Driving Genetics With Experimental Visualization

Alex Bigelow

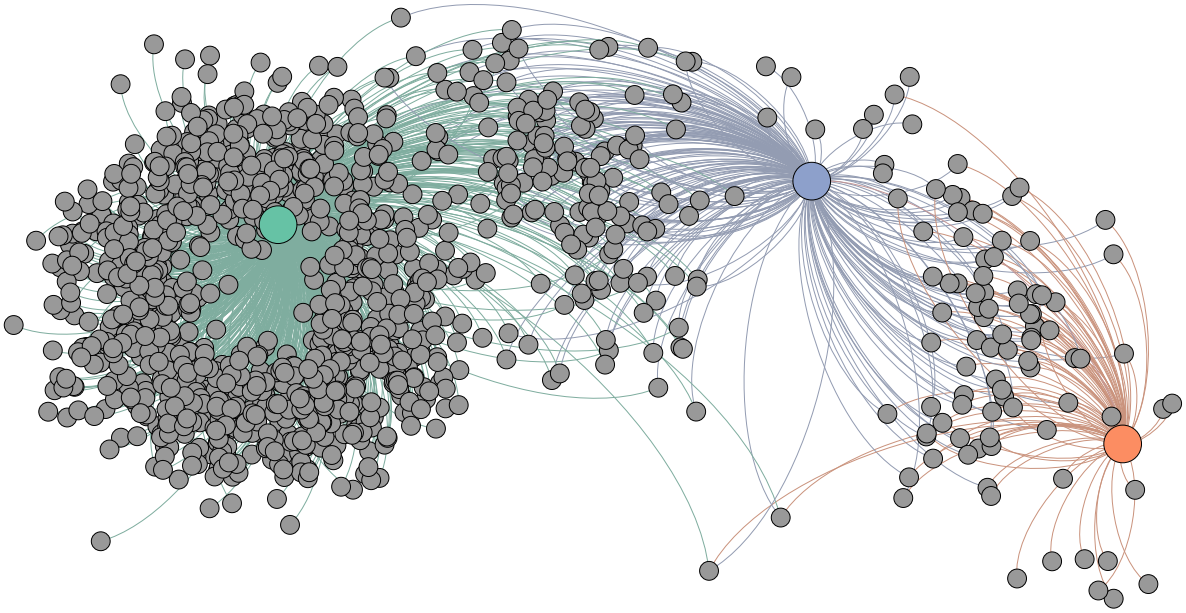


Fig. 1. Variants similar to rs3817578, rs3834129, and rs6723097 [4] using a basic similarity metric

Abstract—Some misunderstanding is inevitable in any interdisciplinary work, and recognizing and handling misunderstanding quickly is vital to the success of a collaboration. Many misunderstandings arise from what appears to be common knowledge to a domain scientist, but is not something that a computer scientist would think to ask about. The use of experimental visualization as an inter-collaboration communication tool can highlight these important misunderstandings that would otherwise severely impair the collaboration.

Visualization itself, however, is a powerful tool; if used carefully it can open doors to understanding and discovery, but it can also control, warp, or even limit how a visualization designer (and consequently a scientist) thinks about a problem. A willingness to abandon erroneous designs and even systems, regardless of the effort involved in creating and understanding them, is also very important for such a collaboration.

This case study of data exploration via experimental visualization is in the space of next-generation sequencing data. For most of the history of genetics, the main bottleneck in the rate of discovery has been the difficulty in obtaining data. Next generation sequencing is a relatively new technology that allows biologists a way to obtain genetic data at a fraction of the cost and unprecedented speeds compared to conventional sequencing. With the advent of NGS data, the bottleneck has shifted from the burden of data acquisition to the analysis of huge amounts of data.

1 Introduction

1.1 Purpose

Three approaches exist for studying inherited genetic diseases using next-generation sequencing data:

- A single-variant disease model,
- A multiple rare variant disease model (the "OR" question)
- A multiple common, combinatoric variant disease model (the "AND" question)

Each approach to identifying variants can greatly benefit from visualization. The purpose of this thesis is to explore and evaluate derived annotations, visual representations, and visualization technologies that could assist in exploring genetic variants. This will be done mainly in the context of a single-variant disease model, but in a manner that can be inform other two.

More importantly, this thesis is also an exploration of the process of computer scientist-biologist collaboration in the context of visualization and data exploration. In any collaboration, vital misunderstandings often arise from one party assuming a detail is commonly understood that another party would have no intuition to ask about. As a communication tool, experimental visualization can make these misunderstandings readily apparent. The relatively lengthy "Biology Background, Terminology" and "Study Context" sections of this paper were arrived at slowly via much misunderstanding, trial, and error, but clarified via critical analysis of the visualizations in this paper.

1.2 Biology Background, Terminology

A Chromosome consists of two intertwined chains of chemicals called nucleic acids (which, in the case of DNA, are limited to Adenine, Guanine, Cytosine, and Thymine - abbreviated A, G, C, and T, respectively). Each nucleic acid in one chain is bonded to a corresponding

- *Alex Bigelow is a student at the University of Utah.*
- *E-mail: alex.bigelow@utah.edu.*

molecule in the opposite chain - A is always matched with T, and C is always matched with G. Because we can always infer the sequence of one strand from the other, we generally use only one strand for notation purposes.

