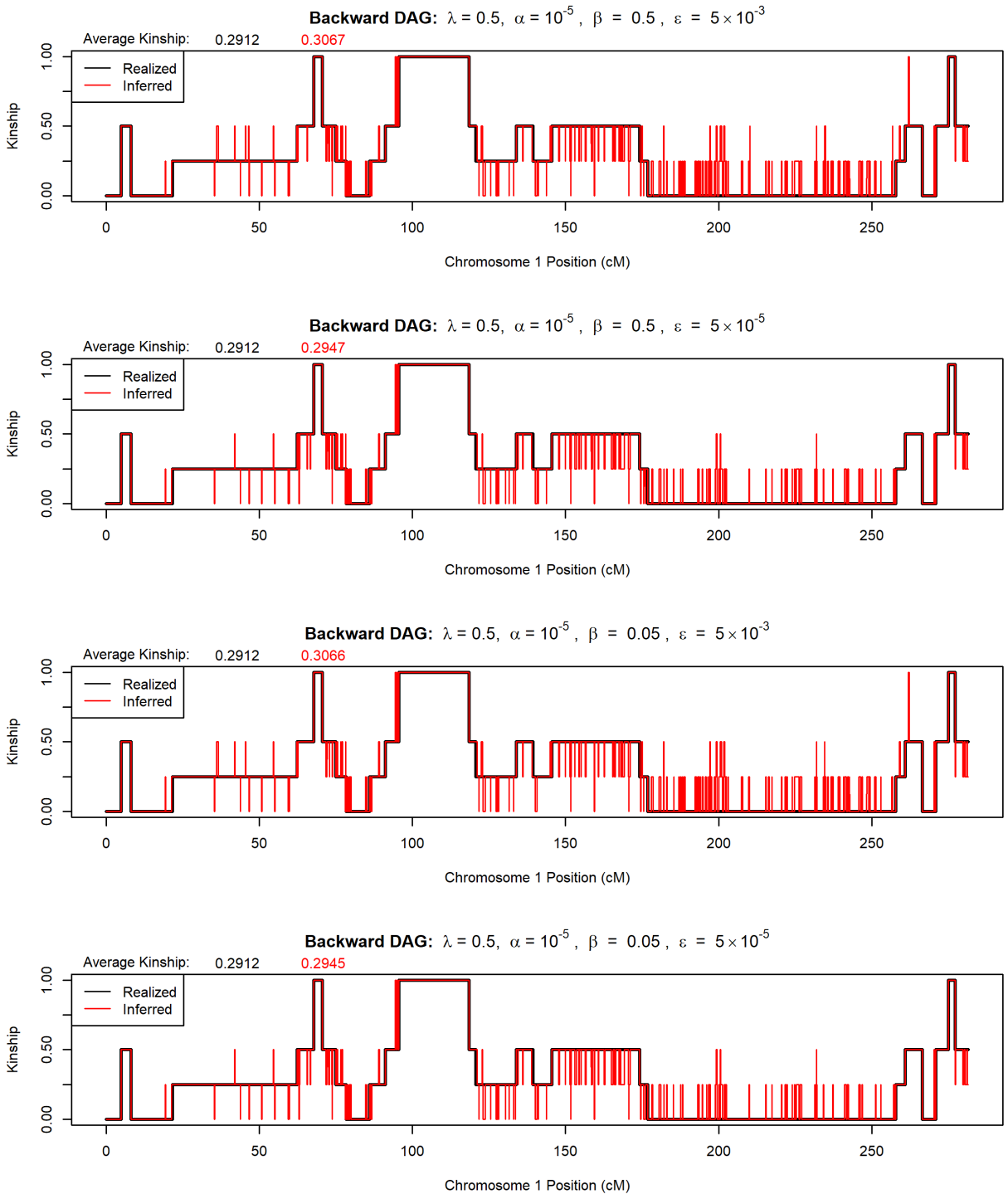

Figure B.26: *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-5}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

