Fall 2012

Genotyping on custom arrays using a parallel data pipeline

Brian Albere
University of New Hampshire, Durham

GENOTYPING ON CUSTOM ARRAYS USING A PARALLEL DATA PIPELINE

BY

Brian Albere

B.S., University of New Hampshire (2011)

THESIS

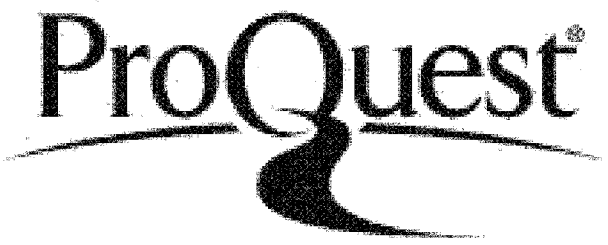
Submitted to the University of New Hampshire
in Partial Fulfillment of
the Requirements for the Degree of

Master of Science

in

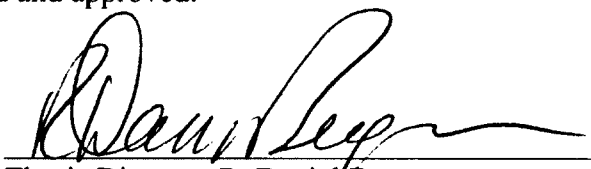
Computer Science

September, 2012



ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

This thesis has been examined and approved.



Thesis Director, R. Daniel Bergeron,
Professor of Computer Science

Philip J. Hatcher,
Professor of Computer Science

W. Kelley Thomas,
Professor of Molecular, Cellular, and Biomedical Sciences

5/15/2012

Date

DEDICATION

This thesis is dedicated to my mom and dad, for their never-ending support throughout my academic career, especially during my five years of college. They were my first teachers, and they are still teaching me today. No lessons from any professor in any classroom are as important as the ones they have taught me.

ACKNOWLEDGEMENTS

I would like to thank Professors R. Daniel Bergeron and W. Kelley Thomas for their help throughout this past year. They introduced me to the project, worked very closely with me throughout the project, and without their guidance I never would have understood the biology surrounding the project. I would like to thank Professor Philip J. Hatcher for being a supportive committee member and someone I could always turn to with a question throughout my college years. I would like to thank Eleanne Solorzano for her expert knowledge and support in the realm of statistics. Lastly, I would like to thank Michael Pfrender and Melissa T. Pullins at the University of Notre Dame, and Dieter Ebert and Jarkko Routtu at the University of Basel for providing the data for this project and suggesting various research paths.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

GENOTYPING ON CUSTOM ARRAYS USING A PARALLEL DATA PIPELINE

by

Brian Albere
University of New Hampshire, September, 2012

In the past, genotyping (determining a set of alleles in an organism) has been an extremely challenging process. The time, monetary, and technology demands of the task have limited genotype data to a small variety of scientific model organisms with the capacity to conduct genetic crosses. New sequencing technology from companies such as NimbleGen, however, can generate custom organism-specific microarrays at relatively low cost. The combination of these arrays and the knowledge of species' genome-wide SNPs allow genotype experiments, such as generation maps, QTL studies, and natural population variation studies, to be conducted on virtually any organism. Although the NimbleGen technology can create appropriate DNA information, there has been no software that can use this data for custom array-based genotyping.

This thesis describes a data pipeline that uses custom DNA microarrays to genotype organisms. The pipeline simplifies the genotyping process, and users can easily customize and run the tool. The pipeline's performance is improved by exploiting parallel aspects of the microarray data, which reduces the genotyping process from days and weeks to minutes and hours. We demonstrate that the pipeline is an effective tool for genotyping custom microarrays across a large number of loci, and describe the effects of user-controlled parameters.

CHAPTER 1

INTRODUCTION

Recent advances in genome sequencing have provided opportunities to develop numerous models to probe biological functions. Approaches, such as the analysis of Quantitative Trait Loci (QTL), provide critical links between the genotype and phenotype of an organism. Essential tools for the analysis of QTL, Genome-Wide Association Studies (GWAS), and genetic maps include access to a set of genome-wide polymorphisms between phenotypically unique strains, and cost effective means to genotype large numbers of individuals. With the dramatic advances in DNA sequencing, the generation of draft genome sequences for multiple individuals of a species provides a reference genome and the identification of a vast number of *single-nucleotide polymorphisms (SNPs)*. These known SNPs make it possible to develop microarray based methods to genotype specific loci.

Essential to these studies is the ability to genotype organisms. Genotypes provide a basic method of comparison between the genetic variations among individuals. If it were possible to genotype a large number of loci within a large number of individuals, subsequent studies may drastically improve.

Standard protocols for high throughput genotyping have in the past been limited to a small number of model systems where genotyping arrays are commercially available.

The number of genotyped loci has been relatively small due to the "loci by loci" genotype procedures. Fortunately, recent opportunities to create cost-effective custom arrays bring the genotyping capability to a larger number of organisms. The microarray design of the custom array procedure massively parallelizes the genotyping process. This allows a large number of loci that span an organism's entire genome to be genotyped.

NimbleGen has created a custom array technology that identifies nucleotides at specific positions within an organism's genome. Through the use of oligonucleotides and chemical markings that generate luminescent effects when DNA fragments hybridize, NimbleGen's array hybridization technique should be able to genotype thousands of loci across many individual organisms' genomes. The lab procedure is cost-effective, and refer to the *Array Hybridization Technique* section for more information (http://www.nimblegen.com/, 2012).

Bioinformatic tools that use these arrays to genotype organisms are not readily available. To address this issue we have developed a computational pipeline that produces genotypes based on array data. The pipeline is an improvement upon past genotyping methods because it can genotype a large number of loci within a large number of individuals in a relatively short amount of time (Routtu et al., 2010).

1.1 The Experiment

The pipeline was originally developed for a QTL analysis and genetic mapping using a *Daphnia magna* panel. *Daphnia* is one of the best-understood ecological models in biology, and it has recently been the focus of genomic resource development (Colbourne et al., 2011). In our experiment, the focus was on a genetic cross between

two distinct ecotypes of *Daphnia magna* (see *Fig. 1*). Genotyping these organisms, and their various offspring, was essential to the QTL analyses and genetic mapping.

The experiment's biological structure includes three generations of *Daphnia magna*. Two distinct ecotypes were selected as the parent clones. The first was the Iinb1 clone (*I clone*) from Munich, Germany, and the second was the Xinb3 clone (*X clone*) from Tvärminne, Finland (Routtu et al., 2010). Note that the term "clone" here refers to the organisms' ability to reproduce by parthenogenesis: a form of asexual reproduction that produces genetically identical offspring. The parent clones were selected for the following reasons:

- The clones are phenotypically different. This allows a larger number of phenotypes to be mapped to genotypes in a QTL study.
- The clones are genotypically different. In theory, this should increase the number of SNPs with one genotype in one parent, and a different genotype in the opposite parent.
- Each clone was heavily inbred. In theory, this should increase the number of homozygous SNPs within each clone's genome.

The two parent clones were crossed to generate a single offspring: the *F1*. The goal was to identify SNPs that were heterozygous in the F1 and homozygous in the parent clones. In theory, the number of heterozygous F1 SNPs should be high because the parent clones are genotypically different. The F1 was assembled using a de novo assembly, and thousands of heterozygous SNPs were identified across its genome. The X

clone was reference assembled against the F1, which made it possible to determine a subset of the identified SNPs that were heterozygous in the F1 and homozygous in the X clone. A subset of 16673 SNPs was selected, and attempts were made to span the entire *Daphnia magna* genome with the subset. The I clone was not assembled, and it was assumed the SNPs were homozygous in the I clone. The SNP nucleotides in the F1 and the X clone were determined by examining their assemblies, and the SNP nucleotides in the I clone were assumed to be the SNP nucleotides in the F1 that were not inherited from the X clone.

The F1 organism was cloned to produce a second F1, and the two organisms mated multiple times to generate the *F2* generation. Roughly 350-450 F2 organisms were generated (refer to the *Experimental Evaluation* section for more specifics). The F2 organisms are the focus of the experiment and the focus of the pipeline. Due to the heterozygosity of the SNPs in the F1 organisms and the homozygosity of the SNPs in the parent clones, each SNP within each F2 has three possibilities (once again, see *Fig. 1*):

- The SNP's nucleotides on both associated homologous chromosomes were inherited from the I clone. This should occur 25% of the time. These SNPs are homozygous, and their associated mapping allele is identified as "A".
- The SNP's nucleotides on both associated homologous chromosomes were inherited from the X clone. This should occur 25% of the time. These SNPs are homozygous, and their associated mapping allele is identified as "B".
- The SNP's nucleotides on one of the associated homologous chromosomes were inherited from the X clone, and the SNP's nucleotides on the opposite

chromosome were inherited from the I clone. This should occur 50% of the time. These SNPs are heterozygous, and their associated mapping allele is identified as "H".

The pipeline's purpose is to calculate which of the above cases applies to each SNP within each F2. The pipeline streamlines and simplifies these calculations, and the result is a genotyping of each F2 at every SNP. The pipeline's results allow the genetic mapping and QTL studies to move forward. At the time this paper was written, both studies were still in progress.

The genetic mapping will examine each SNP genotype within each F2, and it will use the recombination events to determine linkage groups, chromosomes, and an ordering of the *Daphnia magna* contigs. During sexual reproduction, an offspring receives pairs of homologous chromosomes from its parents. When considering a single pair of homologous chromosomes (called a "Y" pair), it is easy to visualize the reproduction process as taking one chromosome from one parent's Y pair (the "A" parent) and one chromosome from the other parent's Y pair (the "B" parent). However, it is not that simple. Many times, a combination of parent A's Y pair chromosomes is passed to its offspring. *Crossover events* (recombination events) cause DNA from one chromosome to bind with subsequent DNA from the opposing homologous chromosome, resulting in a brand new chromosome (see *Fig. 1*). The new chromosome is passed to the offspring as one half of the offspring's Y pair. The other half is generated by the B parent, and similar crossover events can occur. In terms of an entire genome, crossover events happen relatively infrequently (Routtu et al., 2010). Therefore, the genetic mapping

determines a relative ordering of the *Daphnia magna* SNPs by comparing each SNP's genotypes across all F2 organisms against every other SNP's genotypes across all F2 organisms. If one SNP's genotype is the same as another SNP's genotype within a high percentage of the F2 individuals, it infers the SNPs are close within the *Daphnia magna* genome. If SNPs "A" and "B" have a higher similarity percentage than SNPs "A" and "C", the relative ordering of the SNPs will be A, B, C. The algorithm is obviously more complicated than suggested, but the end result is a relative ordering of all the SNPs and their containing contigs. The ordering can be used to identify linkage groups, chromosomes, and an overall genetic structure of the *Daphnia magna* genome (Routtu et al., 2010).

The Quantitative Trait Loci study will relate *Daphnia magna* genotypes to phenotypes. Imagine the X and I clones differ in a certain phenotype ("C"). The QTL study examines the F2 individuals and determines the subset ("S") that expresses the X clone's version of the C phenotype. The S individuals' SNPs are then examined, and the subset ("H") that is homozygous and inherited from the X clone across all S individuals is determined. It is then inferred that the H SNPs and the surrounding DNA must be responsible for the C phenotype (Broman et al., 2003).

Genetic mappings and QTL studies are two examples of how the pipeline's genotype data can be used. Researchers should be able to conduct similar studies on any custom organism by utilizing the pipeline and NimbleGen's technology.

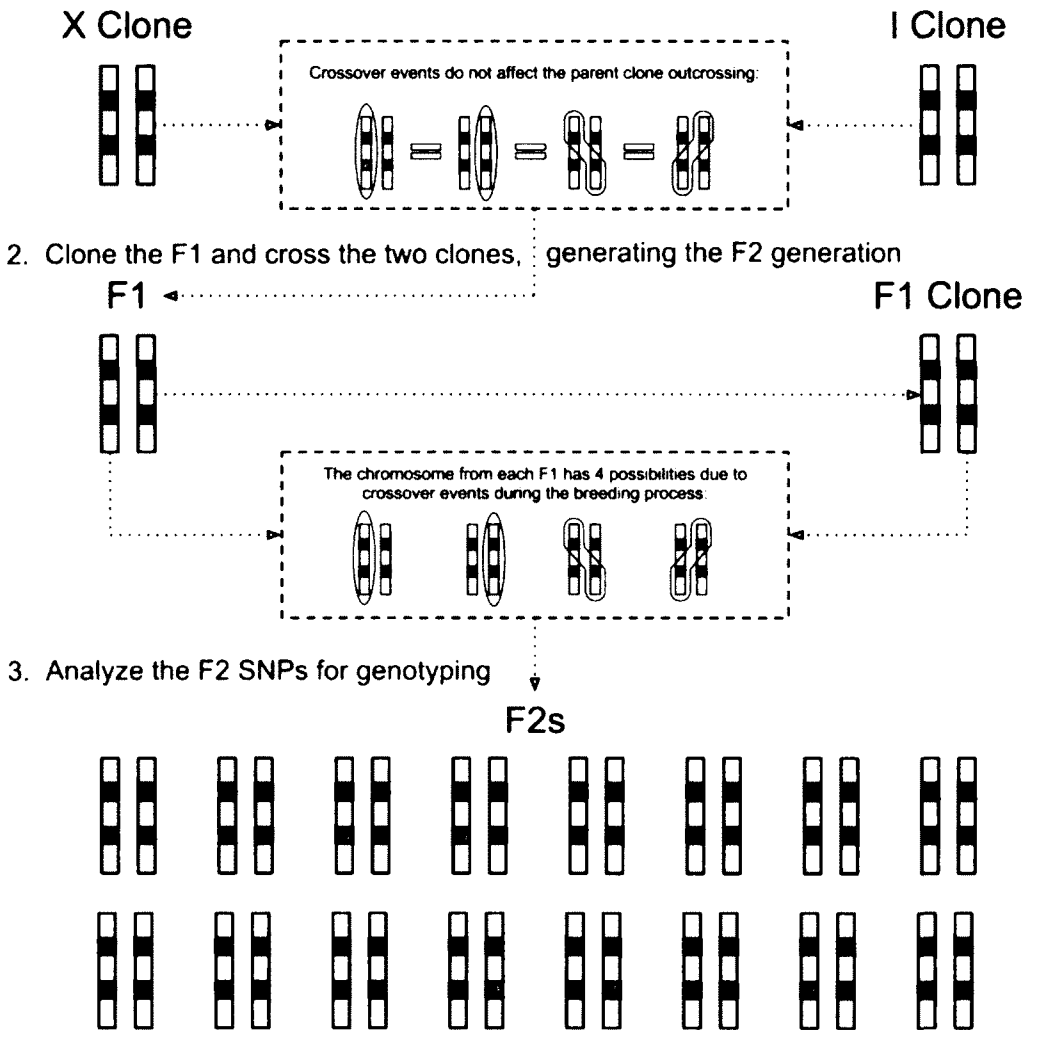


Figure 1 - The Biological Structure of the Genotype Experiment

There are three generations of organisms. Each pair of elongated rectangles represents the same pair of homologous chromosomes in different organisms. The arrows represent the breeding processes, and the small squares indicate SNP nucleotides (each individual has two SNPs represented on one pair of homologous chromosomes). Each offspring takes one chromosome from each of its parents. Crossover events do not affect the outcrossing between the X and I parent clones because the homozygosity of the parent clone SNPs causes the SNP nucleotides on the chromosome inherited from each clone to not change even when a crossover event occurs. The F1 SNPs are therefore heterozygous. The F1 mates multiple times with its clone organism to produce the F2 generation. The potential of crossover events during the F1 mating process results in 16 possible F2 individuals in the above example. Each individual SNP across all F2 individuals is homozygous from the X clone 25% of the time, homozygous from the I clone 25% of the time, and heterozygous from both parent clones 50% of the time. Note that this distribution only occurs in a perfect experiment.

The figure illustrates the genetic mapping concept that relative distances between SNPs can be determined based on the frequency of crossover events. When a crossover event does happen, the two illustrated SNPs do not have the same genotype, inferring they are likely far apart in the genome. When a crossover event does not happen, the two illustrated SNPs have the same genotype, inferring they are likely close in the genome. The frequency that two SNPs share the same genotype across all F2 individuals determines the relative distance between the two SNPs.

1.2 The Array Hybridization Technique

The array hybridization technique is NimbleGen's lab procedure that creates and processes microarray slides, and generates the pipeline's input. The technique identifies nucleotides at specific positions in an organism's DNA. In our experiment, the specific positions are SNPs, and the DNA comes from *Daphnia magna* organisms. The pipeline was designed to interface with NimbleGen's array hybridization technique and the output of NimbleGen's NimbleScan software (http://www.nimblegen.com/, 2012).

The array hybridization technique begins by preparing *microarray slides*. A slide is generated for each individual to be genotyped, and each slide contains *oligonucleotides* (oligos) at specific locations. Oligonucleotides are short sequences of DNA, and each SNP has eight associated oligos. Four of the associated oligos relate to the *forward DNA strand*, and the remaining four relate to the *reverse DNA strand*. The oligos match the associated organism's DNA surrounding the associated SNP in the associated strand. Each oligo's nucleotide at the associated SNP's position is different within the four strand-related oligos. One oligo contains an "A", another a "C", another a "T", and the last a "G". The associated organism's DNA is sheared and converted into a labeled DNA fragment solution. The solution is washed over the slide, and DNA fragments containing SNPs hybridize with their complement oligo. The process is not perfect, and fragments may hybridize with oligos that are not an exact complement. The DNA fragments and the oligos are chemically altered to generate a luminescent effect when hybridized. The more DNA fragments that hybridize at an oligo position, the brighter the effect (http://www.nimblegen.com/, 2012).

NimbleGen generates an image for each microarray slide (refer to appendix D for a sample microarray image). This ends the lab procedure. The images are then sent through NimbleGen's NimbleScan software. Based on the known oligo locations within each image, an output file, called a *bases* file, is generated for each image. Each bases file contains every oligo's luminescent readings from the associated image (http://www.nimblegen.com/, 2012). Refer to appendix D for detailed input specifications. The bases files act as the pipeline's input. It is the pipeline's job to parse the luminescent data and determine each SNP's genotype within each F2.

CHAPTER 2

THE PIPELINE

The pipeline's purpose is to genotype organisms based on the luminescent data in bases files. The high-level logic to determine a SNP's genotype within an F2 is as follows:

- Classify the SNP as heterozygous or homozygous based on the SNP's eight luminescent readings (four readings for the forward DNA strand, and four readings for the reverse DNA strand).
- Based on the SNP's classification and luminescent readings, determine the SNP's nucleotides on the associated pair of homologous chromosomes.
- Compare the SNP's nucleotides to the X and I clones' nucleotides to determine the SNP's mapping allele.

In a general sense, the pipeline contains one or more specific computation steps for each logical step. Extra computations remove invalid or noisy data, and every F2 and their associated SNPs are processed at once. The details of each pipeline step are discussed in the *Implementation* section; however a high-level description is provided below (see *Fig. 2*).

There are five pipeline steps, and the pipeline's input is composed mostly of bases files (refer to appendix D for detailed input specifications). Note that each organism, or individual, relates to a single microarray slide, which relates to a single microarray image, which relates to a single bases file, which is interchangeable with the term microarray data set. The steps are executed in the following order.

The *Spatial Smooth* step corrects non-biological noise in the bases files' luminescent readings. Microarray slides are susceptible to an uneven distribution of the DNA fragment solution during the hybridization process (see *Fig. 3*). Consequently, luminescent readings in physical sections of the slides are generally higher or lower than others (Wang et al., 2006). The Spatial Smooth step balances the bases file data before the mapping alleles are calculated.

The *HomoIndex* step computes each SNP's mapping allele within each individual. The key calculation identifies each SNP as heterozygous (the SNP's nucleotides are different on the associated homologous chromosomes) or homozygous (the SNP's nucleotides are the same on the associated homologous chromosomes). The calculation achieves this by computing standardized numbers called *HomoIndex values*. Each HomoIndex value is based on four strand-related luminescent readings pertaining to a specific SNP. Therefore, each SNP within each individual has two associated HomoIndex values. The values are between zero and one, and values near zero indicate homozygous SNPs, and values near one indicate heterozygous SNPs. The homozygous or heterozygous classifications and the luminescent readings determine each SNP's nucleotides in both strands on both associated homologous chromosomes. The nucleotides are compared to the nucleotides in the X and I clones, and each SNP's

mapping allele is determined. Pre-filter computations recognize data errors caused by imperfections in the hybridization process. Mapping alleles are marked as valid or invalid based on these errors, and subsequent pipeline steps remove entire SNPs and microarray data sets based on the invalid markings.

The *FIGMDP* step (Filters, Initial Genotype Mapping, and Dip Preparation) removes invalid data and prepares subsequent pipeline calculations. It is hypothesized that certain oligos (microarray slide locations) are faulty across all microarray slides during the hybridization process. It is also hypothesized that entire microarray slides are faulty. Therefore, based on the *HomoIndex* step's pre-filter computations, invalid SNPs and entire microarray data sets are removed. Initial genotype mapping files are created using the remaining unfiltered mapping alleles, and the pipeline's data is rearranged to prepare subsequent pre-filter computations.

The *Dip Test* step executes the second set of pre-filter computations. A dip test is a statistical evaluation of whether a data distribution is bimodal. In other words, it determines whether a data distribution contains two groupings with a "dip" in-between (Hartigan et al., 1985). In terms of our genotype experiment, a SNP should be half heterozygous and half homozygous across all microarray data sets according to the biological structure of the experiment. Therefore, about one half of a SNP's HomoIndex values should be close to one and the other half should be close to zero across all microarray data sets (see *Fig. 4*). Dip tests determine whether each SNP's HomoIndex values fit the bimodal distribution, and the results are passed to the next pipeline step.

The *Final Genotype Mapping* step removes SNPs that do not meet the distribution described above. The removal acts as the final pipeline filter, and the pipeline concludes.