Humans DNA mainly consists of 24 chromosomes: there are two sex-determining chromosomes X and Y, and the others are simply numbered from 1-22. Humans are known as "diploid" organisms, meaning that every human has two of every chromosome (one is inherited from each parent). If we were to line up the sequence from every chromosome from one person, we would call that the person's *genome*.

Fig. 2. Some terminology

A *variant* is defined as a place in the genome that is known to vary from individual to individual. Common variants include single nucleotide polymorphisms (*SNP*) and insertions/deletions (*INDELs*). If both chromosomes in Figure 2 were from the same individual, this person would have SNPs at the sixth, ninth, and twelth positions, and an INDEL at the second base pair. An *allele* is the base (or bases, in the case of an INDEL) a person has for a given variant. The person's *genotype* at the first SNP would be `A/T`. There is a major effort in genetics to catalogue these variants, and new ones are discovered all the time. Because we know which chromosome each allele is on, we know both *haplotypes* in this region (`CG-A-C-C` and `C-T-T-G`). Though every person receives a separate, distinct chromosome from each parent, *recombination*, or the exchange of sections of a chromosome, serves to break down haplotypes from generation to generation. Haplotypes are thus inherited, but through successive generations, they become smaller and smaller.

Another important concept is the idea of a *reference genome*. The reference genome is not an "average" genome, but rather a standardized reference built from several individuals' DNA to which other genomes can be compared. While it would be inaccurate to say that the reference genome defines what is "normally" seen at any location in the human genome, it provides a frame of reference to identify variants. Hypothetically, if the reference genome for the chromosomes in Figure 2 was:

`G C G T T A T T C G T C T G`

our running example would actually have an additional SNP in the last position (where the `G` differs from both `A`s). We would say that the person is *homozygous* for the *alternate allele*. If, in fact, the last reference base was an `A` (matching our example), but it was a known variant, we would say that the person has a homozygous genotype for the *reference allele*. The person is *heterozygous* for the other variants with two mismatching alleles.

A *gene* is a section of the genome that is transcribed into *protein* that is typically several hundred base pairs long. Regions of the genome can be classified as either *coding* or *non-coding*; how a protein is built depends on the DNA sequence of coding regions. Non-coding regions of DNA don't directly translate to protein, but they can influence proteins and cell function in other ways. Typically a variant in a coding region is more *deleterious* (i.e. damage-causing) than a variant in a non-coding region because it can directly alter the shape and/or function of the protein, but that is not always the case. A coding SNP, for example, could cause a benign change in protein structure that doesn't make much of a difference, while a SNP in a nearby (or not-so-nearby) region could dramatically affect how often the protein is generated. Another complication is that variants can have a cumulative effect; a single change may not do much on its own, but a specific combination of variants could have a significant impact.

Sequencing technology attempts to find the entire genetic sequence of an individual; *genotyping* technology only looks at specific locations - usually previously identified variants. Sequencing is much more expensive and time-consuming than genotyping, but with the advent of *next-generation sequencing* (aka "Short read," "Second-generation," "High Throughput" or "NGS" sequencing), it has become quicker and less expensive. Unfortunately, genotyping technologies are limited in their abilities to determine haplotype information - only genotypes can be directly obtained from this technology. Sequencing, however, can provide *phased* (where alleles are assigned to haplotypes with a probability) data. Though this haplotype information is not perfect, it proved particularly useful for the analyses in this paper.

1.3 Study Context

In a previous step in this study, a 3-variant haplotype was identified as a good proxy for a *genetic association* (statistical relationship) between the CASP8 gene and breast cancer. [5] This risk haplotype is seen in approximately 30% of north western europeans. While the individual variants on the haplotype are not believed to have great significance in of themselves, the underlying causal variant or variants are likely to be on the haplotype. For example, if the deletion in Figure 2 were causing some deleterious change to a protein, and we observed the final `T-T-G` part of the haplotype as a good proxy for disease via genotyping a large population, selecting individuals for sequencing that have the `T-T-G` haplotype would give us a better chance of finding the underlying deletion that is contributing to disease.

For this step of the study, 50 individuals were selected from the previous genotyping step who either:

- have homozygous alternate genotypes for the risk haplotype (*cases*) or
- have homozygous reference genotypes for the risk haplotype (*controls*)[1]

This selection was done to remove the haplotype ambiguity inherent in sequencing technology. These individuals were sequenced using the ABI SOLiD sequencing platform, and 2919 SNPs and INDELs were identified in the general CASP8 region of chromosome 2 and the DR4/5 regions of chromosome 8 (which had shown a similar association as CASP8) where at least one individual sequenced differed from the reference genome. An ethnically-matched public data set from the 1000 Genomes Project [1] was also used as a "background" population.