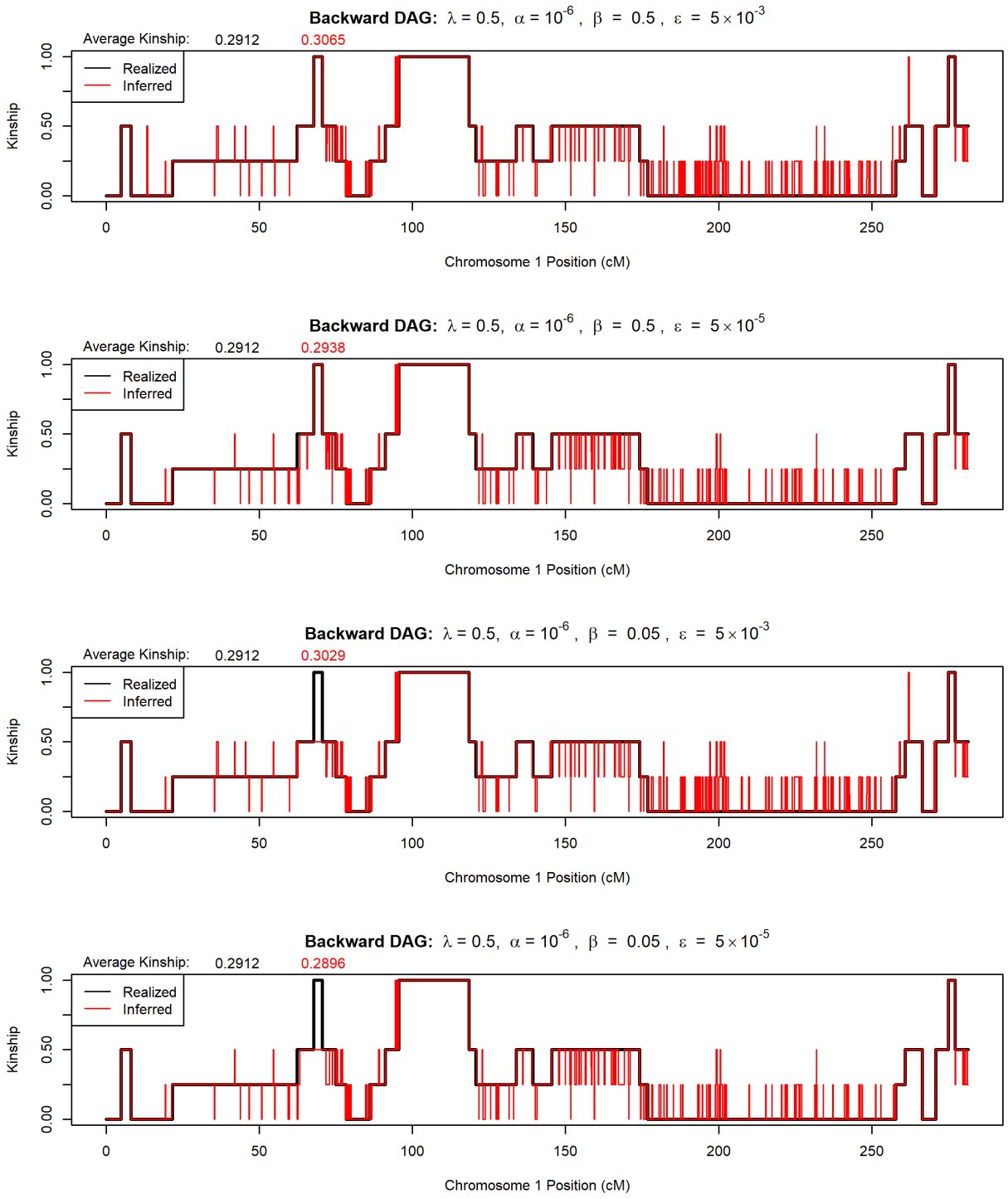

Figure B.27: *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-6}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