The final genotype mapping files contain the remaining unfiltered F2 mapping alleles at each unfiltered SNP, and the report files contain statistics about the pipeline's performance and results (refer to appendix E for detailed output specifications).

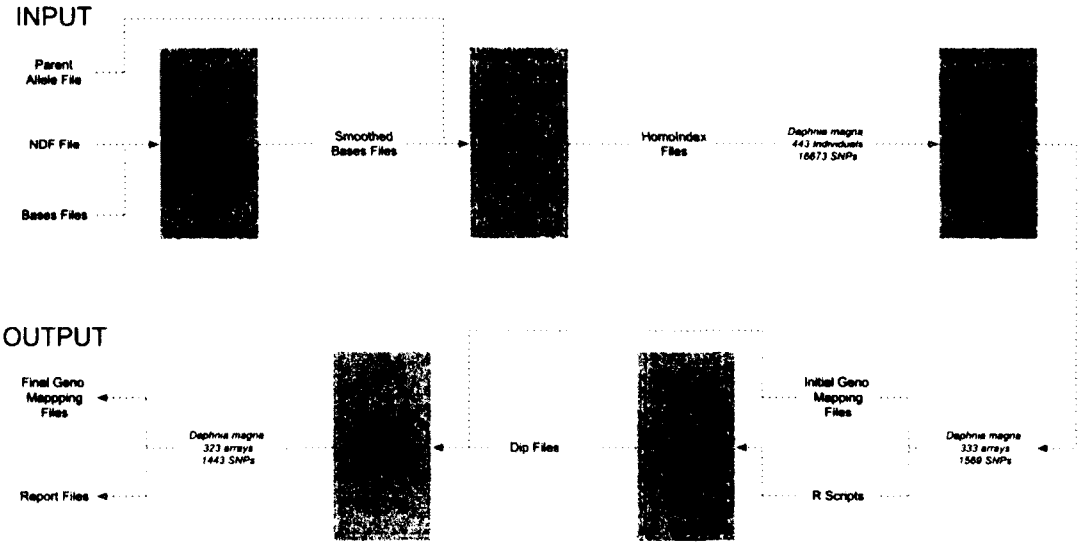


Figure 2 - The Pipeline Steps and Data Flow

The large boxes represent pipeline steps. The small boxes represent input files, output files, and data numbers for a default run (refer to the *Experimental Evaluation* section and appendix A for various run statistics). The arrows indicate the direction of data flow.

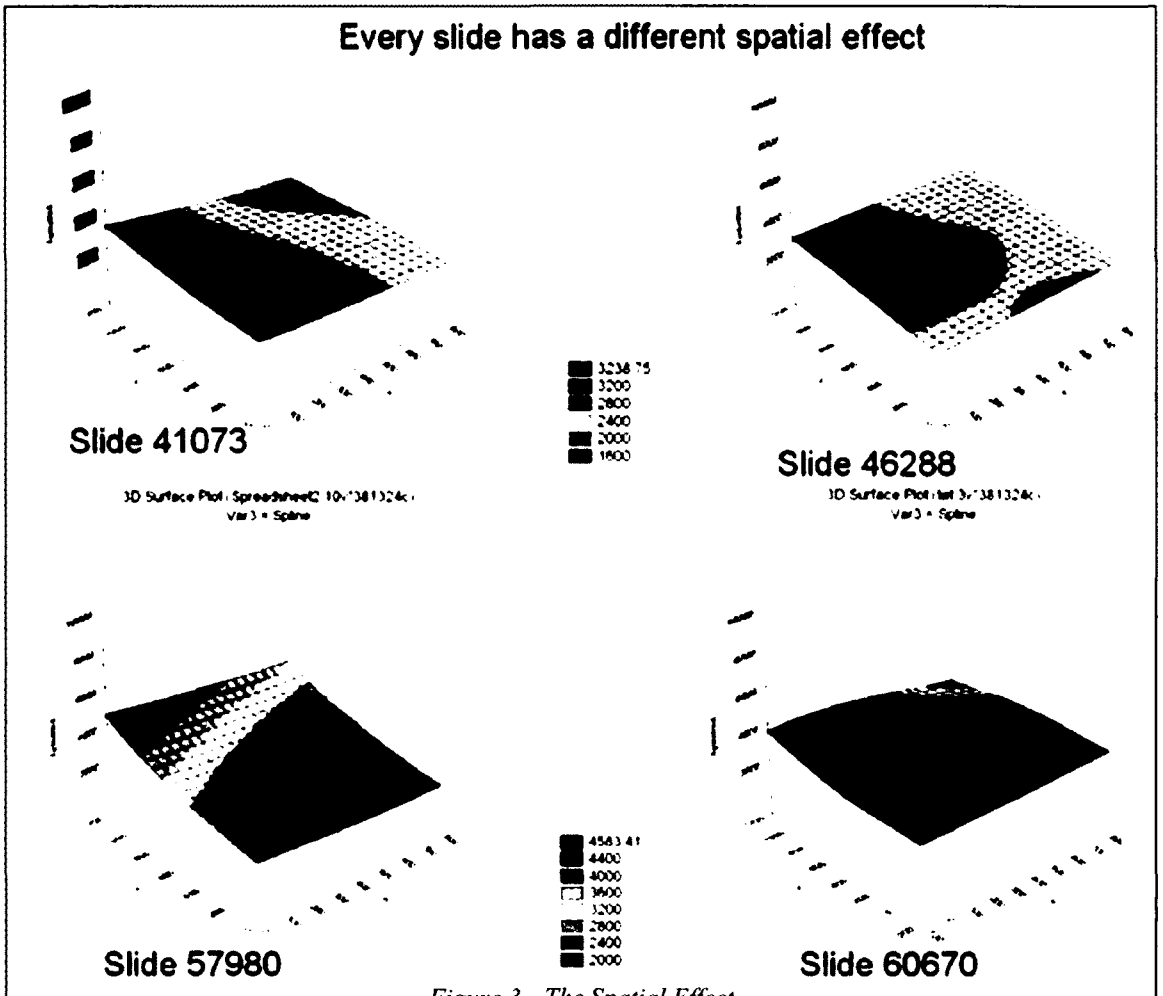


Figure 3 - The Spatial Effect

This figure was taken from Xiangfeng Wang and Hang He's website (Wang et. al., 2012). The illustration depicts the spatial effect on four NimbleGen microarray slides. The vertical axis represents signal intensity, and the colors represent similar intensity regions. The non-uniform distribution of the signal intensities is attributed to uneven washing of the DNA fragment solution over the slides or uneven scanning of the slides (refer to the *Array Hybridization Technique* section for more information). The MCDB Plant Genomics Group at Yale University gave permission to use this figure.

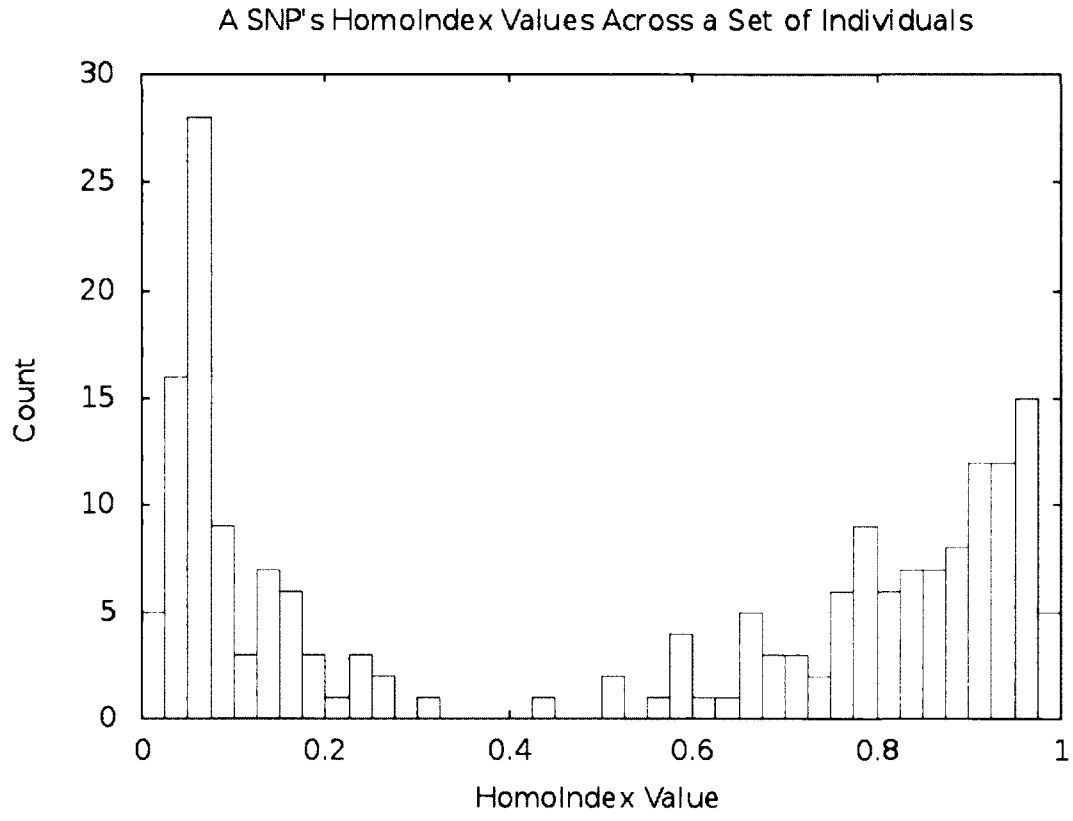


Figure 4 - A Bimodal SNP

The above figure was generated using the results of the *NAIndividuals Filter (.1)* run, detailed in appendix A. The plot is a histogram of an unfiltered SNP's HomoIndex values in the forward DNA strand across all unfiltered individuals. The distribution is obviously bimodal, and the dip test that analyzed these values generated a perfect p-value (refer to the *Dip Test* portion of the *Implementation* section for more information) of 0.

CHAPTER 3

IMPLEMENTATION

The pipeline's steps, parameters, inter-process communications, and data movements are all managed through a specific implementation structure. The pipeline executes in parallel using MPI (http://www.mcs.anl.gov/research/projects/mpich2/), and statistical elements are computed using R (http://www.r-project.org/).

3.1 Desired High Level Pipeline Qualities

The pipeline is designed to satisfy four qualities:

- The tool should be easy to customize and run. The target audience is non-technical users, and the interface should be accessible to them.
- Logging should be available at every part of the pipeline. The tool contains numerous steps and sub-processes, and each part should document information about the current pipeline run.
- The pipeline's algorithms should exploit the parallel aspects of the input data. The algorithms should be as efficient as possible.

- Errors should be elegantly handled due to the pipeline's numerous steps and sub-processes. Steps should not continue if a previous step has failed, and communication among the steps is crucial.

3.2 The Pipeline Parameters

The pipeline parameters are organized in two groups: *software parameters* and *user parameters*. The software parameters affect infrastructure elements such as locations of required executables, log files, and output directories. Pipeline administrators should manage the software parameters. The user parameters affect the pipeline's computations and results. Filter thresholds, input files, and output file formats are example user parameters. People using the pipeline for a genotype experiment should manage the user parameters. Refer to appendix C for detailed parameter specifications.

3.3 The Pipeline Structure

The pipeline's implementation uses Perl, C, MPI, R, and XML. It is designed for Linux clusters using a shared file system. A process known as the *Commander* manages the pipeline. The Commander parses the pipeline parameters, initializes and monitors the success of pipeline steps, and generates multiple output files. The Commander employs MPI to execute pipeline steps in parallel (*Fig. 5*).

Most of the pipeline data has obviously parallel components. Almost every pipeline step processes independent files:

- The *Spatial Smooth* step corrects the luminescent readings in each bases file, independent of other files.
- The *HomoIndex* step generates mapping alleles for each bases file, independent of other files.
- The *Dip Test* step computes the bimodality of each SNP's HomoIndex values independent of other SNPs.

Consequently, the pipeline is designed to process files in parallel. The Commander generates parallel instances of each pipeline step, and the instances are responsible for determining which files to handle. The files are evenly divided among instances, and each instance handles a different set of files. When a set of instances completes, the Commander begins the next pipeline step using the same parallel structure.

Note that some pipeline steps are not parallel processes. The Commander initiates these steps using the same parallel structure, but only one instance is created.

The Commander and the pipeline steps generate separate log files documenting information about each pipeline run. The parallel instances also generate status files to inform the Commander whether computations are successful. When a set of instances completes, the Commander examines the set's status files, and the pipeline terminates if any errors occur.

The Commander's final responsibility is to generate the pipeline's report files. Each parallel instance of each pipeline step passes statistics to the Commander through the associated status file. The Commander concatenates the statistics from every parallel

instance of every pipeline step into two report files: the *performance report file* and the *biology report file*. Refer to appendix E for detailed output specifications.

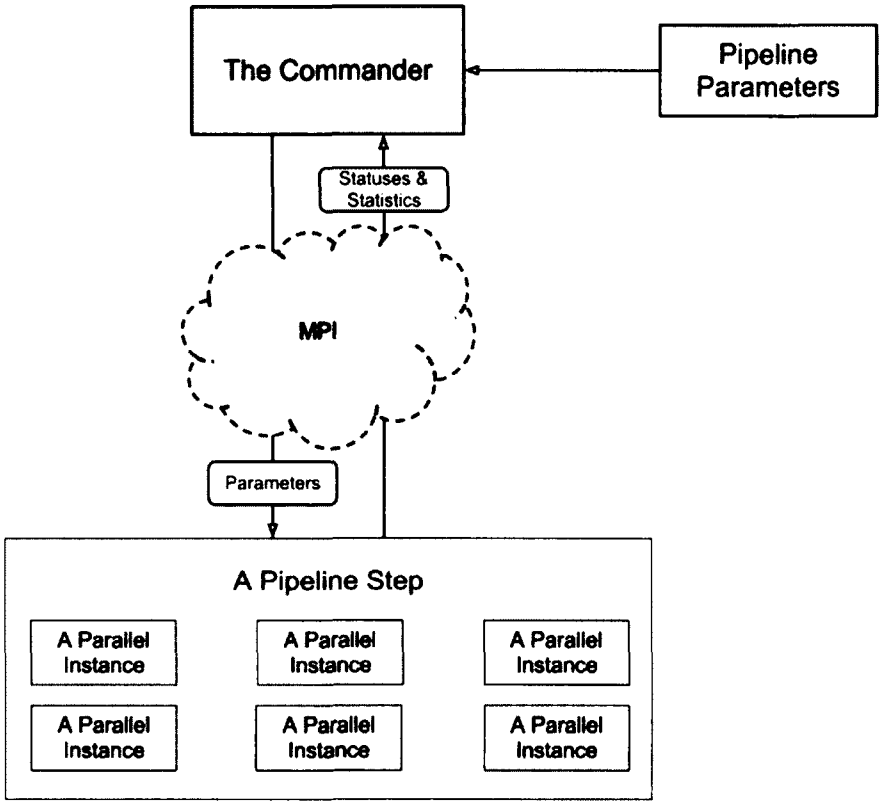


Figure 5 - The Parallel Pipeline Structure

The Commander uses MPI to generate multiple instances of each pipeline step. The instances process files in parallel. The Commander also manages the pipeline parameters, and appropriate parameters are passed to each pipeline step. When a set of instances completes, each instance passes its statistics and completion status back to the Commander. The Commander stores the statistics, and the next pipeline step begins if all the previous step's parallel instances completed successfully.

3.4 Quality Scores

The concept of quality scores must be discussed before describing the implementation of the pipeline steps. It is not obvious how to quantify the quality of a final genotype mapping file. One could argue that the final number of unfiltered SNPs and microarray data sets are an indication of the quality of a final genotype mapping file

(the higher the numbers, the better the quality). However, high numbers could indicate the pipeline filters were not stringent enough, and low numbers could indicate the pipeline filters were excessively stringent. Consequently, the pipeline generates three categories of quality scores to assist the assessment of a final genotype mapping. The quality scores are computed and utilized throughout multiple pipeline steps, and understanding them is necessary to understanding various pipeline calculations.

The first quality score is the *contig consistency score*. Since crossover events are rare when considering an entire genome and contigs tend to be relatively short sequences of DNA, mapping alleles should rarely be different within a single individual if the associated SNPs are located on the same contig. The pipeline uses this concept to generate the contig consistency score based on the final genotype mapping file. The computation steps through each individual and counts the number of contigs that contain one or more valid mapping alleles (mapping alleles that have been marked as invalid by one or more pre-filter or filter computation are ignored). For example, consider a single contig (contig "A") in two individuals (individuals "Y" and "Z"). Imagine the contig contains two SNPs. If the mapping alleles for both SNPs are valid in individuals Y and Z, the contig is counted twice. If the mapping allele for either SNP is invalid in individual Y, the contig will not be counted for individual Y. The same applies to the contig in individual Z. The contig consistency score is the percentage of the counted contigs that contain SNPs with the same mapping allele within their associated contig. A higher contig consistency score should indicate a better final genotype mapping. Note that a SNP identifier is composed of its containing contig identifier and its position on the

contig. The SNP identifiers are parsed to determine what SNPs are on what contigs (refer to appendix D for detailed input specifications).

Two additional statistics give context to the contig consistency score:

- The percent of the final unfiltered SNPs located on a contig with at least one other SNP. The contig consistency score better indicates the quality of a final genotype mapping if a high percentage of the unfiltered SNPs lay on contigs with at least one other SNP.
- The average distance (in nucleotides) between two consecutive SNPs on the same contig. As the average distance increases, the contig consistency score should decrease because the chances of crossover events that occur between SNPs will increase. The inverse should also be true.

The second quality score is actually a set of statistics called *HomoIndex quality scores* (HIQSs). Every mapping allele within each microarray data set has two associated HomoIndex values. In an ideal data set each HomoIndex value would either be zero or one. An ideal data set will never exist. Therefore, a HomoIndex value's quality score is its distance from zero or one depending on it being homozygous or heterozygous. If a HomoIndex value is not homozygous and it is not heterozygous (it falls between the maximum homozygous HomoIndex value and the minimum heterozygous HomoIndex value), its HomoIndex quality score is either its distance from zero if it is less than the average of the homozygous and heterozygous cutoff values, or its distance from 1 if it is greater than the average of the homozygous and heterozygous cutoff values. Lower

HIQSs indicate better HomoIndex values because the HomoIndex values are closer to one of the perfect values of zero or one. Every HIQS along with average HIQSs are output for a user to further manipulate (refer to appendix E for detailed output specifications). The extent to which HIQSs can be used to evaluate the quality of a final genotype mapping is not fully understood (refer to the *Future Research* section for more information), but the current implementation provides a basic measure of quality. Lower HIQSs overall should indicate a better final genotype mapping.

The third quality score is also a set of statistics. According to the experiment's biological structure, the mapping alleles across all SNPs and all individuals should be 25% homozygous from one parent clone, 25% homozygous from the other parent clone, and 50% heterozygous from both parent clones (see *Fig. 1*). Therefore, this set of statistics is the percent of each mapping allele out of the valid mapping alleles across all unfiltered SNPs and individuals. The closer the percentages are to 25%, 25%, and 50%, the better the final genotype mapping.

3.5 Spatial Smooth

The *Spatial Smooth* implementation is based on code developed by Xiangfeng Wang and Hang He (2006). The general algorithm is described below. Note that each luminescent reading consists of a mean and a standard deviation (refer to appendix D for detailed input specifications). Throughout this paper, any reference to a "reading" refers to its mean component.

The *Spatial Smooth* implementation can process each microarray data set independently; therefore, the *Spatial Smooth* step is a parallel process. The algorithm

divides a microarray slide into a grid of zones. The median luminescent reading of all the readings within a zone is computed for every zone. Every luminescent reading is compared to every zone's median luminescent reading using a distance-weighted intensity adjustment equation. Refer to the *Foreground Intensity Smoothing Equation* created by Wang and He. The equation increases or decreases each luminescent reading. The result is a smoothing effect across the entire microarray data set (see *Fig. 6*).

The adjustment of a mean is assumed to reflect an equivalent adjustment of each of the raw intensity values that contributed to the original mean. Consequently, each standard deviation value can be, and is, adjusted by the same percent as its associated mean value (refer to appendix D to understand the mean and standard deviation components of each luminescent reading).

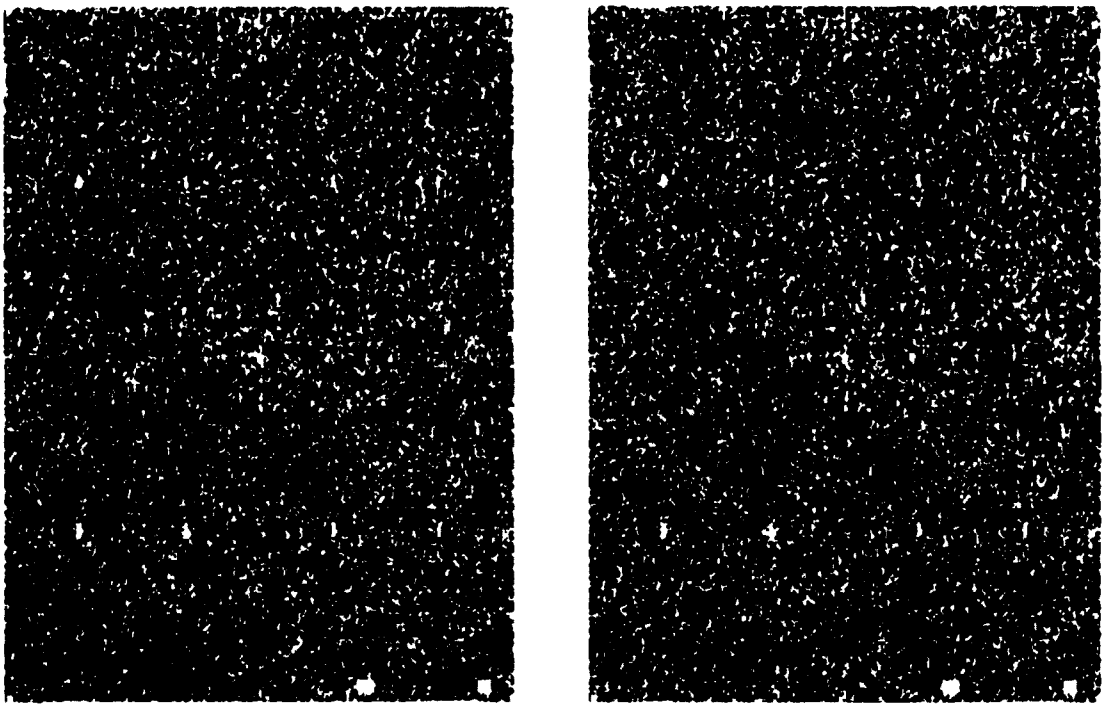


Figure 6 - The Spatial Smooth Effect

This figure was taken off Xiangfeng Wang and Hang He's website (Wang et. al., 2012). The illustration is a two-dimensional representation of the signal intensities on a microarray slide. One can easily see that the intensities in the top right corner of the original slide are different than the intensities in the bottom left corner of the original slide. The smoothed slide has a more even distribution of intensities. The MCDB Plant Genomics Group at Yale University gave permission to use this figure.