The current stage of this study is to identify which of these 2919 variants (or other known SNPs or INDELs in the region that showed no variation from reference) should be selected for genotyping across a larger population. How to prioritize these variants (outside of obvious, deleterious coding changes) is somewhat of an open question in Biology. There are three patterns that we could look for:

- The single-variant model: we look for a variant that can differentiate cases and controls on its own and is seen in an appropriate percentage of the background. This is the most ideal, but, as is common with complex diseases like cancer, the least likely scenario.

[1]The individuals were actually divided into further subgroups based on their genotypes for other variants identified in the haplotype identification step, but for the purposes of this paper, the generalization into two "case" and "control" groups will suffice

- The "OR" disease model: we look for (potentially rare) variants that contribute to disease risk on their own that happen to cluster around some feature in the genome (i.e. having any one of the identified variants contributes to disease risk by interfering with the same gene or genomic feature).

- The "AND" disease model: we look for specific combinations of variants that could contribute to disease risk together (i.e. each variant on its own has a minimal effect, but a specific combination of these relatively benign variants leads to a larger impact). This is by far the hardest of the three patterns to identify because of the inherent combinatoric complexity.

The scope of this paper is mostly to pursue the first problem, but in a way that can be adapted to the other two.

2 Standard Approaches

2.1 Automated Tools

2.1.1 VAAST [16]

As per its website: "VAAST (the Variant Annotation, Analysis and Search Tool) is a probabilistic search tool for identifying damaged genes and their disease-causing variants in personal genome sequences. VAAST builds upon existing amino acid substitution (AAS) and aggregative approaches to variant prioritization, combining elements of both into a single unified likelihood-framework that allows users to identify damaged genes and deleterious variants.... VAAST can score both coding and non-coding variants, evaluating the cumulative impact of both types of variants simultaneously. VAAST can identify rare variants causing rare genetic diseases, and it can also use both rare and common variants to identify genes responsible for common diseases...."

VAAST is an automated tool that looks at the single-variant and "OR" models of disease: it looks for potentially rare variants in the same gene across individuals that could contribute to disease. If three people each have a distinct variant that disrupts the same gene, VAAST is ideally suited to finding them. Unfortunately (at the time of this writing), support for INDEL and non-coding variants are still in beta and extremely experimental. It also does not really address the "AND" or "multiple insult" model - where only individuals with a specific set of variants (that aren't necessarily rare) show an increased risk.

We have been able to run VAAST successfully on the data set for this project for coding regions only; running with non-coding regions is proving more problematic. In practice, VAAST also seems more geared toward finding single alleles in Mendelian diseases (where a disease is directly caused by a single variant), and looking for the cumulative impact of several variants is also proving more cumbersome. However, VAAST is still very new (we are actually beta testing it - hence many of the difficulties we are experiencing). The results we have obtained from exonic-only runs have not been very informative.

2.1.2 Other Bioinformatics Information

Our collaborators have provided other bioinformatic information about regions of interest (such as conservation scores across species, open chromatin data, etc). This information will likely be useful for all three disease models, and incorporating this information visually is one of the goals of this thesis.

2.1.3 hapConstructor

[7] The "AND" perspective on these variants is a relatively novel approach that does not receive nearly as much attention in the community as the "OR" perspective. This is largely due to the inherent combinatoric nature of the problem, as well as the loss of haplotype information that occurs in sequencing and particularly genotyping technologies.

HapConstructor is a program that was developed in our research group that attempts a brute-force approach to mining haplotypes with the "AND" perspective, but the combinatoric problem prevents it from scaling analyses beyond only a few variants at a time. It was designed with genotyping data in mind, and definitely does not scale to orders-of-magnitude larger sequencing data.

2.2 Derived Data, Visualizations

2.2.1 Manhattan plot

A basic tool in a genetic association study is the *Odds Ratio*, which is defined as:

$$\frac{caseAlleles(2controlIndividuals - controlAlleles)}{controlAlleles(2caseIndividuals - caseAlleles)}$$

This metric gives an idea how unique a variant is to cases or controls, but masks how much data is behind it. In a typical association study, the p-values associated with these odds ratios can be visualized via a Manhattan plot [14]. In an actual Manhattan plot, p-values for Odds Ratios are plotted against genome position so that variants with high statistical significance stand out. This stage of the study does not have enough power (only 50 individuals) to have significant results (otherwise there would be no need for further genotyping of a larger population). Figure 3 shows a scatterplot of odds ratios versus genome position, with the three previously identified variants circled in red:

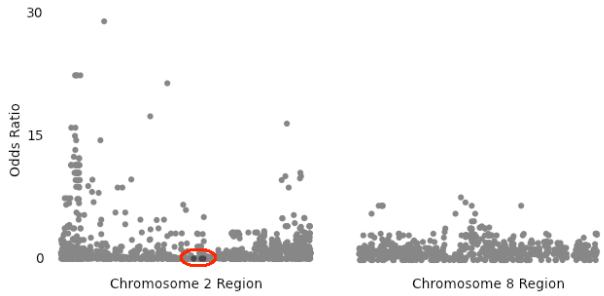


Fig. 3. A scatterplot of odds ratios

The Odds Ratio also suffers from another problem: in the event that one (and only one) control has a homozygous reference genotype, it zeroes out what otherwise could be a large number. For a small data set of only 50 individuals, and especially considering the study design, this statistic is not as useful as it would be in a full association study. Nor does this statistic show us very much about the similarity between variants, let alone similarity to the risk haplotype.