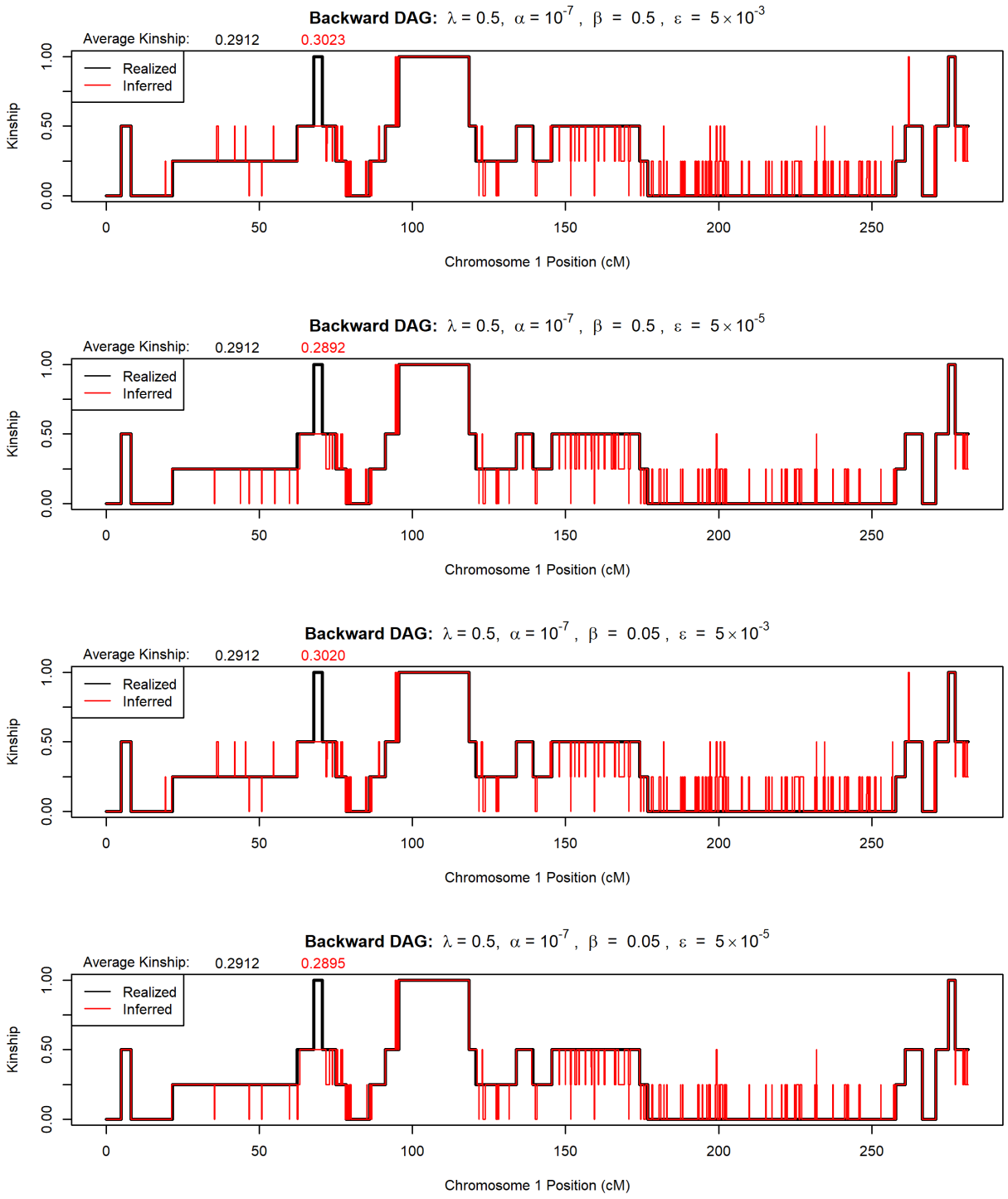

Figure B.28: *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-7}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