3.6 HomoIndex

The *HomoIndex* step computes the mapping alleles and applies pre-filter computations. Each bases file can be processed independently; therefore, the step is a parallel process. A bases file is processed by determining its SNPs' mapping alleles, and each SNP's mapping allele is determined by processing each of its two groups of four luminescent readings (four for the forward DNA strand, and four for the reverse DNA strand).

There are two possible expected distributions of a SNP's four strand-related luminescent readings:

- One reading is relatively high and the others are low. The readings are classified as homozygous. DNA fragments of the appropriate strand from both associated homologous chromosomes during the hybridization process tended to bind to one of the four oligos because the fragments contained the same SNP nucleotide. Hence, one oligo has a high luminescent reading while the others are low.
- Two readings are relatively high, and the others are low. The readings are classified as heterozygous. DNA fragments of the appropriate strand from one associated chromosome during the hybridization process tended to bind to one of the four oligos because the fragments contained the same SNP nucleotide. DNA fragments of the appropriate strand from the matching homologous chromosome tended to bind to a different oligo because the fragments contained a different SNP nucleotide. Hence, two oligos have high luminescent readings while the others are low.
- The readings may not match either distribution. The array hybridization process is not perfect, and DNA fragments can bind to incorrect oligos. The *HomoIndex* step is responsible for recognizing and marking invalid sets of readings.

HomoIndex values are calculated using the *HomoIndex equation* (refer to *Fig. 7).* Each HomoIndex value determines the distribution of four readings. The readings are sorted in decreasing order and fed to the HomoIndex equation.

$$h = \frac{r_2 - r_4}{r_1 - r_4}$$

Figure 7 - The HomoIndex Equation

Every HomoIndex value is computed based on four luminescent readings. The readings are sorted before they are inserted into the equation, r1 being the largest, and r4 being the smallest. h is the HomoIndex value.

The equation uses the following logic. If the distribution is homozygous, the difference between r2 and r4 is much less than the difference between r1 and r4; the value will be close to zero. If the distribution is heterozygous, the difference between r2 and r4 is slightly less than the difference between r1 and r4; the value will be close to one. Hence, each HomoIndex value classifies four readings as heterozygous or homozygous based on the value's vicinity to one or zero.

The HomoIndex equation does not correctly handle a specific distribution of readings. For example, imagine the following four readings: 100, 99, 98, 1. The readings should not be considered heterozygous or homozygous because there are three high readings rather than two or one. However, if the numbers are inserted into the HomoIndex equation, the HomoIndex value will be close to one and considered heterozygous. Therefore, each set of four readings with a heterozygous HomoIndex value is evaluated with a second equation that we call the heterozygous validation equation (refer to *Fig. 8*). If the value produced by this equation is near zero, the second and third readings are far apart, indicating that the original HomoIndex value is valid, and the associated mapping allele is computed using the original HomoIndex value. On the other hand, if this equation produces a value near one, the second and third readings are too close for the original HomoIndex value to be considered heterozygous, and the associated mapping allele is considered invalid.

$$h_2 = \frac{r_3 - r_4}{r_2 - r_4}$$

Figure 8 - The Heterozygous Validation Equation

The equation is used to reevaluate heterozygous HomoIndex values. The readings are sorted, r1 being the largest, and r4 being the smallest.

Two user parameters specify the maximum homozygous HomoIndex value and the minimum heterozygous HomoIndex value. The default values for these parameters are symmetrical around 0.5 with an invalid area between (refer to appendix C for detailed parameter specifications). The third set of quality scores described in the *Quality Scores* section may indicate that the default cutoff values are incorrect. If the percent of heterozygous mapping alleles is significantly below 50%, the range of heterozygous HomoIndex values may need to be larger than the range of homozygous HomoIndex values. The same concept can be applied to the range of homozygous HomoIndex values. Invalid distributions of the mapping alleles may be caused by the structure of various pipeline computations, the data generated by the array hybridization technique, or a multitude of unknown factors. Pipeline users should be prepared to alter these default parameters based on the third set of quality scores and the distribution of HomoIndex values generated by a pipeline run.

Each SNP has two heterozygous or homozygous classifications based on its two HomoIndex values. These classifications, combined with the eight associated luminescent readings, determine a SNP's nucleotides in both strands on both associated homologous chromosomes. If the classification for a group of four readings is homozygous, the oligo with the highest reading contains the SNP nucleotide on both chromosomes in the associated strand. If the classification for a group of four readings is

heterozygous, the oligo with the highest reading contains the SNP nucleotide on one chromosome in the associated strand, and the oligo with the second highest reading contains the SNP nucleotide on the matching homologous chromosome in the associated strand.

A SNP's mapping allele is calculated by comparing the SNP's nucleotides in both strands on both associated homologous chromosomes to the SNP nucleotides in the X and I clones. Refer to the pseudo-code in appendix B for the detailed comparison computation.

Multiple pre-filter computations designate potentially invalid data during the above calculations:

- There is a level of ambiguity in the HomoIndex equation. If either of a SNP's two HomoIndex values is not close to zero or one (two user parameters define the cutoff values for homozygous and heterozygous HomoIndex values), the SNP's mapping allele is marked invalid.
- The heterozygous validation equation can mark invalid mapping alleles.
- An optional pre-filter computation marks invalid mapping alleles based on HomoIndex quality scores (the *HomoIndex quality score pre-filter* computation). A user parameter enables the pre-filter computation, and a second user parameter specifies the maximum valid HIQS (refer to appendix C for detailed parameter specifications). Recall that lower HIQSs are better HIQSs. If a mapping allele has an excessively high HIQS, the mapping allele is marked invalid.

- If any errors occur during a SNP's nucleotide comparison computation with the X and I clones (refer to appendix B for the detailed comparison computation), the SNP's mapping allele is marked invalid.

A fifth pre-filter computation was implemented to correct a luminescent reading phenomenon specific to the data associated with our experiment. There are no parameters related to this pre-filter computation, and it will most likely be removed for future genotype experiments. We determined that other pre-filter computations and filters already remove the majority of the data marked invalid by this fifth pre-filter computation. Therefore, we concluded that the pre-filter computation has small effects on the pipeline's results, and the pre-filter computation is not evaluated in the *Experimental Evaluation* section. Nonetheless the pre-filter computation is implemented and has small effects on the results presented by this paper; therefore its workings and motivation are discussed in appendix F.

Subsequent pipeline steps remove invalid SNPs and entire microarray data sets based on the invalid mapping alleles. The HomoIndex values and mapping alleles are passed to the next pipeline step.

3.7 Filters, Initial Genotype Mapping, and Dip Preparation

The *FIGMDP* step removes invalid data, generates initial genotype mapping files, and prepares dip test calculations. The FIGMDP calculations are not microarray data set or SNP independent; therefore, the step is not a parallel process.

Five filters are applied in the following order:

- The *F1 Filter*: The genetic structure of the experiment suggests every SNP should be heterozygous in all F1 organisms. Therefore, SNPs with a non-heterozygous HomoIndex value in an F1 microarray data set are removed from all microarray data sets. Note that each SNP has two HomoIndex values in each F1 microarray data set (one for the forward DNA strand, and one for the reverse DNA strand). Consequently, there is a user parameter associated with the F1 filter called the OR/AND parameter. If the parameter is set to OR, a SNP is removed if it has a non-heterozygous HomoIndex value in any F1 microarray data set in either strand. If the parameter is set to AND, a SNP is only removed if it has a non-heterozygous HomoIndex value in any F1 microarray data set in both strands.
- The *Parent Filter*: The genetic structure of the experiment also suggests every SNP should be the appropriate homozygous mapping allele in the X and I clones. Therefore, SNPs that are not the appropriate homozygous mapping allele in the X and I clone microarray data sets are removed from all microarray data sets. Note that there is also an OR/AND user parameter associated with this filter. If the parameter is set to OR, a SNP is removed if it has an incorrect mapping allele in either parent clone. If the parameter is set to AND, a SNP is only removed if it has an incorrect mapping allele in both parent clones.
- The *NA SNPs Filter*: Every microarray slide contains the same oligos in the same locations. It is hypothesized that certain oligos (microarray slide locations) are

faulty across all microarray slides. Therefore, SNPs with a high percentage of invalid mapping alleles across all microarray data sets are removed from all microarray data sets. A user parameter defines the invalid percentage that is considered too high.

- The *NA Individuals Filter*: It is also hypothesized that entire microarray slides are faulty. Therefore, microarray data sets with a high percentage of invalid mapping alleles across all SNPs are removed from the pipeline's data. A user parameter defines the invalid percentage that is considered too high. This filter executes last because the previous filters affect the total number of SNPs. Therefore, the previous filters affect the percentages examined by this filter. In a similar manner, the designation of entire microarray data sets as invalid and the removal of all SNP information associated with those data sets raise questions about the consistency and uniformity of the previous filters' results. The invalid mapping alleles in the deleted microarray data sets counted as invalid in the previous filters, but now they are no longer in the data; similarly, valid mapping alleles in the deleted microarray data sets are also no longer in the data. Consequently, we re-execute the previous filters on the remaining microarray data sets in an attempt to remove SNPs based on only the best microarray data sets. The filters execute in the same order, and the filters reevaluate every SNP, including the previously removed SNPs.
- The *Duplicates Filter*: The duplicates filter executes in conjunction with the NA individuals filter. Many times there are multiple microarray slides generated by different DNA samples from the same organism. Usually this is because a

microarray slide was generated using a bad DNA sample, so a second slide was generated using a different DNA sample in an attempt to correct errors. A second motivation is simply to replicate data and replicate results. The pipeline processes the data from both slides, but the pipeline removes the microarray data set with the higher percentage of invalid mapping alleles. There can be any number of duplicate individuals, but the only microarray data set that passes the duplicates filter is the one with the lowest percentage of invalid mapping alleles. The invalid individuals are removed at the same time as the individuals removed by the NA individuals filter, and if either the duplicates filter or the NA individuals filter is enabled, the first three filters will re-execute.

After the filters complete, the initial genotype mapping file is generated, and the pipeline's data is rearranged in preparation for the next pipeline step. Previously, the data has been arranged by microarray data set (the bases files). The next pipeline step requires the data be arranged by SNP. Therefore, each SNP's HomoIndex values across all microarray data sets are grouped in separate files (R scripts). The next pipeline step uses these files as input.

3.8 Dip Test

The *Dip Test* step executes two dip tests per SNP. One test analyzes the SNP's HomoIndex values in the forward DNA strand across all microarray data sets. The second test analyzes the SNP's HomoIndex values in the reverse DNA strand across all microarray data sets. The dip test results are the pre-filter computations used by the next

pipeline step. Each dip test can be executed independently; therefore, the step is a parallel process.

The dip tests execute in R using the "diptest" package (Maechler, 2012), and each test ignores HomoIndex values associated with invalid mapping alleles. Note that each test also ignores HomoIndex values from F1, X and I clone microarray data sets because the expected bimodal distribution only applies to HomoIndex values across the F2 individuals (see *Fig. 1*). Each dip test generates a *p-value* indicating the strength of the bimodal distribution. The p-values range from zero to one, and lower p-values indicate more distinct bimodal distributions (Maechler, 2012). The p-values are passed to the next pipeline step.

3.9 Final Genotype Mapping

The *Final Genotype Mapping* step applies the final pipeline filter by removing SNPs that do not fit the bimodal distribution. The filter computations are not microarray data set or SNP independent; therefore the step is not a parallel process.

- The *P-Value Filter*: Each SNP has two associated p-values, one for the forward DNA strand and one for the reverse DNA strand. If a SNP's p-values are excessively high it is removed from all microarray data sets. A user parameter defines the maximum p-value. Note that there is also an OR/AND user parameter associated with the p-value filter. If the parameter is set to OR, a SNP is removed if either of its p-values are excessively high. If the parameter is set to AND, a SNP is only removed if both of its p-values are excessively high.

After the filter executes, the initial genotype mapping files are altered accordingly, and the pipeline terminates by generating the output files (refer to appendix E for detailed output specifications). Note that a list of individuals can be provided to the *Final Genotype Mapping* step that designates individuals that should always be removed from the final genotype mapping files. In this experiment, the F1 individuals and the X and I clones are always removed because the experiment's focus is the F2 individuals.

3.10 Nucleotide Mapping Alleles

Late in the development of the pipeline a capability was added to generate a different type of mapping alleles. The motivation behind the capability was a set of NimbleGen microarray data sets that was unrelated to the experiment discussed by this paper. The data sets were not generated using the X and I clone, F1, F2 biological structure; rather, they were simply a set of individuals with luminescent readings for a set of SNPs. The goal was to calculate the nucleotides at each SNP within each individual. Therefore, a user parameter was added that changes the output of the mapping alleles. Rather than being "A", "B", "H", or "NA" (refer to the *Experiment* section for more information on the parent mapping alleles), the mapping alleles are one nucleotide ("A", "C", "T", or "G") if the SNP is homozygous, and two nucleotides if the SNP is heterozygous. These mapping alleles indicate a SNP's nucleotide, or nucleotides, on the associated pair of homologous chromosomes in the forward DNA strand.

When this capability is enabled, the F1 filter and the parent filter are automatically disabled because they are based on logic that only applies to the experiment discussed by this paper. All other user parameters are still customizable.

Note, however, that the p-value filter should most likely be disabled because it is based on the notion that each SNP should be 50% homozygous and 50% heterozygous across all F2 individuals. The assumption will not be valid for many experiments.

This capability has not been heavily used or evaluated. It is not discussed in the *Experimental Evaluation* section of this paper, but the capability is implemented and ready for use.

CHAPTER 4

EXPERIMENTAL EVALUATION

The pipeline was evaluated under various configurations. There are a large number of customizable user parameters, and the experimental evaluations are designed to provide insight into the effects of those parameters. The parameter values and the biological statistics associated with the pipeline's output were recorded for 24 pipeline runs that processed the same input data. The pipeline's input consisted of 352 unique individuals and 91 duplicate individuals. Each individual contained 16673 SNPs. The results of all pipeline runs are summarized in *Table 1* of appendix A, and the parameters of the runs are summarized in *Table 2* of appendix A. The following sections discuss the highlights of those runs.

The pipeline's performance was not evaluated. The pipeline only needs to generate results in a relatively short amount of time in order for the subsequent biological studies to progress. Therefore, great effort was not put into optimizing the performance of the algorithms. However, the pipeline's run time was greatly improved by parallelizing the computation of separate files. The pipeline was tested on a machine with 12 cores. Each file that is processed during each parallel step requires nearly the same amount of processing power, therefore the run time of the pipeline was improved by roughly a factor of 12. The parallelization allows pipeline runs to finish in a matter of

minutes rather than hours, and it is essential to the pipeline's implementation. One parameter, or filter, that is not mentioned in the following sections is the duplicates filter. The duplicates filter was enabled for every pipeline run, except the *NAIndividuals Filter Off* run.

4.1 Default Parameter Run

The *Defaults* run gives a general sense of the pipeline's typical results. A quick observation that applies to most of the pipeline runs is that a large portion of the input data is filtered out. The final number of SNPs in the *Defaults* run drops from 16673 to 1443, and the final number of individuals drops from 444 to 323. This, however, is acceptable because the numbers are sufficient for the subsequent genetic mapping and QTL study to progress.

A positive result of the *Defaults* run is shown by the differences between *Fig. 9* and *Fig. 10.* Both figures plot the HomoIndex values across all SNPs and all individuals, one before the filters are applied, and the other after the filters are applied. A large focus of the pipeline's computations is to remove SNPs that are not 50% homozygous and 50% heterozygous across all individuals. Therefore, the bimodal improvement from one plot to the next is evidence that the pipeline's filters are working. However, the final distribution is not perfectly bimodal, and whether that poses a problem is not yet known.

Two of the *Defaults* run's quality scores imply the results are relatively good. The distribution of the mapping alleles across the final SNPs and individuals is close to the desired results. The mapping alleles are 47% "H" (the desired percentage is 50%), 27% "B" (the desired percentage is 25%), and 25% "A" (the desired percentage is 25%).

Also, 17.5% of the final SNPs are located on a contig with at least one other SNP. Out of those contigs, 95% of them contain consistent SNPs. This coincides with the notion of rare crossover events, which causes SNPs on the same contig to have the same mapping allele most of the time.

As for the third quality measure, the HomoIndex quality scores, it is not obvious what those values should be. *Fig. 11* illustrates the distribution of the final HIQSs in the *Defaults* run. The homozygous and heterozygous cutoff values are 0.475 from zero and one respectively, and the majority of the HIQSs seem far less than 0.475. In fact the majority of the HIQSs may even be less than half of 0.475. Does that indicate the results are of high quality? It is unsure. *Fig. 12* compares the HIQS distributions from multiple pipeline runs. One would expect the shape of the distribution to shift left if a run was worse than the *Defaults* run, and one would expect the shape of the distribution to shift right if a run was better than the *Defaults* run. The plot has different counts for the various runs because the runs generated different final numbers of SNPs and individuals, but the general shapes are the same. The plot illustrates only a subset of the evaluation runs, but it is interesting that there are not more evident differences. The runs vary in their final number of SNPs, final number of individuals, and quality scores, but interestingly they do not vary in their HIQS distributions. The plot may suggest that HomoIndex quality score distributions are a poor measure of quality, or it may suggest that the plotted runs are of similar quality. This requires further investigation.

Moving away from the HIQS distribution, the *Default* run's mean HIQSs show an interesting trend. The mean "H" HIQS is much greater than the mean "A" and "B" HIQSs in almost every pipeline run. The mean "A" and "B" HIQSs are usually very

similar, and the mean "H" HIQS is much higher. This trend is supported by the *Defaults* run's unfiltered HomoIndex values and filtered HomoIndex values, presented in *Fig. 9* and *Fig. 10*. The homozygous HomoIndex values are more tightly clustered in both plots. The homozygous HomoIndex values also seem closer to zero than the heterozygous HomoIndex values are to one. This is consistent with the mean HomoIndex quality scores, but the cause is unknown. Is the HomoIndex equation not suited to identify heterozygous SNPs? Is the array hybridization technique generating erroneous data for heterozygous SNPs? Should the homozygous HomoIndex values be more spread out? The fact that the initial HomoIndex values are heavily weighted towards homozygous cases is also a point of interest, and all of these questions should definitely be sources of further investigation. The mean HIQSs and the HIQS distribution may be more helpful when comparing pipeline runs.

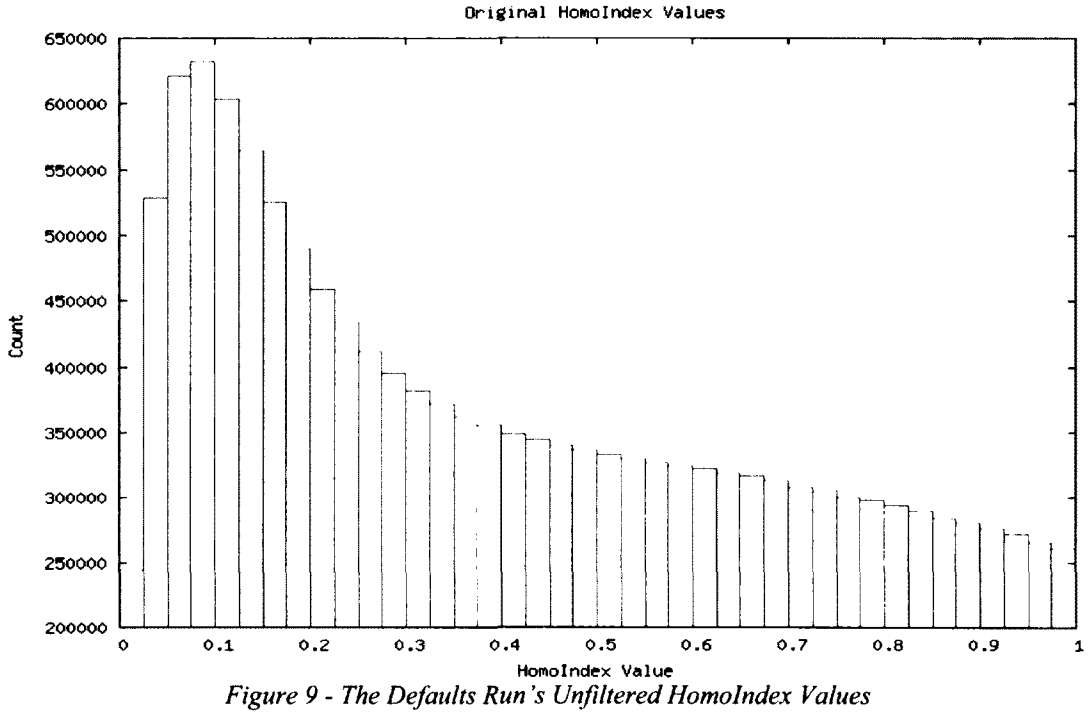


Figure 9 - The Defaults Run's Unfiltered HomoIndex Values

The plot is a histogram of all the *Defaults* run's HomoIndex values across all individuals and SNPs before any filters are applied. Refer to appendix A for details about the *Defaults* run's results.