2.2.2 Genome Browser

The standard visualization for sequencing data is a genome browser. This displays information with the x-axis representing position in the genome, and the y-axis or color scheme can visually encode almost anything. These are commonly used to visualize alignment of sequence data, both for verification of the alignment itself, and for verification of downstream analyses' results.

Genome browsers are very effective for many uses, but their drawbacks include an extremely localized viewing window that can severely hamper analyses involving many variants and/or large regions.

2.2.3 Tabular Filtering

The other standard way to look at sequencing data usually involves a tabular approach such as a spreadsheet or a large data-capable tool such as MedSavant [9], and applying filters. This also works very well for many analyses, but there is an implicit reliance on user-defined, potentially biased heuristics, and there is no global sense for the structure of the data.

3 Exploratory Visualizations

3.1 A Naive Data Space

3.1.1 The Zero Problem, Defining Distance

One problem with finding similar variants to these original three lies in the space of the data; most freely-available machine learning toolkits assume Euclidean coordinate spaces and a notion of zero for each feature.[2] Looking at variants individually, a "zero" variant is ill-defined, as is "distance" between variants in terms of their distributional similarity.

While manual implementation of known clustering or PCA algorithms for this data space is an option (especially for future work), a fundamental prerequisite to doing this is an understanding of the space in which this data resides, not to mention a feel for the type of clustering that would make the most sense. Much effort will likely be wasted without preliminary visual exploration of these ideas.

As an exploratory step to see if there are obvious patterns in the data, we considered a 3-dimensional space in which variants are points. The x-coordinate represents the total number of homozygous reference genotypes (i.e. the number of individuals with a homozygous reference genotype) for the variant, the y-coordinate represents the total number of heterozygous genotypes, and the z-coordinate represents the total number of homozygous alternate genotypes. While this representation has an intuitive "zero," it has obvious flaws in that the distance between two points has no meaningful interpretation; two points near each other could have wildly different genotype distributions among individuals, and just happen to have approximately the same number of each type of genotype. Further problems with this space defnition, as well as perceptual issues inherent in 3-dimensional scatter plots, become obvious when the data is plotted (see Figure 4; the red points - if you can find them - are the original 3-variant haplotype).

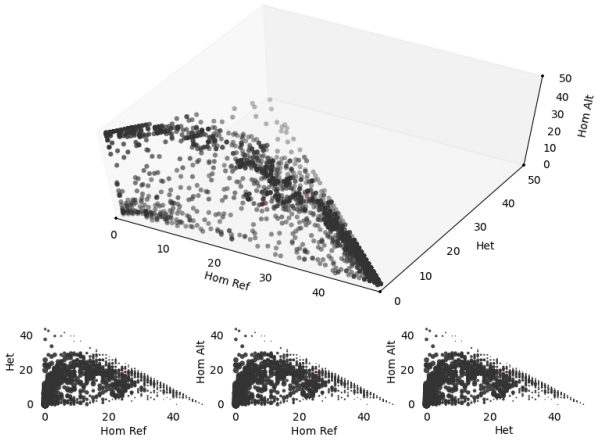


Fig. 4. A 3D Scatterplot of the data in a naive space

The result from this prelimiary plot appears to be a noisy tetrahedron with a slightly convex opposite plane (which we would expect, given the spatial defnition; the plane opposite consists of variants that saw all possible genotypes, the other faces represent variants that only saw two genotypes, and the corners represent variants that only saw one genotype). A weakness of this data space is that even if some kind of structure is observed, extracting meaning from a structure would be very difficult.

To create this scatterplot, we used the Python library matplotlib [8]. While matplotlib is reasonably efficient and user-friendly when creating basic plots, it has a very steep learning curve when it comes to creating custom visualizations and interactions.

[2]We tried using KNIME [3] and SciPy's sklearn [10] to perform automatic machine learning on the data; trying other packages is another area of potential future work

3.1.2 Hierarchical Clustering

Given that there is no obvious notion of "zero" in this data and "distance" between points is ill-defined, one of the first logical approaches is a machine-learning approach that can be independent of both. To experiment with hierarchical clustering, We used GENE-E [6], a program already used in genetics, but for analysis of gene expression data. We attempted to run its hierarchical clustering algorithm with columns as individuals and rows as variants. We used a Pearson Correlation with Single, Complete, and Average linkage for:

- All three genotypes (0 = Homozygous Reference, 1 = Heterozygous, 2 = Homozygous Alternate)
- Homozygous Reference genotypes only (0 = Not Homozygous Reference, 1 = Homozygous Reference)
- Heterozygous genotypes only (0 = Not Heterozygous, 1 = Heterozygous)
- Homozygous Alternate genotypes only (0 = Not Homozygous Alternate, 1 = Homozygous Alternate)

Unfortunately, this approach did not give us the clustering we expected (the resulting dendrograms are far too large to include in this paper). There is some doubt as to its effectiveness in the face of the enormous data and complexity; we were able to manually identify variants that had identical genotype patterns that the clustering algorithms had not put together.