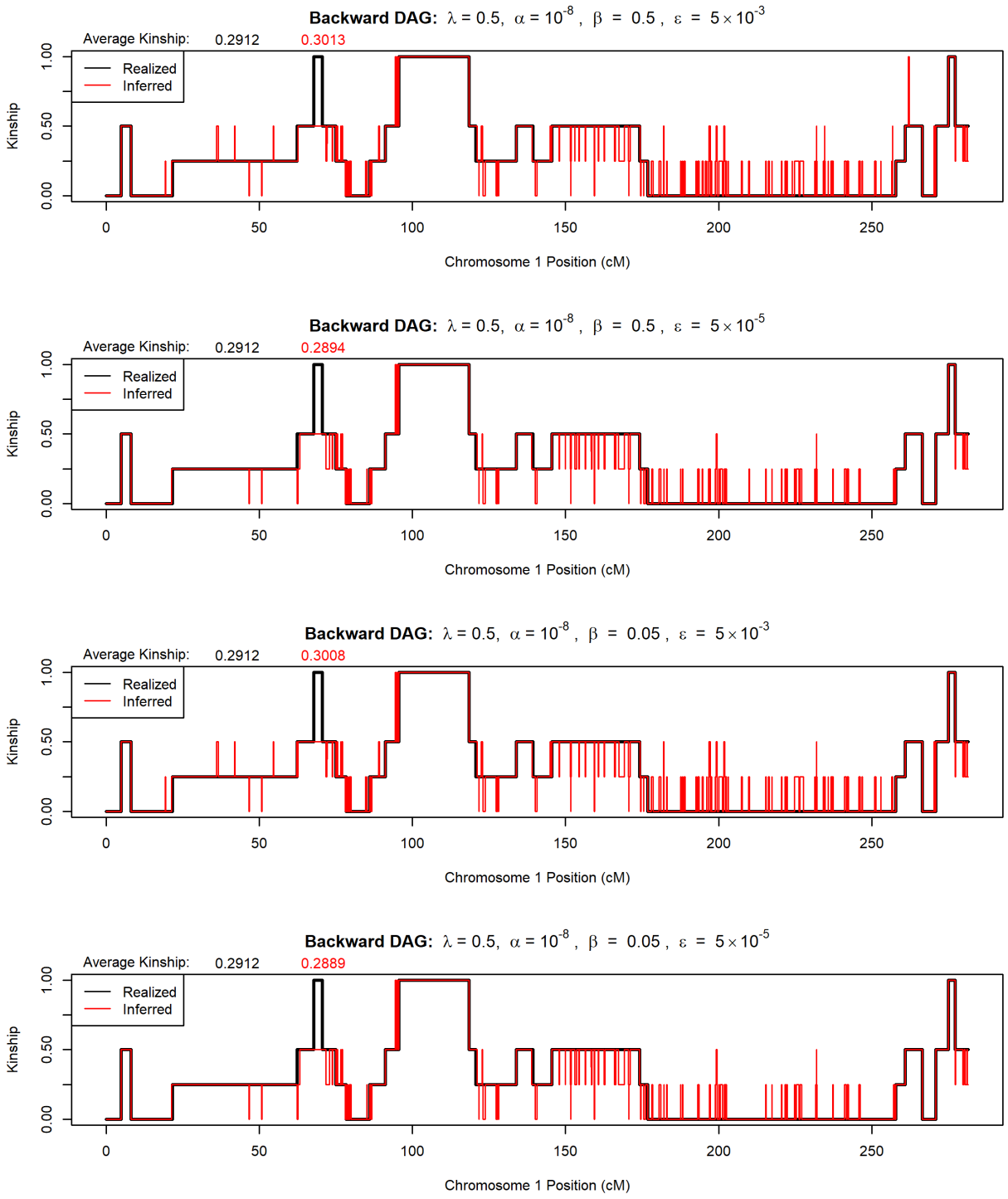

Figure B.29: *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-8}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