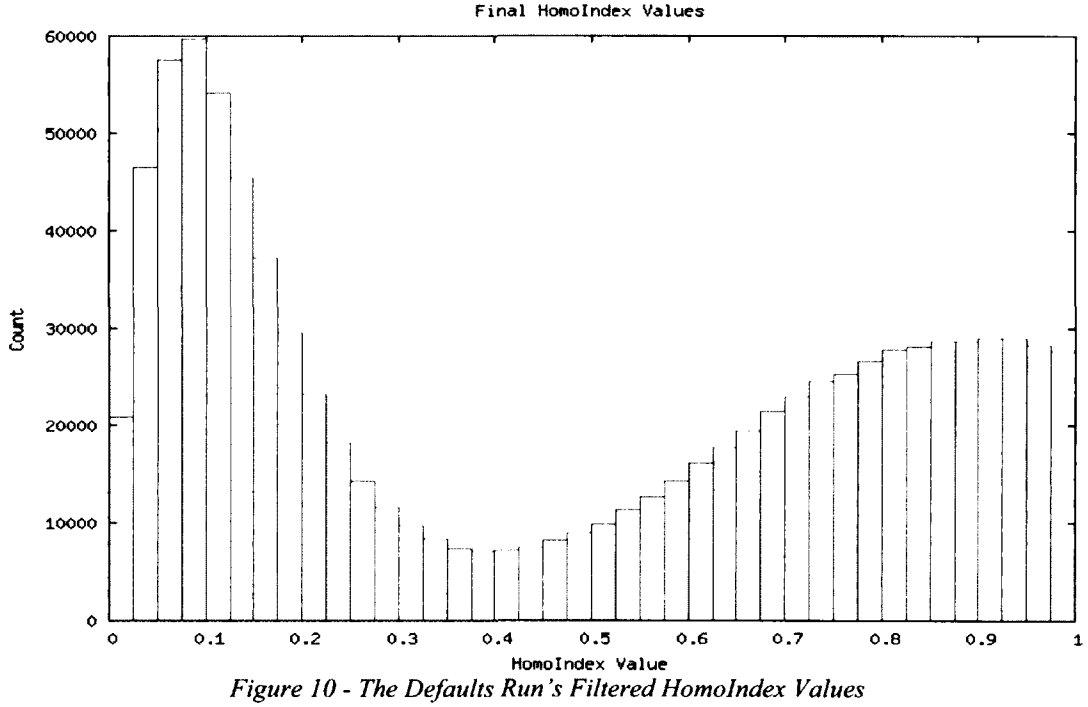


Figure 10 - The Defaults Run's Filtered HomoIndex Values

The plot is a histogram of all the *Defaults* run's HomoIndex values across all unfiltered individuals and SNPs after the pipeline filters have been applied. Refer to appendix A for details about the *Defaults* run's results.

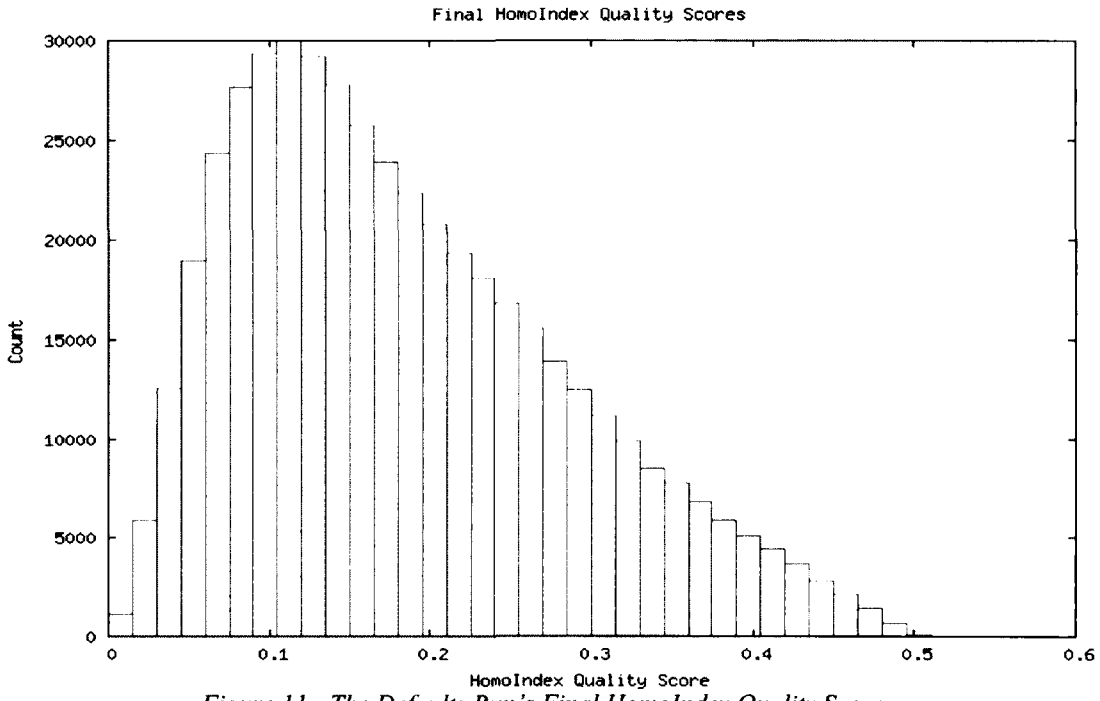


Figure 11 - The Defaults Run's Final HomoIndex Quality Scores

The plot is a histogram of all the *Defaults* run's HomoIndex quality scores across all individuals and SNPs after the pipeline has executed. Refer to appendix A for details about the *Defaults* run's results.

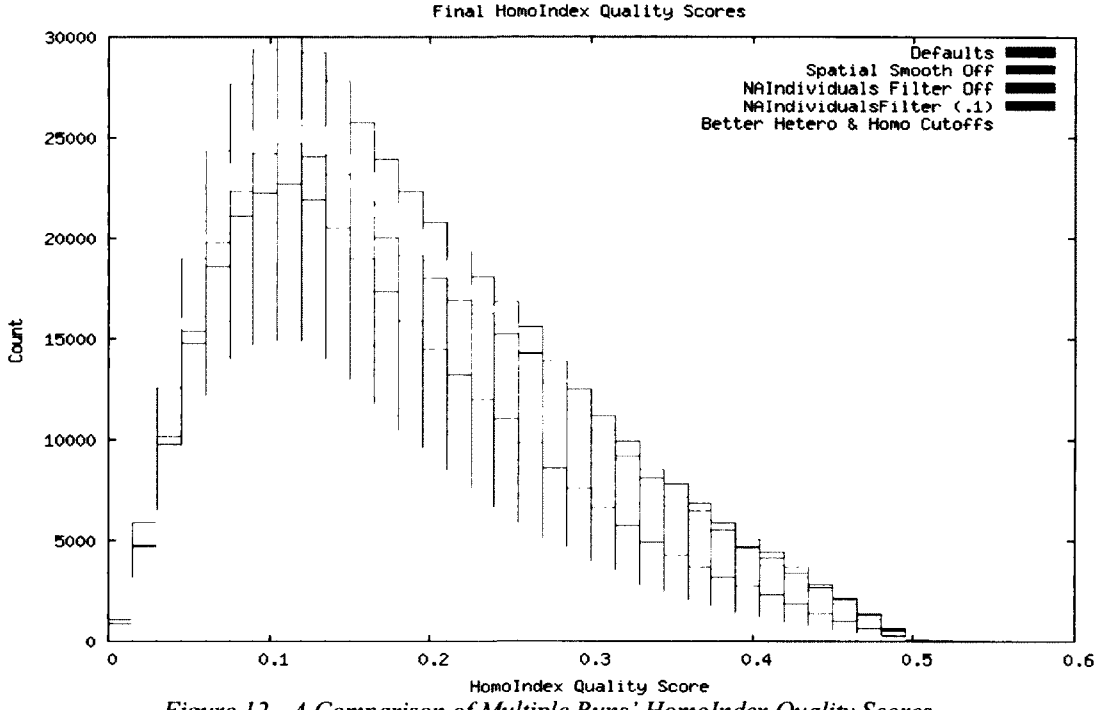


Figure 12 - A Comparison of Multiple Runs' HomoIndex Quality Scores

The plot is a histogram of multiple runs' HomoIndex quality scores. Refer to appendix A for details about the various runs' results.

4.2 Spatial Smooth Off Run

The *SpatialSmooth Off* run attempted to evaluate the effectiveness of the *Spatial Smooth* step. The *Spatial Smooth* step was disabled, and we hypothesized that the results would be worse when compared to the *Defaults* run. The final number of SNPs supports the hypothesis because the number decreased by 166. Surprisingly, the mapping allele distribution and the contig consistency score slightly improved, but the mean HomoIndex quality scores slightly worsened. Overall, the differences in the quality scores between this run and the *Defaults* run are relatively small. Therefore, it seems like the overall quality of the pipeline's results are not greatly affected by the *Spatial Smooth* step (according to the implemented quality scores). However, the final number of SNPs increases by 166 when the *Spatial Smooth* step is enabled. Whether or not that number is significant is uncertain, but more SNPs is usually preferable over fewer SNPs. Further investigation involving a wider range of pipeline configurations with the *Spatial Smooth* step enabled and disabled is required in order to truly evaluate the effectiveness of the *Spatial Smooth* step.

4.3 F1 Filter Off Run

In the *Defaults* run the F1 filter removes most of the SNPs (14557 SNPs out of the initial 16673). Therefore, if a large final number of SNPs is desired, it might make sense to disable the F1 filter. However, this may significantly decrease the quality of the final genotype mapping file. The *F1 Filter Off* run was aimed at evaluating these effects.

A key observation is that a large majority of the SNPs usually removed by the F1 filter are also removed by other filters. The NA SNPs filter removes 11697 of the 16673 SNPs, and the p-value filter removes 2578 SNPs. However, the final number of SNPs is still 955 more than the *Defaults* run. All three of the quality measures are worse, which may indicate that the 955 new SNPs are of poor quality. These results support the hypothesis because the final number of SNPs increased at the cost of lower quality. The quality scores are not drastically worse than the *Defaults* run, so it may be acceptable to disable the F1 filter if a large final number of SNPs is desired.

4.4 P-Value Filter Off Run

A hypothesis similar to the *F1 Filter Off* run was proposed for the *PValue Filter Off* run: will disabling the p-value filter increase the final number of SNPs at the cost of lower quality? The final number of SNPs increased slightly, but not as drastically as the *F1 Filter Off* run. The quality scores went in multiple directions. The mapping allele distribution slightly improved, the mean HIQSs slightly worsened, and the contig consistency score stayed the same.

These small effects are most likely the result of the p-value filter executing last. Most of the invalid SNPs have been removed before the p-value filter executes. Therefore, the p-value filter cannot possibly have the same effect as the F1 filter because the filters process very different amounts of data. The *PValue (.2,OR) & NASNPs Filters Only* run attempted to investigate this notion by disabling all the filters except the p-value filter. Unfortunately, the NA SNPs filter had to remain enabled because some SNPs have invalid mapping alleles across all individuals, and the dip tests required by the p-value

filter cannot process SNPs without at least one valid HomoIndex value. The number of SNPs removed by the p-value filter did increase from 126 to 2454 when compared to the *Defaults* run. Also, the number of SNPs removed by the p-value filter in the *F1 Filter Off* run increased from 126 to 2578 when compared to the *Defaults* run. These runs support the hypothesis that the p-value filter is important and would have a greater effect if it somehow executed before the other filters. However further investigation is required in order to confirm this notion.

4.5 NA Individuals Filter Off Run

Logically, the NA individuals filter seems like an important pipeline process. Remember that multiple SNP filters re-execute after the NA individuals filter completes so that the SNP filters only base their computations on data from the best individuals. Therefore, we hypothesized that disabling the NA Individuals filter may have large effects. Note that the *NAIndividuals Filter Off* run was the only run that also disabled the duplicates filter in order to get a sense of the pipeline's results when zero individuals are removed (also note that the F1, X and I clone individuals were still removed, making the final number of individuals 433 rather than 443).

The most obvious observation is the decrease in the final number of SNPs in comparison to the *Defaults* run. This supports the notion that many SNPs are removed based on data from invalid individuals when invalid individuals are not removed. However, once again the quality scores did not show drastic changes.

Logically this seems like a filter that should never be disabled, and the initial results indicate it significantly affects the final number of SNPs. This is further

supported by the *NAIndividuals Filter (.1)* run, where the cutoff percentage designating valid or invalid individuals was changed from 25% to 10%. The final number of SNPs increased in comparison to the *Defaults* run, and the mapping allele distribution and the mean HIQSs also improved (the contig consistency score remained the same). There is likely a limit to how low the cutoff percentage can be and still generate positive results. If an immense number of individuals are removed by the NA individuals filter, a SNP with just a few invalid mapping alleles across all individuals will be removed, and the filter may become counterproductive. Once again, this requires further investigation.

4.6 Better Hetero & Homo Cutoffs Run

Recall that the default maximum homozygous HomoIndex value and the default minimum heterozygous HomoIndex value are symmetrical around 0.5. Each default cutoff value is 0.025 away from 0.5 in the appropriate direction. When *Fig. 10* is examined, it seems logical that these cutoff values should be altered. Homozygous HomoIndex values seem to cluster below 0.3, and heterozygous HomoIndex values seem to cluster above 0.6. This may indicate that in the *Defaults* run, too many HomoIndex values are considered homozygous and too few HomoIndex values are considered heterozygous. This is supported by the mapping allele distribution of the *Defaults* run for which the heterozygous percentage is slightly less than 50%, and the homozygous percentage is slightly greater than 50%.

Consequently, the *Better Hetero & Homo Cutoffs* run set the homozygous cutoff value to 0.3 and the heterozygous cutoff value to 0.6. Logically, this should greatly increase the number of invalid HomoIndex values, which is supported by the significant

drops in the final numbers of SNPs and individuals when compared to the *Defaults* run. All three quality scores improved, but like many other runs, this should be taken with a grain of salt. The mapping allele distribution moved closer to the desired 25%, 25%, 50% distribution, but it is not obvious whether the improvement is significant. Take, for example, the *PValue Filter Off* run. Its mapping allele distribution was even better than the *Better Hetero & Homo Cutoffs* run, which is counter-intuitive since one of the seemingly important filters was disabled! The mean HIQS numbers improved in the *Better Hetero & Homo Cutoffs* run, but this is likely due to poor HIQSs being removed because their associated HomoIndex values were marked as invalid since they were between the new, stricter, homozygous and heterozygous cutoff values. The contig consistency score also improved, but the percentage of SNPs that share a contig decreased (indicating the contig consistency score is a less indicative measure of quality, refer to the *Quality Scores* section for more information), and the mean gap between consecutive SNPs decreased (indicating the contig consistency score should improve automatically, refer to the *Quality Scores* section for more information).

Overall, it is difficult to conclude whether the altered homozygous and heterozygous cutoff values generated improved results. They did improve the mapping allele distribution percentages, which was the original focus of the run. However, if *Fig. 13* is examined, there is not a large difference between the shapes of the distributions. This is most likely explained by the fact that there was not much room for improvement in the *Defaults* run in the first place. The only possibly significant observation is that the low dip between the homozygous and heterozygous HomoIndex values in the *Better*

Hetero & Homo Cutoffs run is wider than in the *Defaults* run, which is consistent with the altered homozygous and heterozygous cutoff values that are farther apart.

Pipeline users should alter the homozygous and heterozygous cutoff values based on analyzing the distributions of HomoIndex values and mapping alleles. It is not obvious why the distribution of HomoIndex values is not symmetrical around 0.5. This may be a product of the pipeline's input data, the HomoIndex equation, or other unknown factors. This issue deserves further investigation.

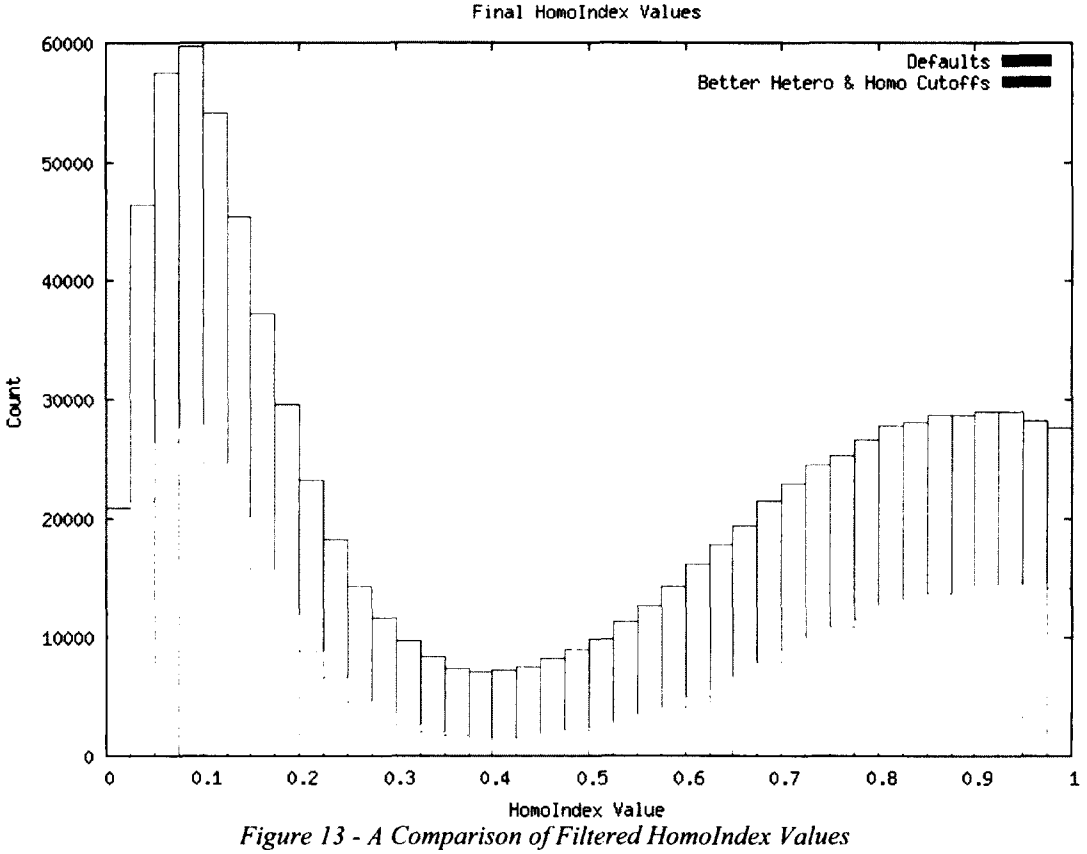


Figure 13 - A Comparison of Filtered HomoIndex Values

The plot is a histogram of all the *Defaults* run and *Better Hetero & Homo Cutoffs* run's HomoIndex values across all unfiltered individuals and SNPs after the pipeline filters have been applied. Refer to appendix A for details about the runs' results.

4.7 The Parent Filter and the HomoIndex Quality Score Pre-Filter

The parent filter and the HomoIndex quality score pre-filter computation are not enabled by default because the features were added late in the pipeline's development. The features were not used to generate the genotype data used by the genetic mapping and QTL studies, and therefore the features have not been heavily evaluated. However, pipeline runs were conducted in an attempt to understand the features' effects.

The Parent Filter Only runs did not generate good results. The parent filter was the only enabled filter, and even though it removed a significant number of SNPs, the quality scores were poor. If used in conjunction with other filters, the parent filter might improve results. The only other run that enabled the parent filter was the *All Pre-Filters & Filters* run, which enabled every pre-filter computation and every filter. The results were even worse because every SNP was removed by one or more filters. Therefore zero genotypes were generated, and it is clear that the run's parameters were too stringent. The parent filter still may produce positive results if used in conjunction with some of the other filters, but this requires further investigation.

The HomoIndex quality score pre-filter in conjunction with the NA SNPs filter also produced negative results (the *HIQS Pre-Filter & NASNPs Filter Only* runs). After implementing the HIQS pre-filter and executing the associated runs, we realized that the pre-filter is redundant. Enabling the HIQS pre-filter with a cutoff value of .25 is equivalent to having symmetrical homozygous and heterozygous cutoff values around 0.5 that are 0.25 from zero and one respectively. Consequently, it is reasonable to conclude that the HIQS pre-filter is not necessary because the same effects can be generated by

altering the homozygous and heterozygous cutoff values. The negative results also suggest that filters that take into account key biological factors (like the F1 filter and the p-value filter) should always be enabled. This is further supported by negative results in the *NASNPs Filter Only* runs.

4.8 OR vs. AND Runs

Three SNP filters have OR/AND options: the F1 filter, the parent filter, and the p-value filter. These have been previously described, but essentially each filter examines two associated values for each SNP in order to determine whether the SNP is valid. If the OR option is enabled, a SNP is removed if either of its associated values is invalid; if the AND option is enabled, a SNP is only removed if both of its associated values are invalid.

Each of these filters was tested with both options. In each case the AND options removed fewer SNPs at the cost of lower quality scores, and the OR option removed more SNPs with the benefit of better quality scores. This is the expected result. The effect of the options on the individual filters is not discussed, and once again requires further investigation.