3.1.3 A Correlation Matrix

To see if we could simplify and reduce the data set to make it more manageable, we wrote a program to display and filter the Pearson correlation matrix. Screen shots of this program can be seen in Figure 5. This program displays high correlations in blue, neutral correlations in white, and negative correlations in orange [4]. It also interactively filters out any variants that do not have a correlation above a specific threshold.

Unfortunately, there was not a very significant decrease in the number of variants, even when the threshold reached 1.0 (there were many variants that still had a perfect correlation with at least one other variant other than itself - this, again, is likely a side effect of the small sample size). The reduction in variants as the threshold increases can be seen in Figure 6.

To generate this visualization, we used the Python game library pygame [11]. While it was more open-ended than matplotlib, very straightforward to create a simple visualization, and scaled reasonably well to large amounts of data, it resulted in a somewhat gritty, unprofessional visualization.

3.2 A More Informed Data Space

3.2.1 Inspiration From Graph Regularization

Revisiting the notion of distance between variants, we tried another approach. Instead of defining three distances from a somewhat nebulous concept of "zero," we defined distance between variants as the total number of matching alleles between each genotype. We later refined this metric through several iterations, as will be explained below.

Using this distance metric, we would hope to see a natural separation of variants that are not similar in terms of their distributions across cases and a background population from the 1000 Genomes Project. We considered building a fully-connected graph for the purposes of graph-based machine learning and visualization of the data. Because the graph would be fully connected, propagating labels from seed variants would be a simple matter of computing a single weighted average for each unlabeled node.

There are problems with this idea, however, that are both practical and theoretical. Simply building a fully-connected graph with weighted edges is an $O(n^2)$ operation, not to mention further computation and/or visualization of the graph. The theoretical problems include the typical way graph methods are used in Machine Learning; they are usually used in semi-supervised learning [17], and involve

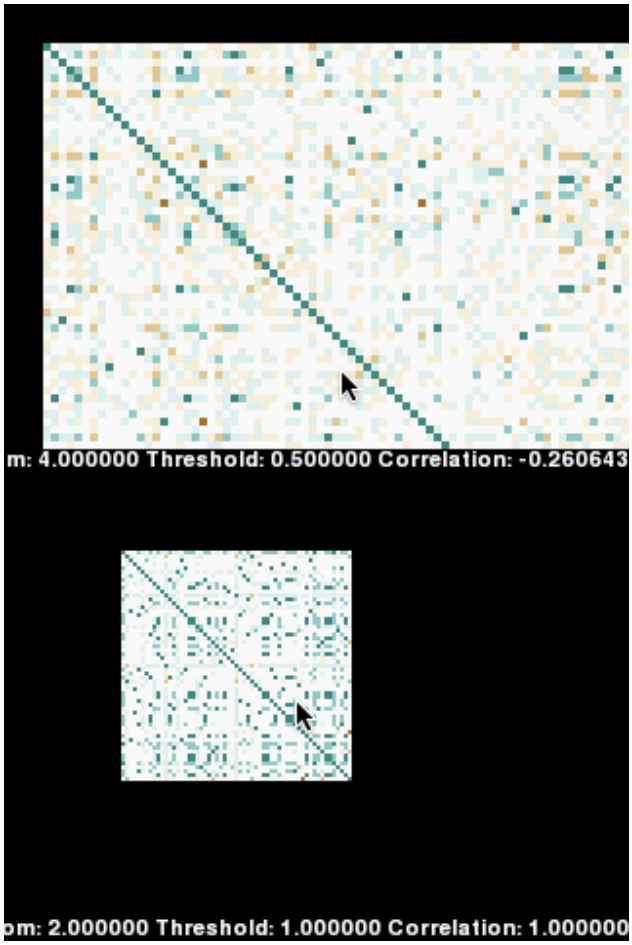


Fig. 5. The correlation matrix exploration app, showing INDEL data only

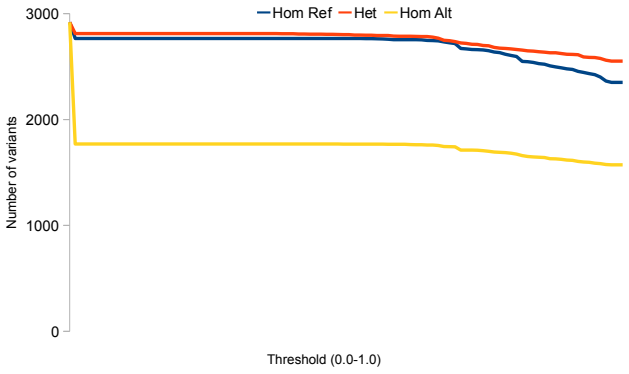


Fig. 6. A plot of the number of remaining variants as threshold is increased

some a priori notion of what labels are to be expected as well as introductory "seed" examples. In this application, we have seed examples, but how they should be labeled (not to mention the other nodes they connect with) is a tough question.