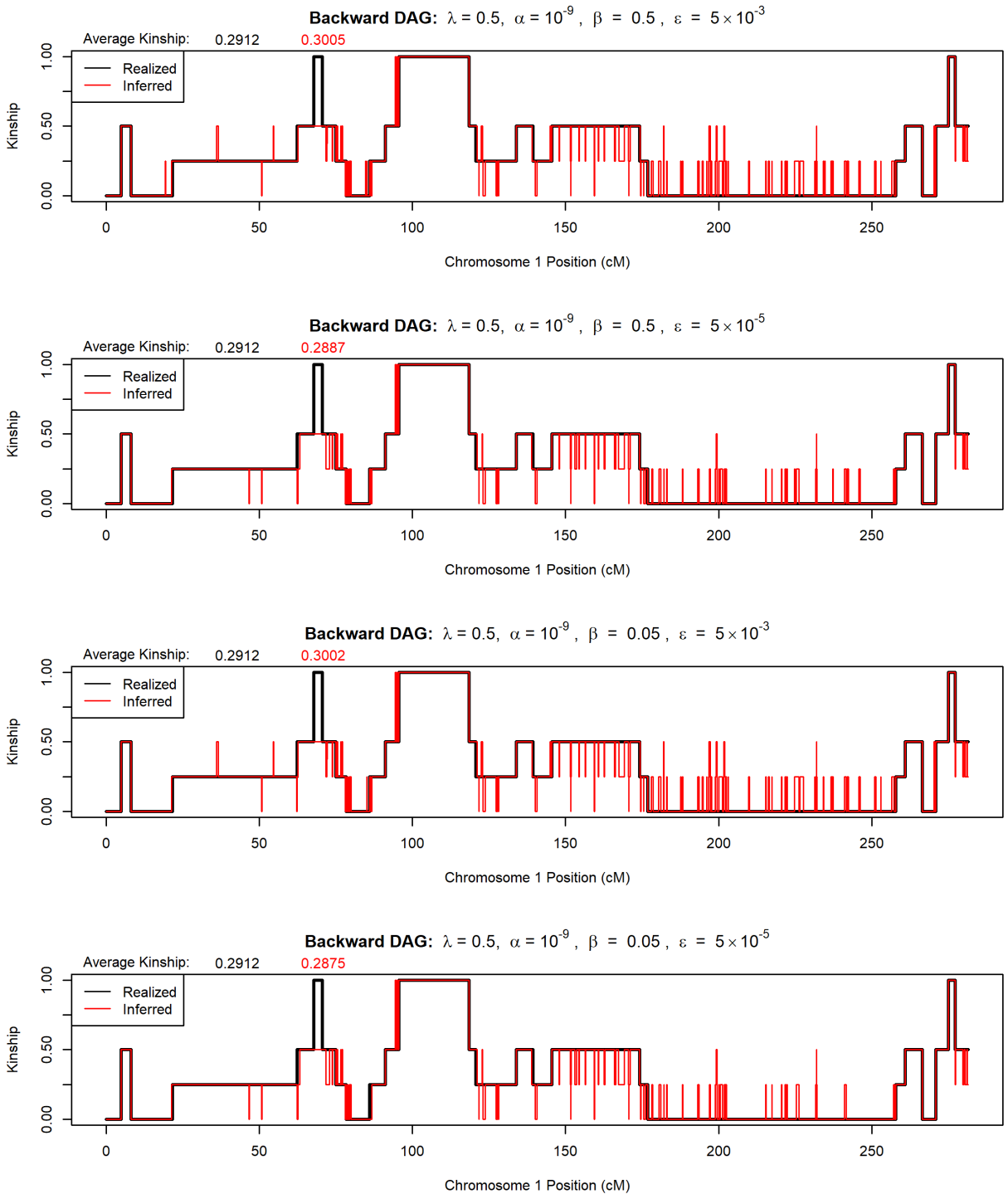

Figure B.30: *IBD segment detection results from our Viterbi algorithm run in the "backward" direction at $\alpha = 10^{-9}$ and various values of β and ϵ. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The DAGs fit using the optimal λ regularization parameter were used. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