4.9 Invalid Runs

Three of the pipeline runs did not generate results: the *NAIndividuals Filter Only* runs, and the *All Pre-Filters & Filters* run. The NA individuals filter removed all the individuals during the *NAIndividuals Filter Only* runs. This suggests there is a significant

amount of invalid data across all the SNPs within each individual, and the invalid SNPs must be removed before the NA individuals filter executes. The F1 filter, parent filter, and NA SNPs filter removed all the SNPs before the NA individuals filter executed in the *All Pre-Filters & Filters* run. This suggests that too many filters can be counterproductive. Pipeline users should take these cases into account when customizing their pipeline runs.

CHAPTER 5

CONCLUSIONS

The parallel data pipeline introduced by this thesis has significantly improved genotyping capabilities. By interfacing with NimbleGen's array hybridization technique, the pipeline can genotype a larger number of loci and individuals than previous genotyping methods. The pipeline's parallel processing capabilities make its computations quick and practical, and the pipeline's parameters make it an easily customizable tool.

The large number of customizable pipeline parameters is a double-edged sword. The ability to customize every pipeline computation can lead to better results. However, the number of possible pipeline configurations only increases with the number of pipeline parameters. Determining the optimal configuration for a specific set of input data will never be an easy task. Pipeline users will most likely have to use the default parameters and execute trial and error runs to determine the best configuration for their data sets.

The pipeline has generated genotype data that has allowed a genetic mapping and QTL study to commence. Unfortunately, analyzing the quality of that data and the semantics of its numbers has proven difficult. On one hand, the majority of the runs presented in the *Experimental Evaluation* section generated relatively good quality scores. The mapping allele distributions were nearly 25%, 25%, and 50%, the mean

HomoIndex quality scores were less than half of their maximum, and the contig consistency scores were above 90%. Does this mean the pipeline is generating high quality results? It is uncertain for the following reason:

- Whether or not the results are of high quality may depend on the results of the subsequent studies. If the genetic mapping and QTL studies are extremely successful, it may be safe to assume that the pipeline is generating high quality data. Also, the pipeline has not processed any data sets from other organisms. Once the pipeline has been utilized by other genotype experiments, the quality of its computations may become more obvious.

Nonetheless, the pipeline implementation provides a substantial code base. As more experiments are conducted, further investigation will shine light onto the quality of the results. The pipeline may be altered to incorporate new filters, capabilities and statistics, but the current implementation provides a new way to genotype large numbers of loci across large numbers of individuals.

CHAPTER 6

FUTURE RESEARCH

The experiment and the pipeline outlined by this paper lay the foundation for faster and larger genotype experiments. However, there is still research that could improve the pipeline and provide insight into the semantics of its results.

- As the genetic mapping and QTL studies progress, they will hopefully become more streamlined. Consequently it will be easier to conduct multiple studies in a short amount of time. This will allow researchers to compare the effects of the pipeline's parameters on subsequent genetic mapping and QTL studies. Do the studies prefer more SNPs or more individuals? Do the studies prefer a small amount of very high quality data? Do the studies prefer something else entirely?
- Why does the HomoIndex equation generate a larger number of more tightly clustered homozygous HomoIndex values as compared to the heterozygous HomoIndex values? Should the HomoIndex equation be altered, or should an additional pre-filter computation correct a bias caused by the array hybridization technique?
- What is causing the "mean vs. standard deviation" relationship in the luminescent readings generated by the array hybridization technique (refer to appendix F)?

- It would be informative to rigorously evaluate the performance of the pipeline's algorithms. How much memory are they using, and what are their time complexities? Is there a better way to parallelize the computations? Should a dynamic load balancing implementation be considered?
- How can the pipeline become more publicly accessible? Should it be installed on a public cluster? Should a web-based graphical user interface be implemented?
- How does the pipeline perform on other data sets? Do other organisms generate positive results? How effective is the nucleotide mapping allele capability of the pipeline (refer to the *Nucleotide Mapping Allele* section)?
- In general, the semantics of the pipeline's results deserve further investigation. The preliminary results presented in the *Experimental Evaluation* section most likely contain additional meaning not yet discovered. Almost any pipeline computation could benefit from further evaluations, and additional tests may suggest computations to remove and computations to add. The HomoIndex quality scores can be manipulated in countless more ways. What is the distribution of the HomoIndex quality scores before any filters are applied? What are the distributions of the HomoIndex quality scores within removed individuals or SNPs?

Overall, the pipeline is an initial version, and it will most likely be improved and better understood in future iterations.

BIBLIOGRAPHY

Broman, Karl W., Hao Wu, Saunak Sen, and Gary A. Churchill. "R/qtl: QTL Mapping in Experimental Crosses." *Bioinformatics* 19.7 (2003): 889-90. *Bioinformatics Oxford Journals*. Web. 06 May 2012. <http://bioinformatics.oxfordjournals.org/content/19/7/889.full.pdf+html>.

Colbourne, John K., Michael E. Pfrender, Donald Gilbert, W. Kelley Thomas, Abraham Tucker, Todd H. Oakley,Shinichi Tokishita, Andrea Aerts, Georg J. Arnold, Malay Kumar Basu, Darren J. Bauer, Carla E. Cáceres, Liran Carmel, Claudio Casola, Jeong-Hyeon Choi, John C. Detter, Qunfeng Dong, Serge Dusheyko, Brian D. Eads, Thomas Fröhlich, Kerry A. Geiler-Samerotte, Daniel Gerlach, Phil Hatcher, Sanjuro Jogdeo, Jeroen Krijgsveld, Evgenia V. Kriventseva, Dietmar Kültz, Christian Laforsch, Erika Lindquist, Jacqueline Lopez, J. Robert Manak, Jean Muller, Jasmyn Pangilinan, Rupali P. Patwardhan, Samuel Pitluck, Ellen J. Pritham, Andreas Rechtsteiner, Mina Rho, Igor B. Rogozin, Onur Sakarya, Asaf Salamov, Sarah Schaack, Harris Shapiro, Yasuhiro Shiga, Courtney Skalitzky, Zachary Smith, Alexander Souvorov, Way Sung, Zuojian Tang, Dai Tsuchiya, Hank Tu, Harmjan Vos, Mei Wang, Yuri I. Wolf, Hideo Yamagata, Takuji Yamada, Yuzhen Ye, Joseph R. Shaw, Justen Andrews, Teresa J. Crease, Haixu Tang, Susan M. Lucas, Hugh M. Robertson, Peer Bork, Eugene V. Koonin, Evgeny M. Zdobnov, Igor V. Grigoriev, Michael Lynch, and Jeffrey L. Boore. "The Ecoresponsive Genome of Daphnia Pulex." Science 331 (2011): 555-61. Science, The World's Leading Journal of Original Scientific Research, Global New, and Commentary. 16 Dec. 2010. Web. 10 Mar. 2012. <http://www.sciencemag.org/content/331/6017/555.full>.

Hartigan, J. A., and P. M. Hartigan. "The Dip Test of Unimodality." *The Annals of Statistics* 13.1 (1985): 70-84. *Project Euclid*. Web. 02 May 2012. <http://projecteuclid.org/DPubS/Repository/1.0/Disseminate?view=body&id=pdf_1&handle=euclid.aos/1176346577>.

Maechler, Martin. "Package 'diptest'" Package 'diptest' 18 Apr. 2012. Web. 02 May 2012. <http://cran.r-project.org/web/packages/diptest/diptest.pdf>.

"MPICH2 : High-performance and Widely Portable MPI." *MPICH2*. Web. 02 May 2012. <http://www.mcs.anl.gov/research/projects/mpich2/>.

"Roche NimbleGen." Roche NimbleGen. 2012. Web. 02 May 2012. <http://www.nimblegen.com/>.

Routtu, Jarkko, Bastiaan Jansen, Isabelle Colson, Luc De Meester, and Dieter Ebert. "The First-generation Daphnia Magna Linkage Map." *BMC Genomics* 11.1 (2010): 508. *BMC Genomics*. Web. 02 May 2012. <http://www.biomedcentral.com/1471-2164/11/508>.

"The R Project for Statistical Computing." *The R Project for Statistical Computing*. Web. 02 May 2012. <http://www.r-project.org/>.

Wang, Xiangfeng, and Hang He. "SpatialSmooth." *Yale Plant Genomics*. Web. 02 May 2012. <http://plantgenomics.biology.yale.edu/nmpp/SpatialSmoot.htm>.

Wang, X., H. He, L. Li, R. Chen, X. W. Deng, and S. Li. "NMPP: A User-customized NimbleGen Microarray Data Processing Pipeline." *Bioinformatics* 22.23 (2006): 2955-957. *Bioinformatics Oxford Journals*. Web. 02 May 2012. <http://bioinformatics.oxfordjournals.org/content/22/23/2955.full>.

APPENDICES

APPENDIX A - THE EXPERIMENTAL EVALUATION SUMMARY

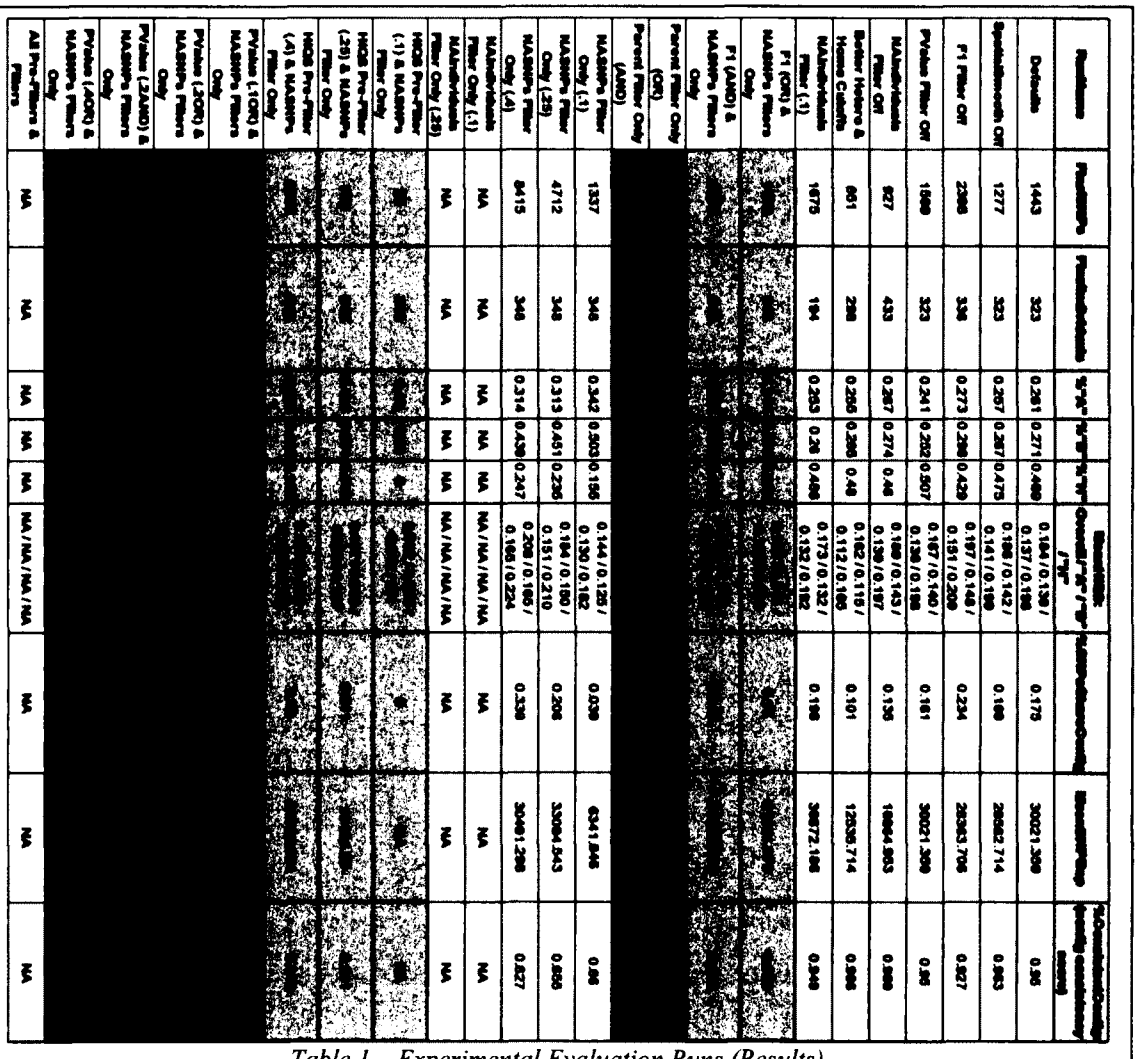


Table 1 – Experimental Evaluation Runs (Results)

The table has been rotated 90 degrees clockwise in order to fit on the page. Each row is a pipeline run, and each column is a result. The final number of SNPs and individuals, the quality scores, and other statistics are reported for each pipeline run. The row colors do not have a particular meaning. Similar pipeline runs are grouped together and their rows have the same color. One group of similarly colored rows is not related to another group of rows of the same color. Each row in this table should line up with the same row in *Table 2*.

Table 2 – Experimental Evaluation Runs (Parameter Values)

The table has been rotated 90 degrees clockwise in order to fit on the page. Each row is a pipeline run, and each column is a parameter. The values in each cell are the parameter values for each run. Each filter column also contains the number of SNPs or individuals removed by that filter. The row colors do not have a particular meaning. Similar pipeline runs are grouped together and their rows have the same color. One group of similarly colored rows is not related to another group of rows of the same color. Each row in this table should line up with the same row in *Table 1*.

APPENDIX B - THE PARENT MATCHING PSEUDO-CODE

The below pseudo-code and the embedded comments should be self-explanatory. The pseudo-code illustrates the process that determines a SNP's mapping allele within an individual.

Note the reason for only one nucleotide being examined in each parent clone: the parent clones' SNPs are homozygous. Therefore, each SNP's nucleotides on the associated pair of homologous chromosome in the forward strand are the same. The nucleotides in the reverse strand do not need to be examined because they are already known: they are the complement of the forward strand's nucleotide. Consequently, the parent allele file (refer to appendix D for detailed input specifications) only lists one nucleotide in the forward strand at each SNP within each parent clone because the other nucleotides are inherently known.

```
1  nuc1_F; //the nucleotide in the forward strand with
2          //the highest luminescent reading
3  nuc2_F; //the nucleotide in the forward strand with
4          //the second highest luminescent reading
5  nuc1_R; //the nucleotide in the reverse strand with
6          //the highest luminescent reading
7  nuc2_R; //the nucleotide in the reverse strand with
8          //the second highest luminescent reading
9  nuc_A; //the nucleotide in the forward strand of the "A" parent
10 nuc_B; //the nucleotide in the forward strand of the "B" parent
11
12 classification_F; //the classification of the forward strand HomoIndex value.
13                   //It can either be "heterozygous", "homozygous", or "unknown".
14                   //By the time this code is executed, all the pre-filter
15                   //computations besides the below matching computations
16                   //have been executed, so this classification is either a
17                   //valid "heterozygous or "homozygous" classification.
18 classification_R; //the classification of the reverse strand HomoIndex value.
19
20 nucleotideGenotypeMappingAlleleOption; //If this option is enabled, the valid
21                                        //mapping alleles returned are
22                                        //the SNP nucleotides in the forward
23                                        //strand rather than the parent matching
24                                        //alleles
25
26 //the two strand classifications don't match
27 if(classification_F != classification_R) { return "NA"; }
28 //homozygous mapping allele
29 else if(classification_F == "homozygous" && nuc1_F == compliment(nuc1_R))
30 {
31   if(nucleotideGenotypeMappingAlleleOption == "enabled") { return "nuc1_F"; }
32   else if(nuc1_F == nuc_A) { return "A"; }
33   else if(nuc1_F == nuc_B) { return "B"; }
34   else{ return "NA"; }
35 }
36 //heterozygous mapping allele
37 else if(classification_F == "heterozygous" &&
38      ((nuc1_F == compliment(nuc1_R) &&
39        nuc2_F == compliment(nuc2_R)) ||
40       (nuc1_F == compliment(nuc2_R) &&
41        nuc2_F == compliment(nuc1_R))))
42 {
43   if(nucleotideGenotypeMappingAlleleOption == "enabled")
44   {
45     return "nuc1_F" + "nuc2_F";
46   }
47   else if(nuc1_F == nuc_A)
48   {
49     if(nuc2_F == nuc_B) { return "H"; }
50     else { return "NA"; }
51   }
52   else if(nuc1_F == nuc_B)
53   {
54     if(nuc2_F == nuc_A) { return "H"; }
55     else { return "NA"; }
56   }
57   else{ return "NA"; }
58 }
59 //the nucleotide compliments do not match
60 else { return "NA"; }
```

Figure 14 – The Parent Matching Pseudo-Code

The pipeline pseudo-code that determines a SNP's mapping allele within an individual based on the SNP's nucleotides in the forward and reverse DNA strands on both associated homologous chromosomes, the nucleotides in the parent clones, and whether the SNP has been classified as homozygous or heterozygous in the associated individual. Refer to the pseudo-code's comments (indicated by "//") for more information.

APPENDIX C - THE PIPELINE PARAMETERS

The pipeline parameters are organized in two groups (refer to the *Pipeline Parameters* section). Each parameter is listed and described here along with its default value. The parameters are inputted to the pipeline using XML files. An example XML file is provided below. Refer to the "userParameters.xml" and "softwareParameters.xml" files, provided with the pipeline code, for more information about the pipeline parameters.

C.1 The User Parameters

C.1.1 Input Parameters

- Parent Allele File - The path and name of the parent allele file described in appendix D. Default Value = ./parentAllele.txt
- NDF File - The path and name of the NDF file described in appendix D. Default Value = ./nimbleGen.ndf
- Bases Files Directory - The path and name of the directory containing the bases files to send through the pipeline. Default Value = ./basesFiles

C.1.2 Spatial Smooth Parameters

- Spatial Smooth - Enable or disable the *Spatial Smooth* step. Default Value = Enabled
- X Slide Dimension - The width of the microarray slides in the X dimension. The number must be evenly divided by the square root of the Number of Zones parameter. Default Value = 480
- Y Slide Dimension - The height of the microarray slides in the Y dimension. The number must be evenly divided by the square root of the Number of Zones parameter. Default Value = 640
- Number of Zones - The number of zones the *Spatial Smooth* algorithm should divide each microarray slide into. The square root of the number must be a whole number. Default Value = 256
- Smooth Number - A trivial parameter that prevents divisions by zero in the *Spatial Smooth* algorithm. Default Value = 100

C.1.3 HomoIndex Parameters

- HomoIndex - Enable or disable the *HomoIndex* step. Default Value = Enabled
- HomoIndex Quality Score Pre-Filter - Enable or disable the HIQS pre-filter computation. Default Value = Disabled
- HomoIndex Quality Score Pre-Filter Cutoff Value - The maximum HIQS that is considered valid. Default Value = 0.2375
- Heterozygous Cutoff Value - The minimum heterozygous HomoIndex value. Default Value = 0.525

- Homozygous Cutoff Value - The maximum homozygous HomoIndex value. Default Value = 0.475

C.1.4 FIGMDP Parameters

- FIGMDP - Enable or disable the *FIGMDP* step. Default Value = Enabled
- F1 Filter - Enable or disable the F1 filter. Default Value = Enabled
- Parent Filter - Enable or disable the parent filter. Default Value = Disabled
- NA SNPs Filter - Enable or disable the NA SNPs filter. Default Value = Enabled
- NA Individuals Filter - Enable or disable the NA Individuals filter. Default Value = Enabled
- Duplicates Filter - Enable or disable the duplicates filter. Default Value = Enabled
- F1 Filter OR/AND - Choose the OR/AND option for the F1 filter. Default Value = OR
- Parent Filer OR/AND - Choose the OR/AND option for the parent filter. Default Value = OR
- NAs Allowed Per SNP - The maximum percent of invalid mapping alleles associated with a SNP across all individuals that will pass the NA SNPs filter. Default Value = 0.25
- NAs Allowed Per Individual - The maximum percent of invalid mapping alleles associated with an individual across all SNPs that will pass the NA individuals filter. Default Value = 0.25

- Output F1 Filter - Enable or disable the option to output the SNPs removed by the F1 filter to the biology report file. Default Value = Disabled
- Output Parent Filter - Enable or disable the option to output the SNPs removed by the parent filter to the biology report file. Default Value = Disabled
- Output NA SNPs Filter - Enable or disable the option to output the SNPs removed by the NA SNPs filter to the biology report file. Default Value = Disabled
- Output NA Individuals Filter - Enable or disable the option to output the individuals removed by the NA individuals filter to the biology report file. Default Value = Disabled
- Output Duplicates Filter - Enable or disable the option to output the individuals removed by the duplicates filter to the biology report file. Default Value = Disabled

C.1.5 Dip Test Parameters

- Dip Test - Enable or disable the *Dip Test* step. Default Value = Enabled

C.1.6 Final Genotype Mapping Parameters

- P-Value Filter - Enable or disable the p-value filter. Default Value = Enabled
- Individuals to Ignore - The path and name of the file listing the names of the individuals, one per line, that should always be removed from the final genotype mapping files by the *Final Genotype Mapping* step. This file usually consists of the F1 and X and I clone individuals. Default Value = ./individuals_to_ignore.txt

- P-Value Filter OR/AND - Choose the OR/AND option for the p-value filter. Default Value = OR
- Significant P-Value - The maximum p-value that is considered valid. Default Value = 0.2
- Output P-Value Filter - Enable or disable the option to output the SNPs removed by the p-value filter to the biology report file. Default Value = Disabled
- Output Ignored Individuals - Enable or disable the option to output the individuals that made it unfiltered to the *Final Genotype Mapping* step, but were removed because they were listed in the "Individuals to Ignore" file. They will be output to the biology report file Default Value = Disabled
- HomoIndex Quality Score Choice - Every mapping allele has two associated HomoIndex quality scores (one for the forward DNA strand, and one for the reverse DNA strand). This parameter controls which HIQS is output to the final HomoIndex quality scores file described in appendix E. The options are the best of the two, the worst of the two, or the average of the two. The indicated option is applied to every mapping allele. The HomoIndex quality scores examined in the *Experimental Evaluation* section were generated using the average of the two. Default Value = The average of the two.