Three possible uses of graphs with this data set:

- Using the traditional semi-supervised application of graphs to machine learning, we could assume two labels: "deleterious" and "protective." Our seed examples would naturally be the variants from our 3-variant deleterious haplotype and variants from the "protective" haplotype also identified by the previous stage of the study. There are a lot of very dangerous assumptions in doing this: we would likely end up doing some kind of interpolation between "deleteriousness" and "protectiveness." Biologically, the notion of a "protective" variant is not as well studied or understood, and a spectrum between the two would probably only make sense in very specific (and likely ill-understood) domains. However, this might be an interesting area for extremely cautious exploration in the future.

- Assuming *no* labels or seed examples, the fully-connected graph could be utilized in an unsupervized learning approach. By iteratively pruning edges below a rising threshold (as well as pruning nodes with no remaining connections), the once-fully-connected graph would begin to "melt" and form distinct clusters that represent groups of variants with high distributional similarity. Further "melting" until only a few points remain would yield the variants that are the most characterstic of their respective groups. This approach could be useful in studies where no case or control groups have been pre-defined. However, it is not particularly pertinent to this study; we already have an idea for both the groups and variants that we are looking for. Another problem with this lies in complexity; assuming i different threshold values, it would take $O(in^2)$ operations to complete.

- A third possibility is an adaptation of the previous idea; we start with our three known variants, and construct a graph only connecting those three variants to every other variant. This graph can then be "melted," by pruning edges below a rising threshold. This yields neighborhoods of variants similar to the original three. Figure 1 (at the top of this paper) is a visualization of this type of graph using the OpenOrd layout plugin to Gephi [2].

3.2.2 Refining The Metric

Figure 7 shows three major iterations of the association metric we developed. We began by simply looking at the proportion of shared alleles (initially across cases and controls, but due to study design considerations and data quality concerns about the controls, we switched to using data from the 1000 Genomes as a background population). We quickly realized a major/minor allele problem; if a major allele co-occurs more frequently with our target minor allele in cases, we really want to be counting the other allele. For example, rs2 and rs3 in Figure 7 have this configuration; we count the 1 as matching the 0 because it co-occurs most in the cases. The graph in Figure 1 shows the resulting graph (with weak edges filtered) from this metric.

The contrived example shows the weaknesses of this first metric; it is very easy for a completely dissimilar variant (rs4) to achieve a score very similar to the scores for variants that are essentially identical (rs3), or show strong correlation in cases (rs2). Due to a misunderstanding[3] of the study design, a causal variant was expected to co-occur only in the cases, and to generally have a random pattern in the background. For this reason, variants were discounted by the number of mismatches in Iteration 2.

One persistent problem with the metric in Iteration 2 was that a variant essentially identical to the target variant (rs3) still got a relatively high score. We don't want to find variants that would co-occur

[3]This key misunderstanding on my part led to further complications that will be discussed in the "Evaluation, Abstraction Breakthrough" section.