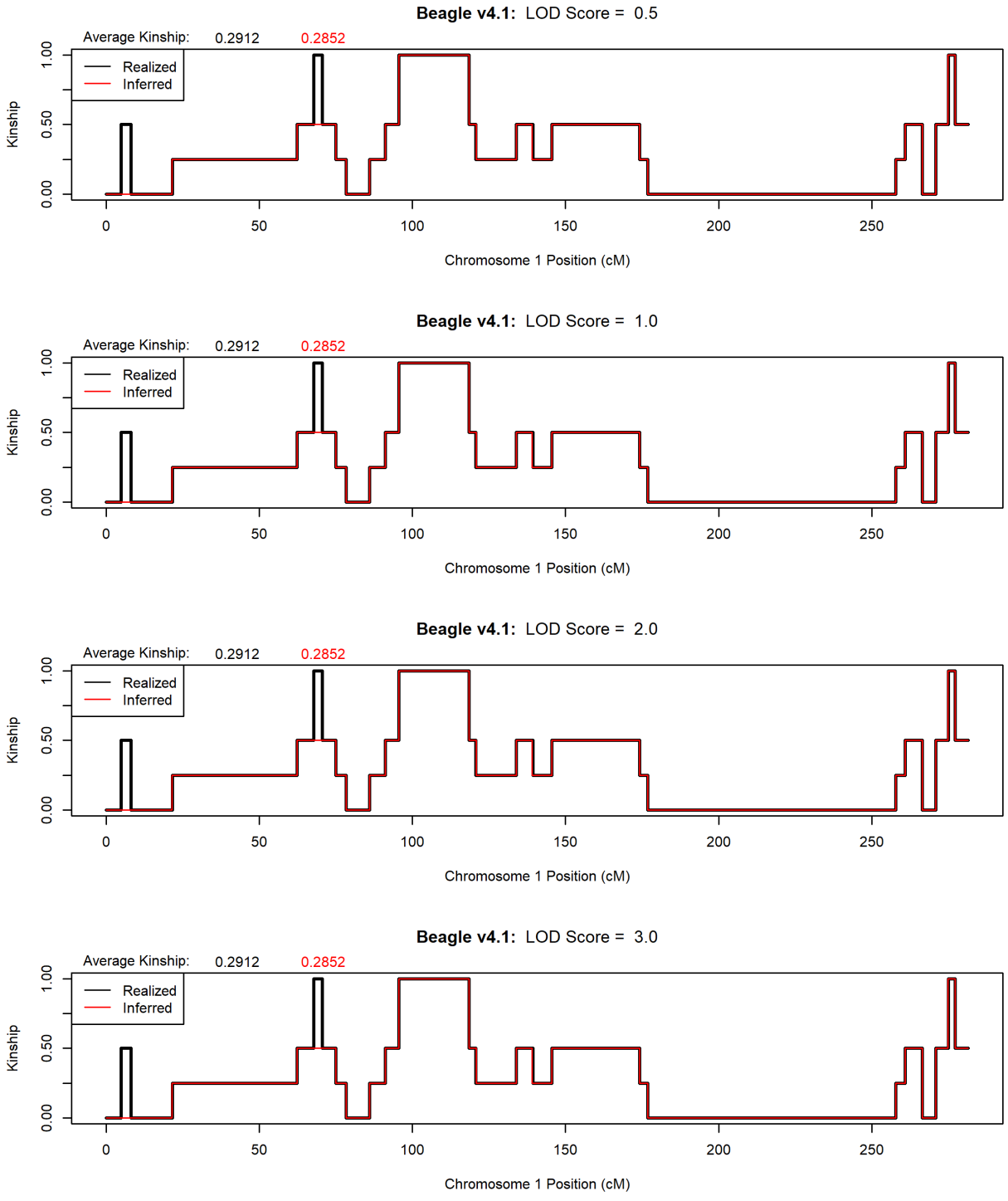

Figure B.31: *IBD segment detection results from Beagle v4.1 at various values of minimum LOD score. Haplotypes were simulated from the siblings in Figure 5.2 and no genotyping error was introduced. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*