C.1.7 Sample XML File

The user parameters XML file is relatively basic. Each parameter has an associated tag, and the tag's content is the value of the parameter. The pipeline uses the default value of any parameter not present in the XML file. Comments are ignored by

the pipeline, and the parameters do not have to appear in any particular order within the file. The example below lists four parameters. Refer to the "userParameters.xml" file, provided with the pipeline code, for a full user parameters file.

```
<?xml version='1.0'?>
<ArrayMapper>
        <bases_files_dir>/home/usr/basesFiles/</bases_files_dir>
        <hetero_cutoff>0.525</hetero_cutoff>
        <NAs_allowed_per_individual>0.25</NAs_allowed_per_individual>
        <significant_pValue>0.2</significant_pValue>
</ArrayMapper>
```

Figure 15 – Sample XML File

The above XML code is an example user parameters XML file. Only four parameters are illustrated. The actual user parameters file has many more parameters.

C.2 The Software Parameters

The software parameters are not provided here. The associated XML file is structured in the same way as the user parameters XML file. The software parameters do not affect the results or statistics generated by the pipeline. The software parameters designate output files and folders, locations of executables, and the specifics of the cluster running the pipeline. The "softwareParameters.xml" file, provided with the pipeline code, contains each parameter, and the file's comments explain the parameters. Refer to the "XMLParser.pl" script, provided with the pipeline code, to view the software parameters' default values.

APPENDIX D - THE INPUT FILES

There are three sources of input to the pipeline: the bases files, the NDF file, and the parent allele file.

D.1 The Bases Files

The bases files are the main source of input. Each bases file contains the luminescent readings from one individual's microarray slide. A sample microarray slide is shown in *Fig.* 17. The bases files are generated by processing the slide images, generated by the array hybridization technique, with the NimbleScan software (http://www.nimblegen.com/, 2012). The bases files contain eight lines of data per SNP: four for the forward DNA strand, and four for the reverse DNA strand. The bases file specification can be found on page 122 at the following URL:

http://www.nimblegen.com/products/lit/NimbleScan_v2p6_UsersGuide.pdf

D.1.1 Sample Bases File

The following are sample data lines associated with two SNPs in a bases file. The columns are in the following order: SEQ_ID, POSITION, REFERENCE_BASE, QUERY_BASE, MEAN, STDEV, PIXELS.

contig80560_159	159	A	A	926.51	177.14	49
contig80560_159	159	A	C	1281.18	391.52	49
contig80560_159	159	A	G	1040.47	215.83	49
contig80560_159	159	A	T	5868.16	3109.31	49
contig80560_159	159	T	A	4440.65	2162.28	49
contig80560_159	159	T	C	1556.98	434.86	49
contig80560_159	159	T	G	1169.12	242.25	49
contig80560_159	159	T	T	1111.18	283.74	49
scaffold00001_4229	4229	A	A	1142.86	347.95	49
scaffold00001_4229	4229	A	C	2695.20	1298.26	49
scaffold00001_4229	4229	A	G	1346.84	423.94	49
scaffold00001_4229	4229	A	T	2072.27	809.00	49
scaffold00001_4229	4229	T	A	2522.43	1109.77	49
scaffold00001_4229	4229	T	C	1171.47	266.41	49
scaffold00001_4229	4229	T	G	2463.10	1107.77	49
scaffold00001_4229	4229	T	T	1252.29	2208.03	49

Figure 16 – Sample Bases File

The above is a data set is from a sample bases file. Refer to this appendix and the NimbleGen bases file specification for more information.

D.1.2 Parsing the Bases Files

Five pieces of information are parsed out of every line:

- SEQ_ID - This column contains the SNP identifier for each line: the unique ID assigned to each SNP. Each SNP identifier appears in eight different lines.
- REFERENCE_BASE - This column designates the DNA strand each line is associated with. Each SNP identifier has four readings per strand. A REFERENCE_BASE of "A" indicates forward strand readings, and a REFERENCE_BASE of "T" indicates reverse strand readings.
- QUERY_BASE - This column designates the nucleotide at the SNP position in the oligonucleotide of each line. A QUERY_BASE can either be "A", "C", "G", or "T".

- MEAN - A scanner examines each microarray's luminescent readings and generates an image for each microarray. The NimbleScan software determines each oligo's luminescent readings within a bases file by examining the known oligo locations within the associated image. Therefore, each oligo position on a microarray slide occupies a finite amount of pixels in the associated image. Each position occupies the same number of pixels. Consequently, the MEAN column contains each line's average luminescent reading across all the associated pixels. Throughout this paper, a luminescent reading refers to the average luminescent reading of all the associated pixels.
- STDEV - This column contains the standard deviation of each line's associated pixels' luminescent readings.

The pipeline makes certain assumptions about the format of the bases files:

- The file name of each bases file is the name of the individual associated with that bases file.
- There is no white space in the names of the bases files.
- There is only one period in each bases file name, and it is the period preceding the file extension.
- The file extension of each bases file is "bases".
- Underscores separate information embedded in the bases file names.
- The last section of text in the bases file names before the file extension and after the final underscore identifies duplicate individuals. If that text is identical

between two bases files, the individuals associated with those bases files are considered duplicates.

- The F1 individuals have "_F1" somewhere in their bases file names, and the X and I clone individuals have "X_clone" and "I_clone" somewhere in their respective bases file names.
- The columns within the bases files are tab delimited.
- There are five header lines within each bases file that the pipeline ignores.
- The SEQ_ID values have a specific format. The values are the SNP identifier for each line, and each identifier is the contig identifier containing the associated SNP, followed by an underscore, followed by the position of the SNP within the contig. The contig identifier can be any number of letters and numbers, and the SNP position can be any number of numbers. For example, "contig36730_681" and "scaffold02338_752" are valid SNP identifiers.
- The REFERENCE_BASE and QUERY_BASE values are capital letters.
- Each SNP's forward strand lines are directly before its reverse strand lines, and the four lines associated with a strand are ordered by their QUERY_BASEs in the order "A", "C", "G", "T".

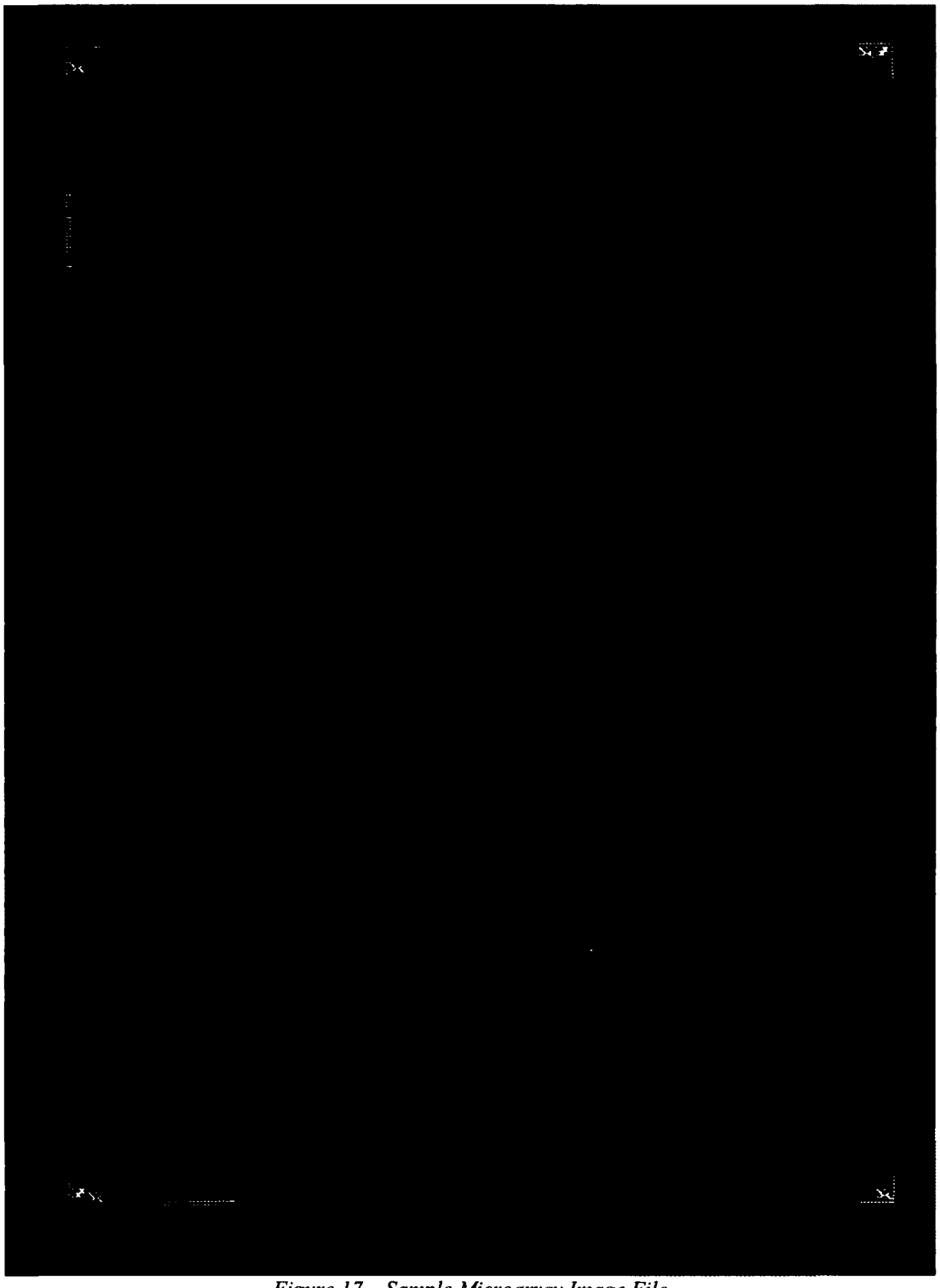


Figure 17 – Sample Microarray Image File

This image was generated by NimbleGen, and it illustrates the luminescent readings on a microarray slide. NimbleGen's software NimbleScan generates bases files by processing images like this one. Each image generates one bases file.

D.2 The NDF File

The NDF file contains one piece of information needed by the *Spatial Smooth* step. The NDF file is generated by NimbleGen, and it represents the design of the microarray slides. Each oligonucleotide's location is listed in the file. Consequently there are eight lines of data per SNP (similar to the bases files). The NDF file contains other information regarding the design of the slides (such as the nucleotides composing each oligonucleotide sequence), but the pipeline ignores all information other than the X and Y locations of each oligo. The NDF file specification can be found on page 106 at the following URL:

http://www.nimblegen.com/products/lit/NimbleScan_v2p6_UsersGuide.pdf

D.2.1 Parsing the NDF File

Five pieces of information are parsed out of every line:

- SEQ_ID - This column contains the SNP identifier for each line: the unique ID assigned to each SNP. Each SNP identifier appears in eight different lines.
- SELECTION_CRITERIA - This column is a combination of the REFERENCE_BASE and QUERY_BASE columns described in the *Bases Files* section of this appendix. Each value is the line's REFERENCE_BASE, a forward slash, and the line's QUERY_BASE. The only difference is the QUERY_BASE is a lowercase letter.

- X - This column contains each oligonucleotide's X position on the microarray slides.
- Y - This column contains each oligonucleotide's Y position on the microarray slides.

The pipeline makes certain assumptions about the format of the NDF file:

- The columns within the NDF file are tab delimited.
- Contrary to the specification at the above URL, the SELECTION_CRITERIA column is the third column (zero indexed), the SEQ_ID column is the fourth column, the X column is the 15th column, and the Y column is the 16th column.
- The NDF file contains one header line that the pipeline ignores.

A sample NDF file is not provided because there are too many columns to display on the page. Refer to the NDF file specification at the above link, or contact one of the contributors listed in appendix G for a sample NDF file.

D.3 The Parent Allele File

The parent allele file lists the SNP nucleotides in the X and I clones. According to the biological structure described in the *Experiment* section, the parent clones' SNPs are homozygous. Also, the nucleotide on one DNA strand can be determined if the nucleotide on the opposite DNA strand is known because the nucleotides must be compliments. Therefore, only the nucleotide in the forward strand on one of the

associated homologous chromosomes is listed for each parent clone SNP. The nucleotide on the reverse strand is the compliment nucleotide, and the nucleotides on the opposite chromosome are the same because the SNPs are homozygous. *Fig. 18* shows a few sample lines from a parent allele file.

There is one line of data per SNP, and there are five columns per line (note that there are no header lines):

- SNP_CONTIG - The contig identifier that contains each SNP. This is a portion of the SNP identifier described in the *Bases Files* section of this appendix.
- SNP_POSITION - The position of each SNP in its containing contig. This is a portion of the SNP identifier described in the *Bases Files* section of this appendix.
- REFERENCE_BASE - This is not used by the pipeline and can be anything.
- X_NUCLEOTIDE - The X clone's SNP nucleotide in the forward DNA strand on one of the associated homologous chromosomes.
- I_NUCLEOTIDE - The I clone's SNP nucleotide in the forward DNA strand on one of the associated homologous chromosomes.

D.3.1 Sample Parent Allele File

The following lines are a sample data set associated with eight SNPs in a parent allele file.

contig00117	113	C	C	T
contig00158	74	G	G	A
contig00167	32	C	C	T
contig00171	157	A	A	G
contig00176	368	G	G	A
contig00181	515	T	T	G
contig00195	46	T	T	A
contig00199	441	T	T	C

Figure 18 – Sample Parent Allele File

The above is a data set from a sample parent allele file. Refer to this appendix for more information

APPENDIX E - THE OUTPUT FILES

The pipeline generates 11 output files of interest. Nine of those files are tab-delimited table files, and two of those files are the pipeline's report files.

E.1 The Tab-Delimited Table Files

The tab-delimited table files all use the same format. Each column pertains to a specific SNP, and each row pertains to a specific individual. The first row is a header row that contains the SNP identifier of every column. The first column is a header column that contains the individual name of every row. The entries in the tab-delimited table are what distinguish the files from each other:

- The two pre-FIGMDP HomoIndex value files (one for each DNA strand).
- The two post-FIGMDP HomoIndex value files (one for each DNA strand).
- The two final HomoIndex value files (one for each DNA strand).
- The final HomoIndex quality score file.
- The post-FIGMDP genotype mapping file.
- The final genotype mapping file.

E.1.1 The Pre-FIGMDP HomoIndex Value Files

There are two pre-FIGMDP HomoIndex value files: one for the forward DNA strand, and one for the reverse DNA strand. An entry in one of these files' tables is the HomoIndex value in the associated strand at the associated SNP in the associated individual. The values are between zero and one, and they are the values at every SNP within every individual before the filters are applied by the *FIGMDP* step. The *FIGMDP* step generates the files.

E.1.2 The Post-FIGMDP HomoIndex Value Files

There are two post-FIGMDP HomoIndex value files: one for the forward DNA strand, and one for the reverse DNA strand. An entry in one of these files' tables is the HomoIndex value in the associated strand at the associated SNP in the associated individual. The values are between zero and one, and they are the values at every SNP within every individual after the filters are applied by the *FIGMDP* step. The *FIGMDP* step generates the files.

E.1.3 The Final HomoIndex Value Files

There are two final HomoIndex value files: one for the forward DNA strand, and one for the reverse DNA strand. An entry in one of these files' tables is the HomoIndex value in the associated strand at the associated SNP in the associated individual. The values are between zero and one, and they are the values at every SNP within every individual after the pipeline concludes and every filter has been applied. The *Final*

Genotype Mapping step generates the files.

E.1.4 The Final HomoIndex Quality Score File

Recall that each mapping allele has two HomoIndex quality scores: one for the forward DNA strand, and one for the reverse DNA strand. The HomoIndex Quality Score Choice parameter described in appendix C designates which of the two HIQSs is output to the final HomoIndex quality score file. Therefore, each entry in this file's table is the HIQS at the associated SNP in the associated individual. The scores are the HIQSs at every SNP within every individual after the pipeline concludes and every filter has been applied. The *Final Genotype Mapping* step generates this file.

E.1.5 The Post-FIGMDP Genotype Mapping File

An entry in this file's table is the mapping allele at the associated SNP in the associated individual. There are two types of mapping alleles: the parent mapping alleles, and the nucleotide mapping alleles. Refer to the *Experiment* section for more information about the parent allele options, and refer to the *Nucleotide Mapping Alleles* section for more information about the nucleotide mapping allele options. The mapping alleles are the alleles at every SNP within every individual after the filters are applied by the FIGMDP step. The *FIGMDP* step generates this file.

E.1.6 The Final Genotype Mapping File

An entry in this file's table is the mapping allele at the associated SNP in the associated individual. There are two types of mapping alleles: the parent mapping alleles, and the nucleotide mapping alleles. Refer to the *Experiment* section for more information about the parent mapping allele options, and refer to the *Nucleotide Mapping Alleles* section for more information about the nucleotide mapping allele options. The mapping alleles are the alleles at every SNP within every individual after the pipeline concludes and every filter has been applied. The *Final Genotype Mapping* step generates this file.

E.1.7 Sample Final Genotype Mapping File

	SNP_2	SNP_3	SNP_4	SNP_4	...
individual_1	NA	B	B	NA	
individual_2	A	NA	A	A	
individual_3	B	A	H	H	
individual_4	H	H	B	NA	
individual_5	NA	B	H	H	
...					

Figure 19 – Sample Final Genotype Mapping File

The above data set is from a sample final genotype mapping file. The file is one of the many tab-delimited table output files generated by the pipeline. The entries in this table are one of the parent mapping allele options. Extra tabs are added to the lines in order to line up columns so that it is easier to read, but the actual tab-delimited table files only contain one tab between each element on a line. Refer to this appendix for more information, and refer to the *Experiment* section for more information on the parent mapping alleles.

E.2 The Report Files

There are two report files: the biology report file, and the performance report file. The biology report file contains statistics generated by each parallel instance of every pipeline step related to the biological results generated by a pipeline run. The performance report file contains statistics generated by each parallel instance of every pipeline step related to the performance of a pipeline run. Both files use the same format:

- The top of the file contains multiple header lines, indicating the date of the pipeline run, the time of the pipeline run, and the user and software parameter files used by the pipeline run.
- After the header lines, every line (except for blank separator lines) has a tag as its first line element. The tags are any number of characters, surrounded by brackets. For example: [this is a tag]. The tags indicate what the lines contain.
- The files are broken up into sections. There is a section for each pipeline step, and there are subsections for each parallel instance of each pipeline step. At the beginning of a pipeline step section, the parameter values used by that step are listed. Each parallel instance section is indicated by the hostname of the machine the instance ran on and the MPI rank of the instance.
- Within each parallel instance section are the statistics generated by each parallel instance. Each statistic uses one line, and each statistic's name is in the line's tag. There is a tab after the line's tag, followed by the statistic's value. Next is a space, followed by a dash, followed by a space, and the rest of the line is any number of words describing what the statistic means.

There are many biological statistics generated by the pipeline. Users should experiment with various pipeline runs to understand the statistics. Note that various user and software parameters enable and disable biological statistics. Refer to appendix C, the "XMLParser.pl" script, the "userParameters.xml" file, and the "softwareParameters.xml" file to understand the relationship between the parameters and the various statistics.

Few statistics are output to the performance report file. The run times of each pipeline step, and the run time of the entire pipeline are output. The performance of the pipeline has not been heavily evaluated, and currently these are the only performance statistics generated by the pipeline.

E.2.1 Sample Report File

The following example illustrates a sample report file. The actual report files contain more sections, more parameters, and more statistics.

```
------------------Run on 5/22/2012 at 11:53------------------
  softwareParametersFileName: ./softwareParameters.xml
  userParametersFileName: ./userParameters.xml

[STEP] HomoIndex ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
[PARAMETER]    0.525 - heteroCutoff - HomoIndex values above this number are heterozygous
[PARAMETER]    0.475 - homoCutoff - HomoIndex values below this number are homozygous
[PARAMETER]    2 - The number of parallel instances

[PARALLEL INSTANCE]    inquiry.unh.edu:0 ____________________________
[NUM SNPS]    16804 - The num of SNP Identifiers in the Parent Allele File
[MY NUM BASES FILES]    6 - This parallel instance processed 6 bases files

[PARALLEL INSTANCE]    inquiry.unh.edu:1 ____________________________
[NUM SNPS]    16804 - The num of SNP Identifiers in the Parent Allele File
[MY NUM BASES FILES]    6 - This parallel instance processed 6 bases files

[STEP] FIGMDP ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
[PARAMETER]    0 - F1Filter - 1 (on), 0 (off), enable or disable the F1 filter
[PARAMETER]    0 - parentFilter - 1 (on), 0 (off), enable or disable the parent filter
[PARAMETER]    1 - NASNPFilter - 1 (on), 0 (off), enable or disable the NA SNPs filter
[PARAMETER]    1 - The number of parallel instances

[PARALLEL INSTANCE]    inquiry.unh.edu:0 ____________________________
[NUM INVALID NA SNPS]    1750 - The num of SNPs removed by the NA SNPs filter
```

Figure 20 – Sample Report File

The above lines illustrate the format and statistics from a sample report file. The statistics are divided into sections pertaining to each pipeline step, and the statistics are further divided within each step into sections pertaining to each parallel instance of a step.

APPENDIX F - THE MEAN VS. STANDARD DEVIATION PHENOMENON

During the pipeline's development, it was realized that an interesting relationship exists between the mean and standard deviation components of our microarray luminescent readings (refer to appendix D for detailed input specifications). See *Fig. 21* for a plot of an individual's luminescent readings.

The plot represents the relationship seen among most of the individuals in our microarray data sets. Immediately we realized that different clusters of luminescent readings exhibit different relationships. We decided to divide each individual's luminescent readings into four clusters. See *Fig. 22* to understand the A, B, C, and D clusters.