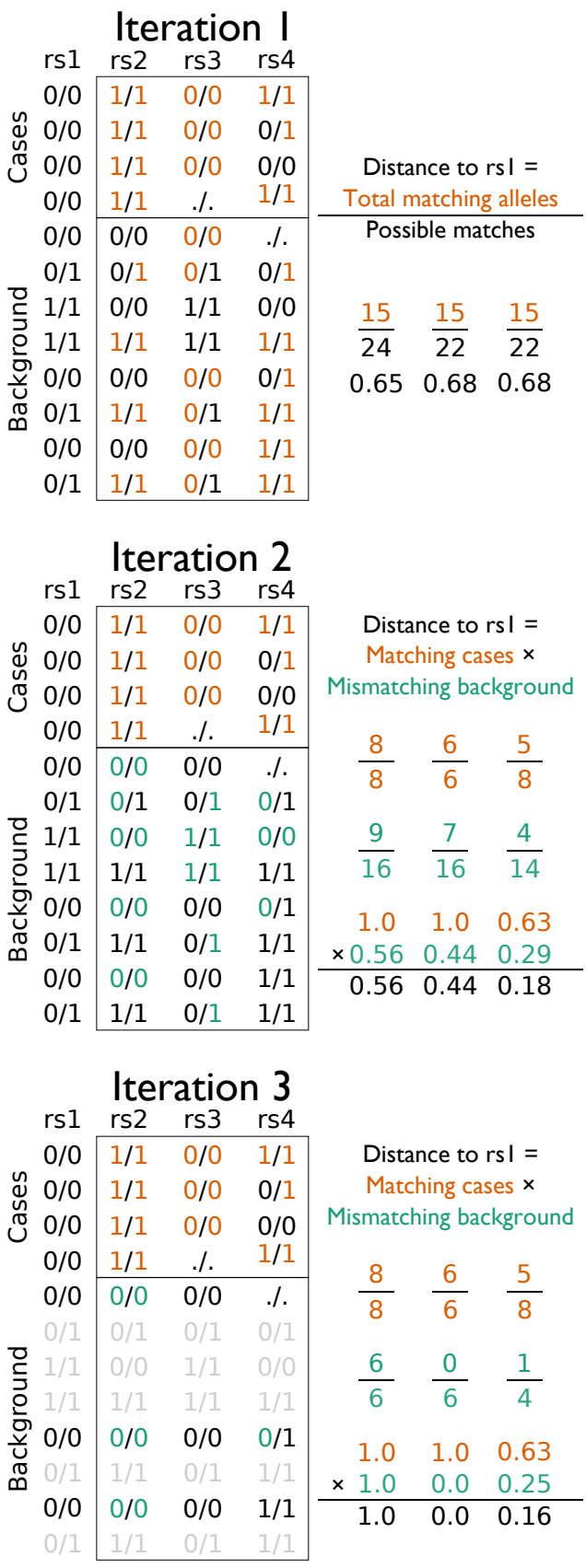


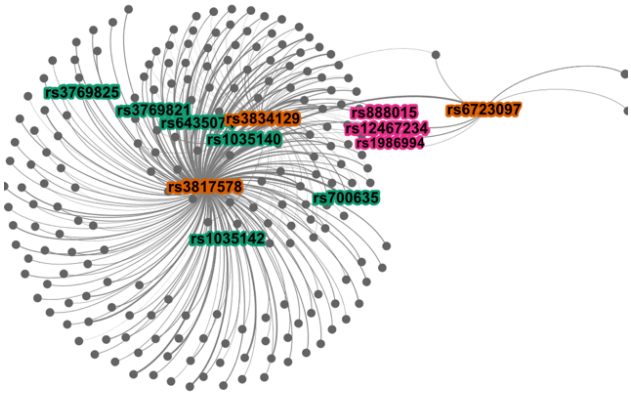
Fig. 7. Three major iterations of an allele sharing association metric. The variants shown here are contrived to emphasize the weaknesses of early metrics.

naturally in just any population; we want to find variants that are abnormally co-occurring in our cases with breast cancer. To reduce this effect, we conditioned the background on individuals that were homozygous for the risk allele to select individuals that have a better fit to how the cases were selected.

3.2.3 Results, Validation

To test the metric from Iteration 3, we created an ad-hoc visualization system using a Tkinter (Python)-based scented widget [15] to interactively pipe data to Gephi. One advantage of this setup was it allowed spreading the computational load across multiple machines; we actually ran the Python widget on a Red Hat Linux machine, and the Gephi visualization on a Macbook. The response was fast for loading data into the graph, however removing nodes and edges was somewhat slow.

The scented widget includes a slider with two histograms; the top histogram represents the visibility of all edges in the graph (all edges with a weight higher than the slider value are visible in the graph; all edges with weaker weights are hidden), and the bottom histogram represents the visibility of all nodes in the graph (a node is only shown if it has at least one visible edge).



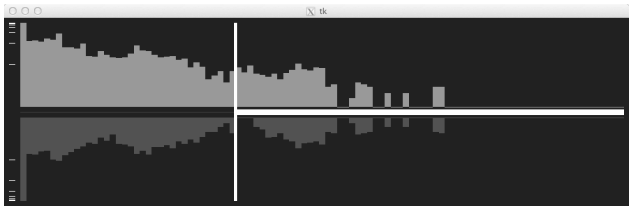


Fig. 8. The graph and widget using the Iteration 3 calculation

The graph in Figure 8 was generated using an ethnically matched subset of the 1000 Genomes data as a background, however, the bimodal distribution seen in the bottom histogram was consistently observed across several populations (British only, British + Utah, and all European subsets). For further validation, null plots [13] were generated by running the visualization using a random selection of "cases" from a pool of combined cases and background individuals. The null plots consistently lacked the bimodal shape (see Figure 9), and the graph growing/melting behavior was distinctly different from the behavior of the actual data. Furthermore, the three same variants shown in pink in Figure 8 were all in the same genomic region upstream from CASP8, each showed a strong connection to all three target variants

regardless of the background population, but none of the three showed strong connections when generating null plots.

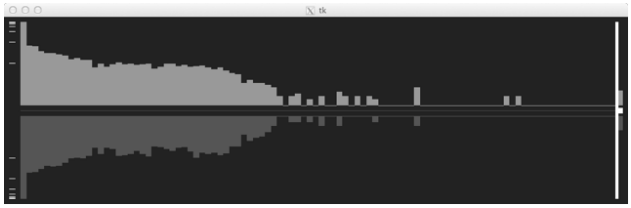


Fig. 9. A null plot of the same calculation

3.2.4 Evaluation, Abstraction Breakthrough

While it is likely that we are seeing some real effect in the data (the three variants did feature prominently in later analyses), it is possible that the patterns observed were simply artifacts from the way that cases were selected. One obvious weakness with a single widget is that it inherently weights each target variant's influence equally, and combines their effects linearly. Another major weakness of this approach is that the definition of distance between variants is extremely nebulous and unintuitive, compounded by the perceptual difficulties inherent in graph navigation. Even if observed patterns are real, interpreting the patterns is a very difficult problem. Upon further reflection and discussion, several pertinent facts about study design and bioinformatics[4] also emerged:

- The risk haplotype was actually not rare; it was observed in approximately 30% of the population
- The variants on the risk haplotype by themselves are actually not very good proxies for the observed association; only the *combination* of the three represented a decent proxy
- NGS data provides phase information about the genotypes that can resolve much of the haplotype/allele ambiguity we encountered in each iteration of the distance metric

A major benefit that emerged from the discussion of this visualization's weaknesses was the idea of creating a "pseudo-genotype" from the risk haplotype: an individual would be assigned a 1 if and only if all three risk alleles were seen together. This way, the search for variants would be reduced to filtering on a single value instead of three inter-dependent values.