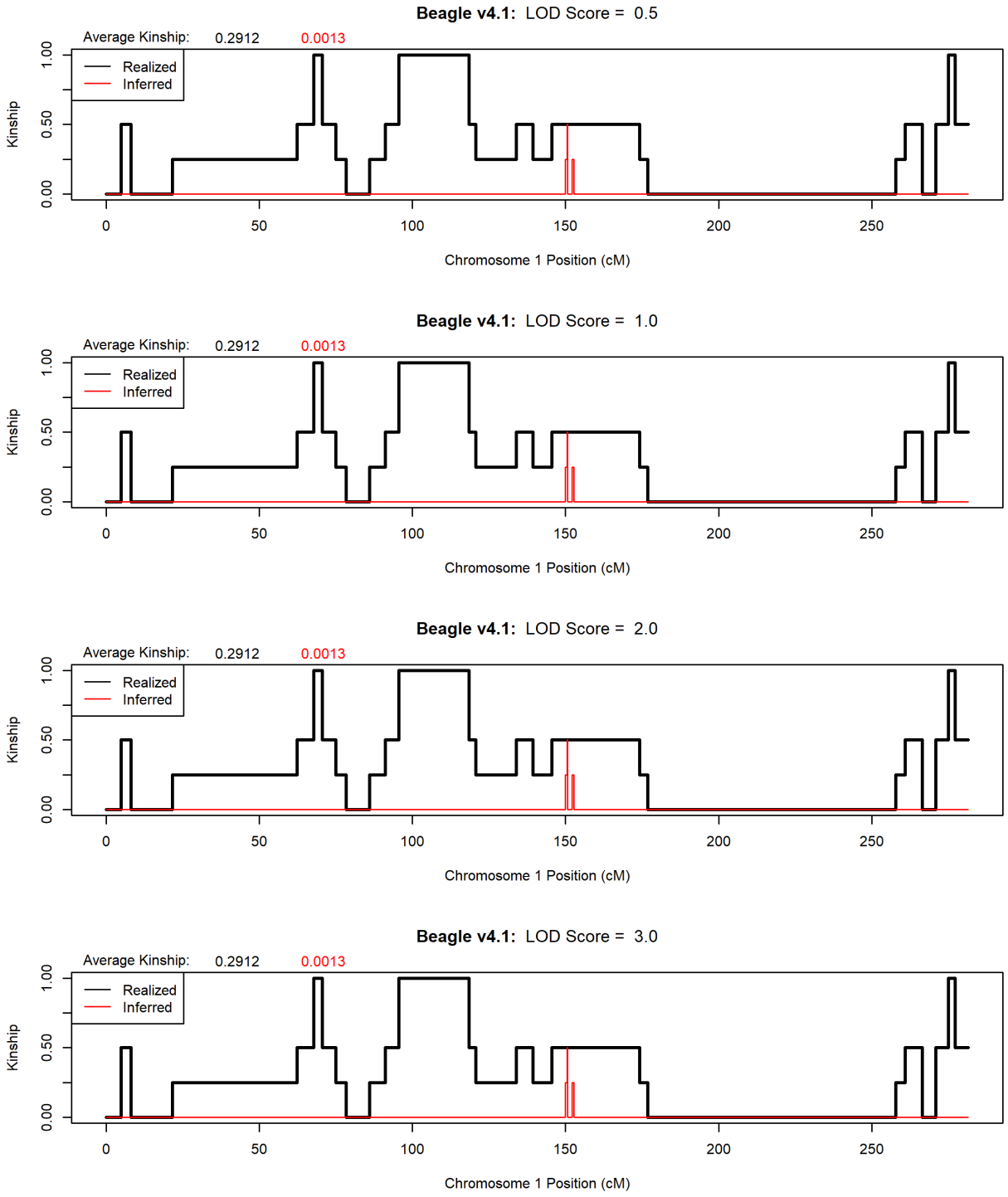

Figure B.32: *IBD segment detection results from Beagle v4.1 at various values of minimum LOD score. Haplotypes were simulated from the siblings in Figure 5.2 and genotyping error was introduced at a rate of 0.001 per marker. The black line is the realized kinship from the simulation, and the red line is the inferred kinship from each method.*