We initially thought that one or more clusters must be caused by an error in the array hybridization technique. Therefore, we needed to implement a filter that would remove the invalid clusters. Immediately we hypothesized that the B cluster is the only valid cluster because the majority of an individual's luminescent readings fall in the B cluster. For the individual depicted in *Fig. 22*, the A cluster contains 629 luminescent readings, the B cluster contains 99877 luminescent readings, the C cluster contains 4956 luminescent readings, and the D cluster contains 327 luminescent readings.

We attempted an investigation into the cause of the clusters in order to confirm

whether the B cluster is the only valid cluster. We looked at four aspects of the mean vs. standard deviation relationship among the various microarray data sets:

- What cluster do the readings associated with SNPs that are not removed during a default pipeline run (refer to appendix A and C for detailed parameter specifications and the *Defaults* run's results) fall into?
- Do individuals that are removed during a default pipeline run exhibit a different mean vs. standard deviation relationship than individuals that are not removed by the same pipeline run?
- Are luminescent readings associated with one cluster coming from a specific location on the microarray slides? In other words, are certain locations or sections of the microarray slides generating invalid luminescent readings?
- Each microarray slide (refer to the *Array Hybridization Technique* section for more information) is actually used for two individuals. The DNA fragment solution and oligos associated with one of the individual are chemically marked using a different color than the DNA fragment solution and oligos associated with the other individual. This way, both individuals can use the same microarray slide, and the scanner that detects the luminescent readings can distinguish between the individuals by differentiating between the colors of the luminescent effects (http://www.nimblegen.com/, 2012). Therefore, we asked the question, does one color exhibit a different mean vs. standard deviation relationship than the other?

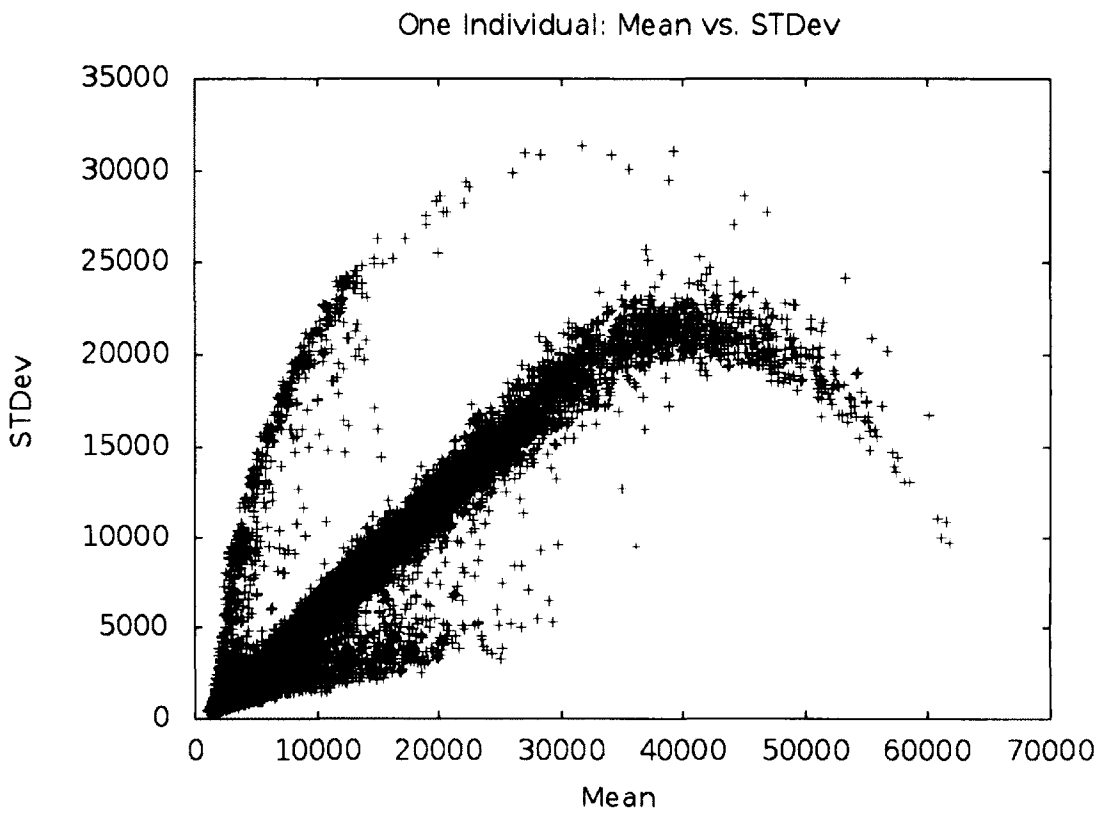


Figure 21 - An Individual's Mean vs. Standard Deviation Relationship

The plot illustrates the mean and standard deviation components of each luminescent reading associated with one individual. Each data point represents one oligonucleotide's luminescent reading. The readings are the exact readings generated by NimbleGen. In other words, no pipeline computations (such as the spatial smooth algorithm or the various pipeline filters) have altered or removed any readings.

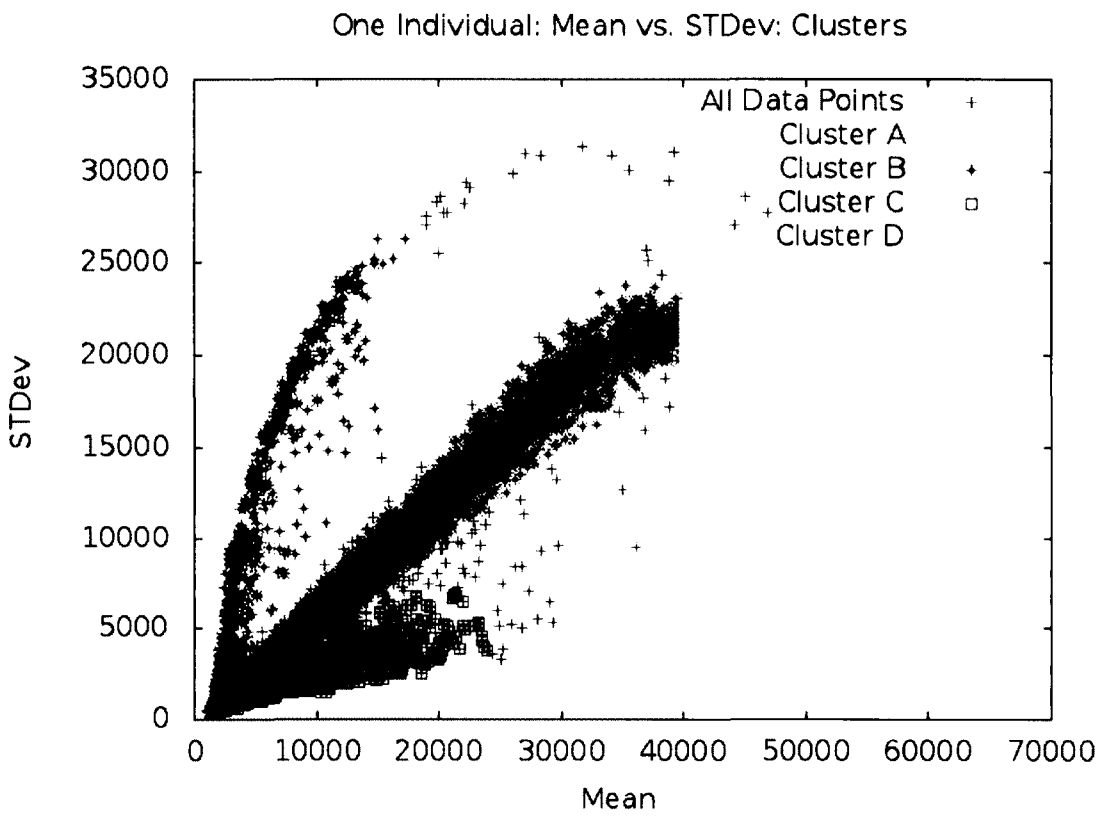


Figure 22 - The Mean vs. Standard Deviation Clusters

The plot illustrates the same data points seen in *Fig. 22*, however the data is divided into four clusters. Most of the individuals in our microarray data sets exhibit the A, B, C, and D clusters. The B cluster contains the majority of the data points. For this specific individual, the A cluster contains 629 data points, the B cluster contains 99877 data points, the C cluster contains 4956 data points, and the D cluster contains 327 data points.

F.1 Which Cluster Contains the Valid SNP Luminescent Readings?

Recall from the *Array Hybridization Technique* section that each SNP within each individual has eight associated luminescent readings. We highlighted the luminescent readings associated with SNPs that are not removed during a default pipeline run. We did this for multiple individuals that were also not removed during the same default pipeline run. The goal was to determine which cluster contains the readings associated with valid SNPs. Every individual generated a plot similar to *Fig. 23*.

The majority of the valid readings reside in the B cluster. Whether or not this supports our hypothesis that the B cluster contains the valid readings is uncertain. The B cluster originally contains the majority of the readings. Therefore, one could argue that the majority of the valid readings should reside in the B cluster. We conducted other investigations in order to alleviate this problem.

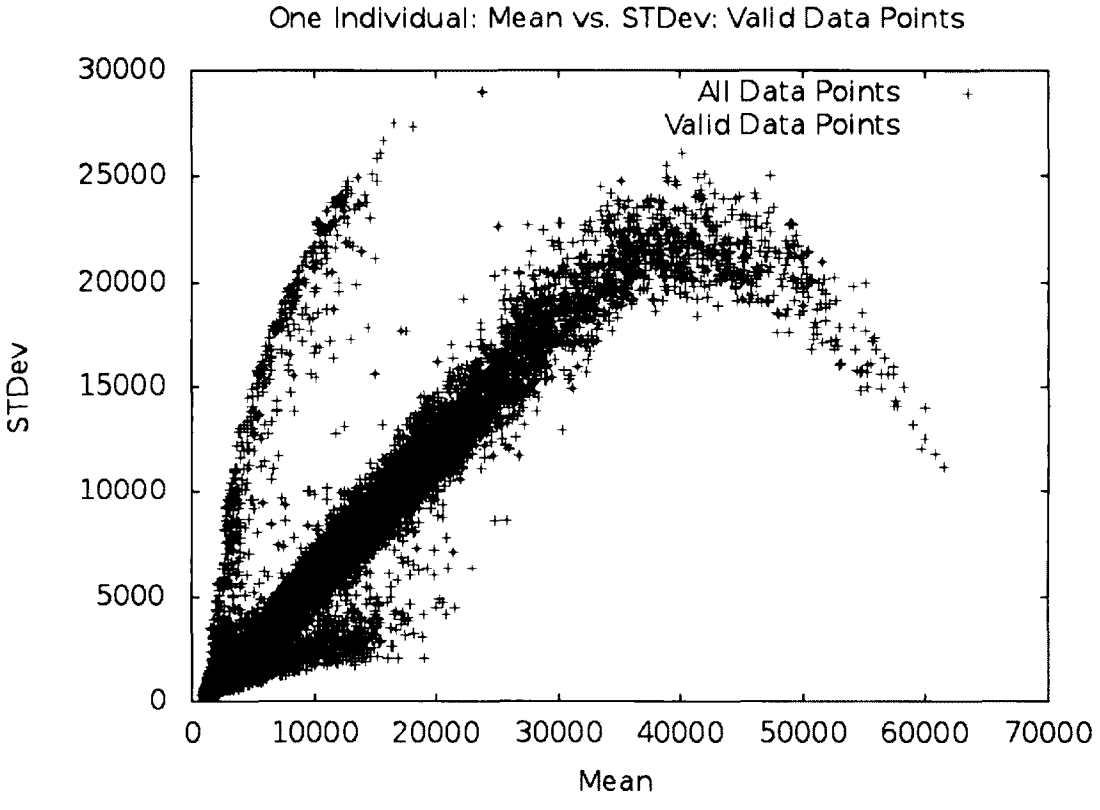


Figure 23 - The Valid SNP Luminescent Readings in the Mean vs. Standard Deviation Relationship The plot highlights the readings associated with valid SNP in one valid individual. The term "valid SNP" refers to a SNP that is not removed during a default pipeline run. The term "valid individual" refers to an individual that is not removed during a default pipeline run. The valid readings seem to mostly reside in the B cluster depicted in *Fig. 22.*

F.2 Valid Individuals vs. Invalid Individuals

We hypothesized that invalid individuals may exhibit a different mean vs. standard deviation relationship than valid individuals. Therefore, we randomly chose four individuals that were not removed during a default pipeline run. We also randomly chose four individuals that were removed during the same default pipeline run. Each individual's mean and standard deviation components of its luminescent readings are plotted in *Fig. 24* and *Fig. 25*.

The valid individuals have more distinct A, B, C, and D clusters than the invalid individuals. The valid individuals' B clusters are longer and thinner, and the invalid individuals do not necessarily fit the A, B, C, and D clustering scheme. This observation, combined with the fact that valid SNP readings (discussed in the previous section) reside in the B clusters of valid individuals, led us to believe that the linear relationship exhibited by the B cluster is the desired mean vs. standard deviation relationship within an individual. The next two investigations attempt to explain what is causing the three invalid clusters.

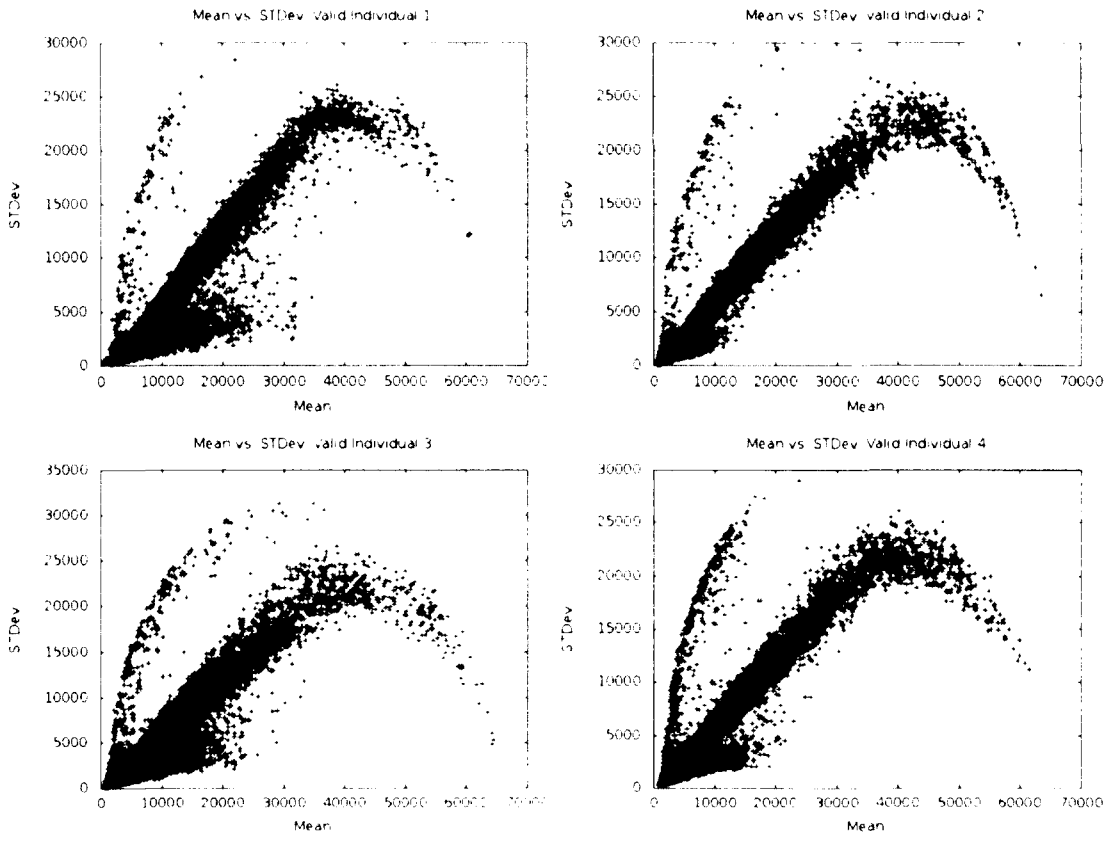


Figure 24 - Valid Individuals' Mean vs. Standard Deviation Relationship

These four mean vs. standard deviation plots illustrate the luminescent readings associated with four valid individuals. In other words, the individuals associated with these plots were randomly chosen from the set of individuals that were not removed during a default pipeline run.

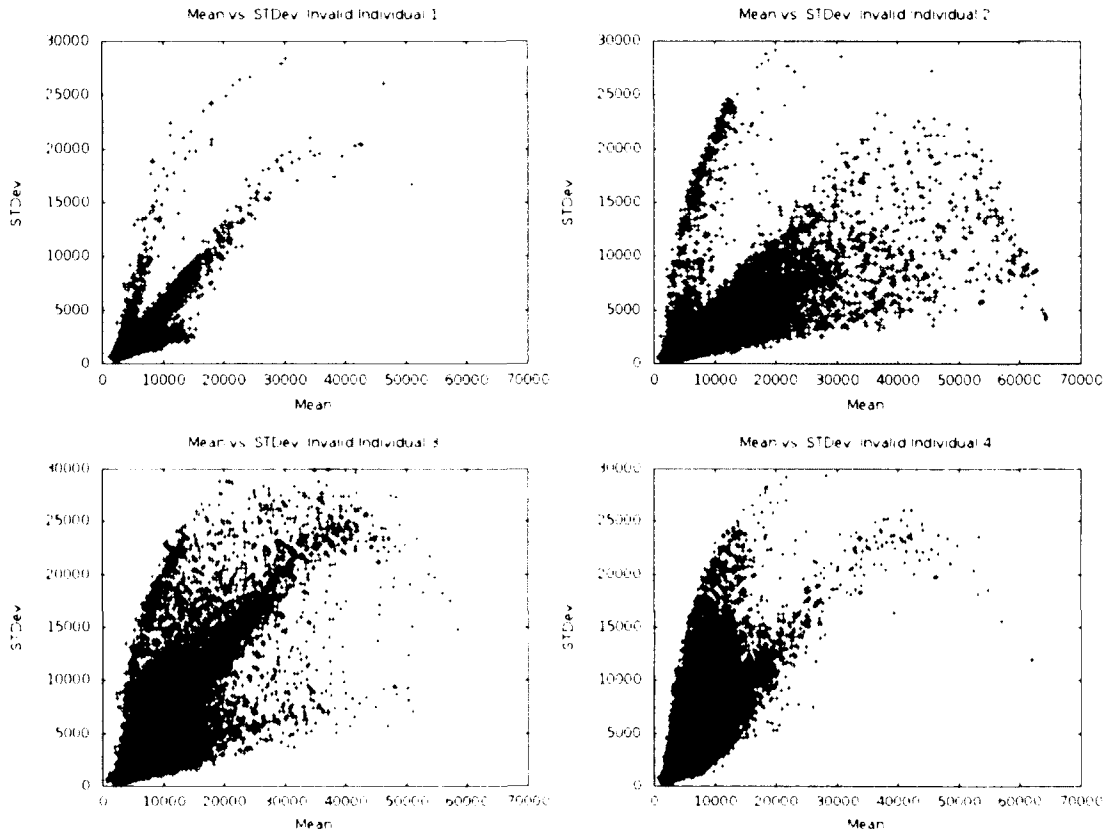


Figure 25 - Invalid Individuals' Mean vs. Standard Deviation Relationship

These four mean vs. standard deviation plots illustrate the luminescent readings associated with four invalid individuals. In other words, the individuals associated with these plots were randomly chosen from the set of individuals that were removed during a default pipeline run.

F.3 Are the Clusters Generated From Specific Microarray Locations

The array hybridization technique is not perfect. Therefore, it may be possible that certain sections of the microarray slides tend to generate invalid readings. Maybe those invalid readings are what compose the various clusters in the mean vs. standard deviation relationship. Refer to *Fig. 26-30* for the X and Y plots of the readings associated with each cluster within the individual depicted in *Fig. 22*.

Each X and Y plot shows a relatively uniform spatial distribution of the

luminescent readings across the microarray slide. The only notable grouping of readings is the (400, 400) location in the C cluster plot. Other than that, there does not seem to be any relationship between the mean vs. standard deviation clusters and the locations on the microarray slide. Consequently, we decided that the spatial distribution of the luminescent readings is not the cause of the mean vs. standard deviation clusters.

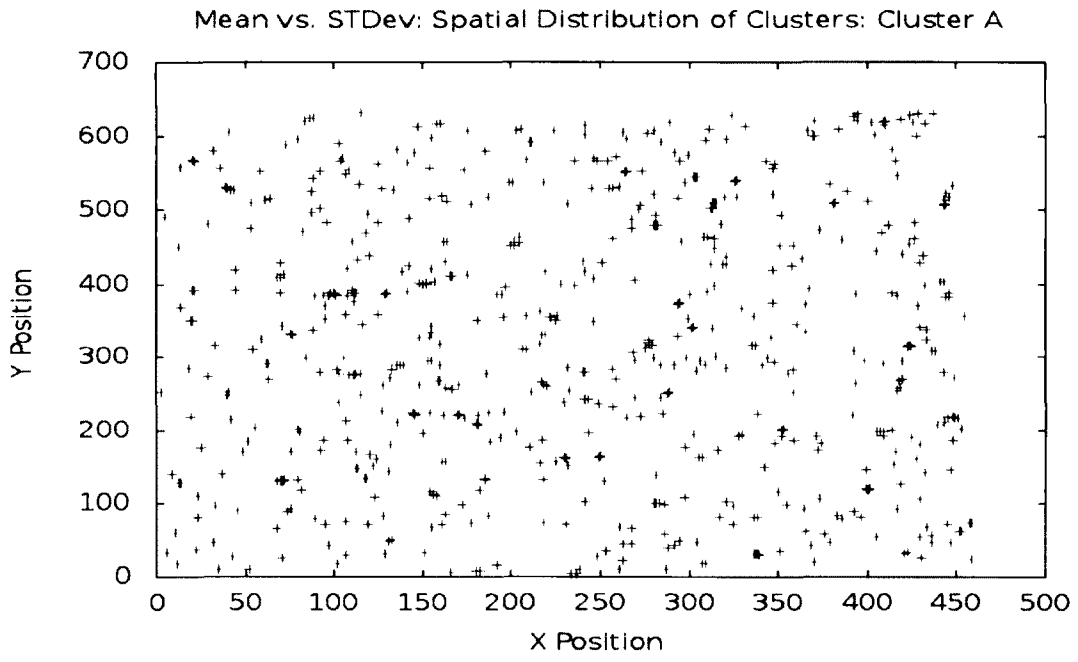


Figure 26 - Spatial Distribution of the A Cluster in the Mean vs. Standard Deviation Relationship
A plot of the X, Y location on the microarray slide of each luminescent reading in the A cluster associated with the individual depicted in *Fig. 22*. There does not seem to be any significant location with more data points than another.

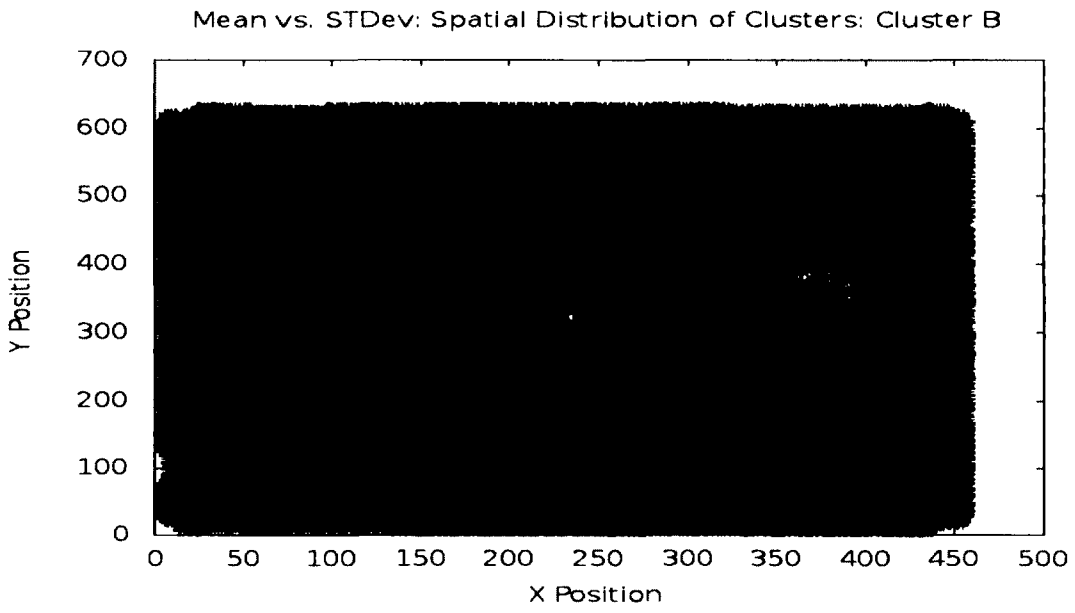


Figure 27 - Spatial Distribution of the B Cluster in the Mean vs. Standard Deviation Relationship
A plot of the X, Y location on the microarray slide of each luminescent reading in the B cluster associated with the individual depicted in *Fig. 22.* The B cluster contains the majority of the individual's readings, and data points fill almost every possible X, Y location on the microarray slide. There does not seem to be any significant location with more data points than another.

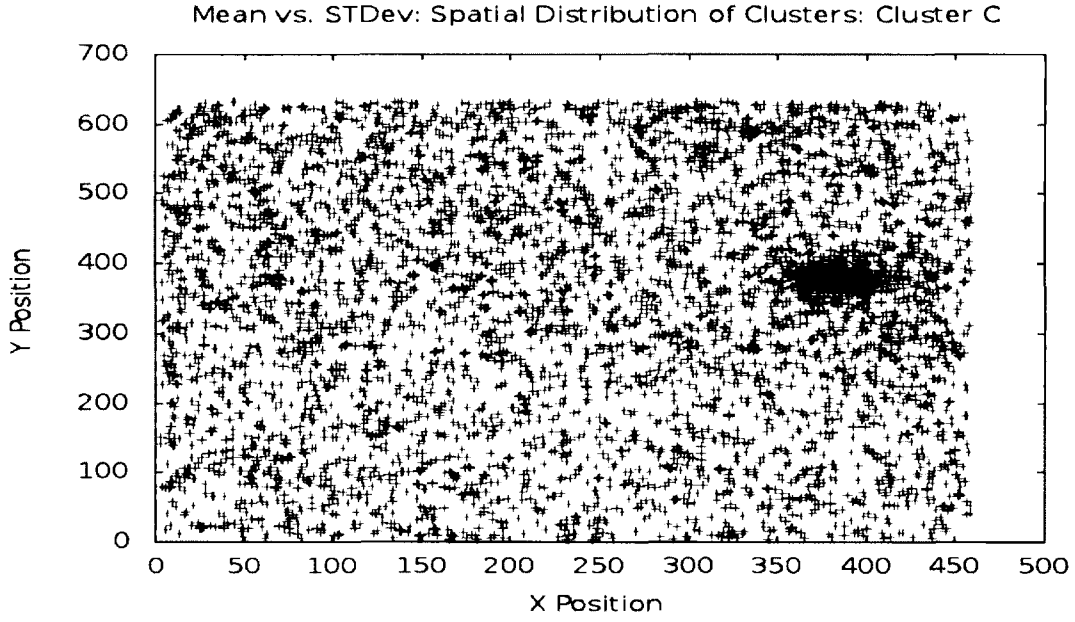


Figure 28 - Spatial Distribution of the C Cluster in the Mean vs. Standard Deviation Relationship
A plot of the X, Y location on the microarray slide of each luminescent reading in the C cluster associated with the individual depicted in *Fig. 22.* There seems to be one grouping of points at the location (400, 400). Besides that, there does not seem to be any other significant location with more data points than another.

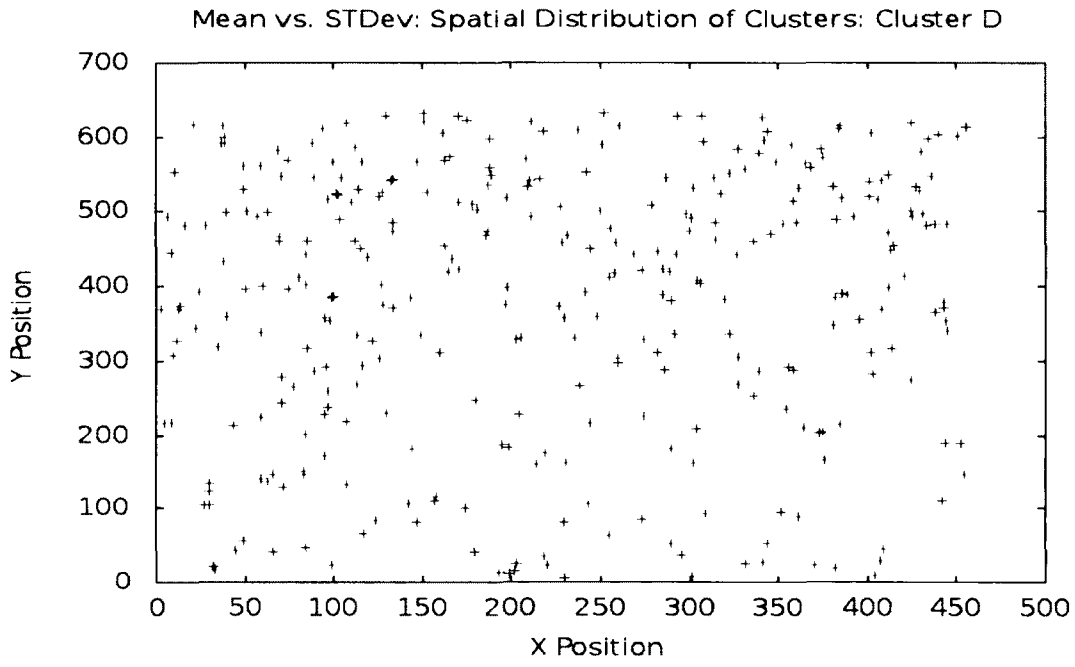


Figure 29 - Spatial Distribution of the D Cluster in the Mean vs. Standard Deviation Relationship
A plot of the X, Y location on the microarray slide of each luminescent reading in the D cluster associated with the individual depicted in *Fig. 22*. There does not seem to be any significant location with more data points than another.

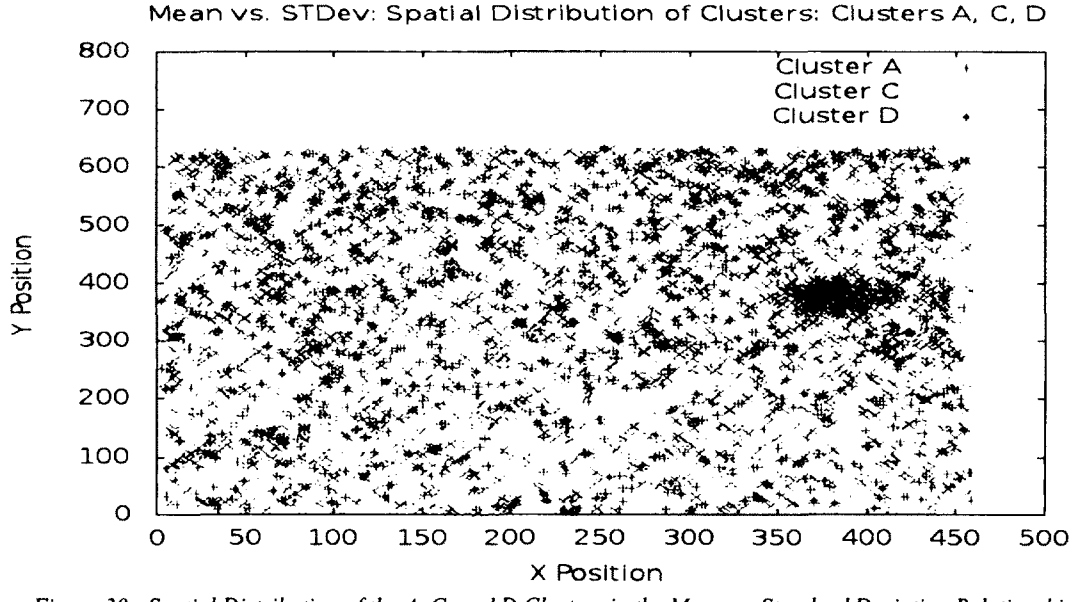


Figure 30 - Spatial Distribution of the A, C, and D Clusters in the Mean vs. Standard Deviation Relationship
A plot of the X, Y location on the microarray slide of each luminescent reading in the A, C, and D clusters associated with the individual depicted in *Fig. 22*. The B cluster is not depicted because it contains too many data points, and the other clusters would be hidden. Other than the C cluster grouping at the location (400, 400), the data points seem to be evenly distributed across the microarray slide.

F.4 Does One Color Exhibit a Different Relationship Than the Other?

Two chemical marking colors are used to generate the luminescent effects during the array hybridization technique. This allows one individual to use one color, another individual to use the other color, and the two individuals can use the same microarray slide (http://www.nimblegen.com/, 2012). We hypothesized that one color may generate a more favorable mean vs. standard deviation relationship than the other. Therefore, four individuals were randomly selected that used color 1. Their mean vs. standard deviation relationships were plotted, and the individuals that used the same microarray slide but the opposite color were plotted on top of the first individuals. Refer to *Fig. 31.*

One color does not seem to generate a different mean vs. standard deviation relationship than the other. Both colors result in the same clusters, and the differences between an individual that used one color and an individual that used the other color on the same microarray slide seem negligible. Consequently we decided that the two colors are not the cause of the mean vs. standard deviation phenomenon.

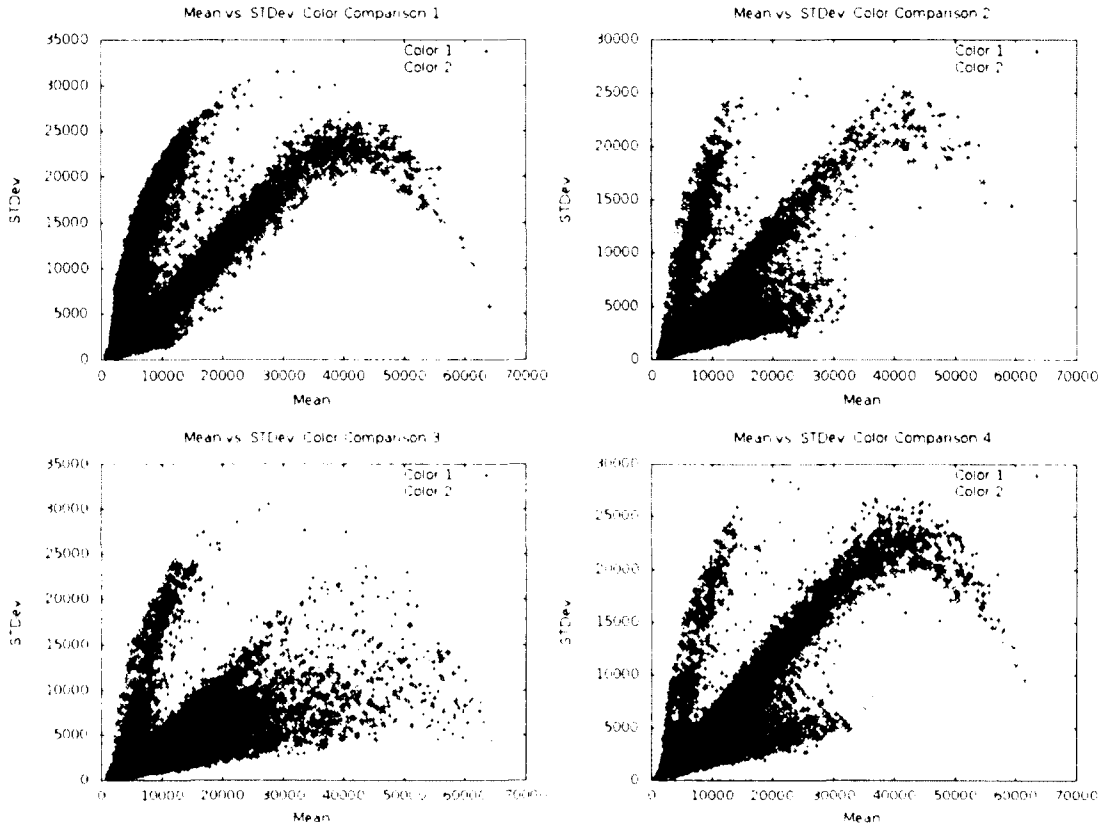


Figure 31 - Chemical Marking Color Comparison of the Mean vs. Standard Deviation Relationship

These plots illustrate four randomly selected individuals' mean vs. standard deviation relationships. The four individuals used the same color during the array hybridization technique (http://www.nimblegen.com/, 2012). Each individual also has a second individual plotted over it. Each corresponding second individual used the same microarray slide as the first individual, but the second individual used the opposite color to generate the luminescent effects. The mean vs. standard deviation relationships between two individuals on the same plot do not greatly vary. They appear to be nearly the same.

F.5 Conclusions and the Implemented Pre-Filter Computation

Through our investigations, we were not able to determine the cause of the mean vs. standard deviation clusters. However, we were able to determine three important things:

- Valid individuals tend to exhibit more distinct clusters, especially a more distinct elongated B cluster.
- The majority of the luminescent readings associated with an individual fall in the B cluster.
- The majority of the luminescent readings associated with valid SNPs fall in the B cluster.

We decided to implement a pre-filter computation that marks mapping alleles as invalid if they use a luminescent reading from a cluster other than the B cluster. The previously implemented pre-filter computations and filters seem to already be marking the majority of these mapping alleles as invalid since the majority of the luminescent readings associated with valid SNPs fall in the B cluster. However, we wanted to ensure that this was the case. Therefore, during the *HomoIndex* step, we mark mapping alleles as invalid if they are generated using at least one luminescent reading with at least one of the following qualities:

- The mean component of the luminescent reading is greater than 40000.
- The standard deviation component of the luminescent reading is greater than 25000.
- The luminescent reading falls outside the lines depicted in *Fig. 32*.

The result is that mapping alleles that use luminescent readings from the A, C, or D clusters are marked as invalid. The *NA SNPs Filter* and the *NA Individuals Filter*

(refer to the *Filters, Initial Genotype Mapping, and Dip Preparation* section) remove entire SNPs and individuals based on the invalid markings.

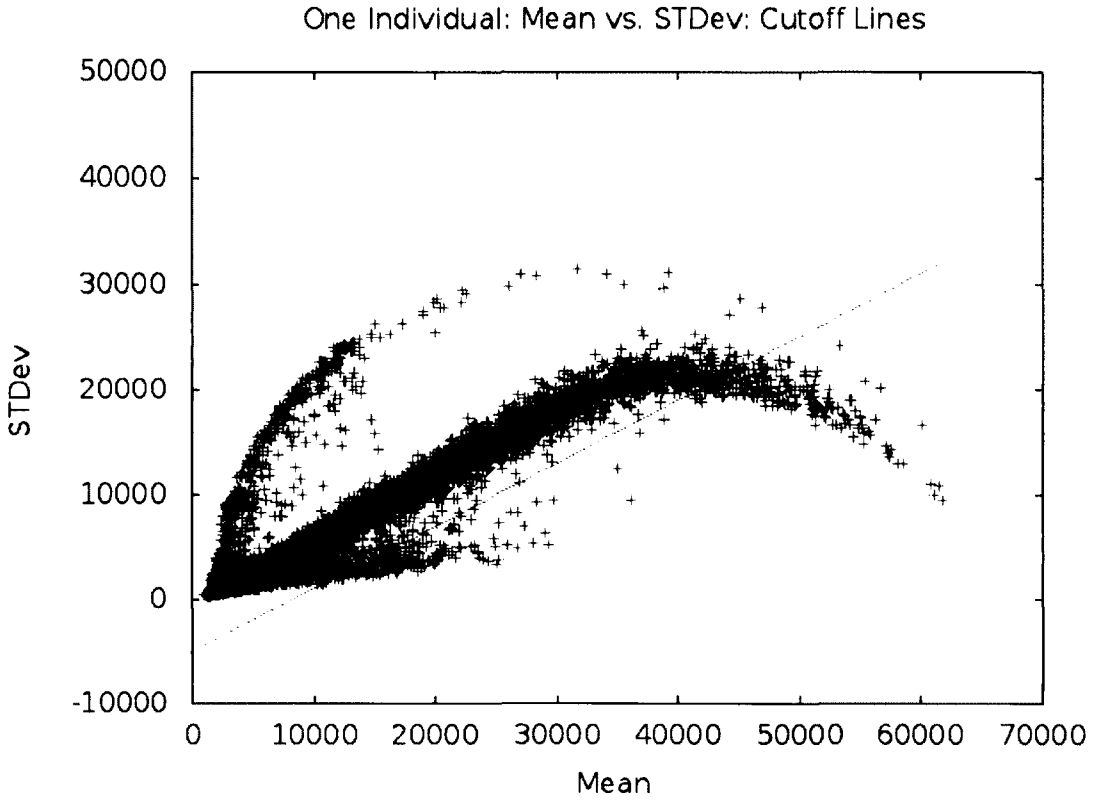


Figure 32 - The Mean vs. Standard Deviation Pre-Filter Cutoff Lines

The two lines indicate the minimum and maximum values of the B cluster depicted in *Fig. 22.* Mapping alleles that use luminescent readings that fall outside those lines are marked invalid. Luminescent readings with a mean component greater than 40000, or a standard deviation component greater than 25000, also cause their associated mapping allele to be marked invalid.

APPENDIX G - CODE AVAILABILITY AND CONTACT PERSONS

Currently the pipeline code and sample parameter files are not available for use on a public cluster or through the web. Contact one of the following people in order to determine the current state of the software, its availability, and how to acquire it.

- R. Daniel Bergeron
    - Professor of Computer Science at UNH
    - rdb@cs.unh.edu
- W. Kelley Thomas
    - Professor of Molecular, Cellular, and Biomedical Sciences at UNH
    - kelley.thomas@unh.edu
- Philip J. Hatcher
    - Professor of Computer Science at UNH
    - hatcher@unh.edu
- Brian Albere
    - Graduate of UNH
    - balbere55@gmail.com

APPENDIX H - INSTALLATION AND USER GUIDE

The following are required in order to install and use the pipeline:

- R, the statistical language (http://www.r-project.org/) must be installed on all the machines that will run the parallel instances of each pipeline step.
- The diptest package for R must be installed. The package can be installed by starting R in a Linux terminal and typing the command "install.packages("diptest")".
- MPI must be installed on the cluster that will run the pipeline.
- Perl and C are required to run the pipeline code. Use MPI's C compiler to compile the one C file: "EPIMPI.c". Specify the resulting C executable in the software parameters file as the "epimpi_executable" parameter.
- The Perl module "XML::Simple" must be installed. The "XMLParser.pl" script uses this module to parse the pipeline parameters from the XML files. More information on the module can be found at http://search.cpan.org/~grantm/XML-Simple-2.18/lib/XML/Simple.pm.

In order to run the pipeline, the appropriate input files must be present. Refer to appendix D for detailed input specifications. Also, the XML parameter files must be

properly configured. Refer to appendix G for more information on obtaining the code and sample parameter files. The sample parameter files should contain comments that describe the purpose of each pipeline parameter.

Once everything has been installed and the pipeline parameters have been configured, the pipeline can be started with the following command:

```
./Commander.pl -s softwareParameters.xml -u userParameters.xml
```

Users should be able to individually execute any of the pipeline steps' Perl executables without any command line arguments to learn more about each pipeline step. As the pipeline is running it will output progress information to the terminal.