3.2.5 Commentary

We had been working with this graph perspective on the data for several months. It had taken a great deal of effort for both parties to understand this representation, and the results also had compelling validation via graphical inference. This model had shaped our thinking. Another strategy that emerged from the discussion was the idea of using variants that had *not* represented the association as an additional filter. However, our thinking had been shaped to the point that our immediate inclination was to continue drawing graphs with a central node representing the haplotype, and additional "hubs" of unassociated variants. The opportunity to abandon the complexities of graph visualization should obviously be taken if a simpler alternative exists, but the visualization had been so compelling that it took several days and deliberate effort to stop thinking about the problem in this context.

3.3 An Even Better Data Space, More Concrete Goals

3.3.1 Abstraction

Once we realized the errors we had been making, it became easy to define exactly what made a variant "interesting," whereas before this was a difficult question to answer directly. Now we were able to state with a new degree of confidence that we are looking for variants that:

- Have a high case allele frequency
- Have a high correlation with the risk haplotype
- Have a low correlation with any unassociated variants
- Have a background allele frequency similar to the 30% haplotype frequency
- Have a low control allele frequency

An additional caveat in the abstraction is to include missing or null data. The likelihood of one or more of these attributes to be missing data is high, but variants may still be interesting if they are missing data.

Defining the context of each of the three disease models (single-variant, the "OR" question, and the "AND" question) also became much easier from this breakthrough. For example, the ability to treat haplotypes as single variants has enormous potential for simplifying the "AND" question. Though the "OR" and "AND" question appear similar because they both involve multiple variants, this abstraction of the single-variant problem makes it apparent that they have fundamental differences that make them respectively more similar to the single-variant question than they are to each other. In many ways, the "AND" question is identical to the single-variant question, with the exception that we are looking at haplotypes instead of single variants. The "OR" question could be thought of as the inverse of the single-variant question: the single-variant question involves looking for a variant that explains the association of a genomic feature, whereas the "OR" question involves identifying a genomic feature that explains a cluster of variants. Because of these realizations, we were able to conclude confidently that each question should be explored using separate, but interconnected tools.

3.3.2 Prototype Sketches

For the single variant model, a sketch of a scatterplot-based design was created in SVG format (see Figure 10). The SVG format showed rapid prototyping potential because a visualization can be sketched by hand in a vector graphics application, and script-based interactivity could be added in a non-invasive way [12]. This format would allow further graphical editing of the visualization, and all visualization components would be defined without using any code. We attempted to load data directly into the SVG file and render it in a web browser, complete with interactivity. However, the size of the (relatively small, compared to whole genome experiments) data and the event-binding approach inherent in SVG interaction made the process intractable.

The SVG format certainly has potential for rapid prototyping of simple interactive visualizations. It also could be used for more data-intensive visualizations, but that will require a different event paradigm.

From this simple sketch, further refinements of the data and interaction abstractions were made. These included a re-prioritization of the data attributes:

- High case allele frequency
- Low control allele frequency
- Background allele frequency similar to the 30% haplotype frequency
- High correlation with the risk haplotype
- Low correlation with any unassociated variants

Other refinements included:

- An ability to filter using an arbitrary number of additional variant-specific attributes to be determined at run-time.[5]

[4]These were some of the erroneous assumptions mentioned earlier

[5]Because of this requirement, parallel coordinates were introduced into the visualization (see Figure 11). The major perceptual danger of parallel coordinates is a cluttered view; they are intended, however, as a scented widget more for the purpose of interactive filtering rather than direct visualization.

- An ability to zoom and define multiple viewing ranges
- An ability to make, compare, and combine multiple selections of variants
- An ability to see the genomic locations of selected variants, as well as any genomic features that they are in (the genomic features, like the variant-specific attributes, can be user-defined and loaded at run-time).

4 Future Work

4.1 This Project

The visualizations created in this paper are only explorations; now that we have a solid abstraction for the single-variant disease model, actual, efficient implementation still needs to be done. The "OR" and "AND" disease models present challenging extensions that also need to be explored, concretely abstracted, and implemented.

4.2 Prototyping Technologies

The alluring possibility of a rapid visualization prototyping via the SVG format remains; the more quickly working visualizations of real data can be created, the more quickly interdisciplinary problems can be understood and facilitated. Rapid visualization prototyping will also allow less time for misunderstanding to fester, making interdisciplinary collaborations more productive and accurate.

5 Conclusion

Over the course of this exploration, we were able to identify the weaknesses of several data-massaging (and resulting visualization) techniques in the space of genetic variations. We were also able to identify strengths and weaknesses of various technologies in this type of visualization. More importantly, we were able to identify uses for, and potential pitfalls of experimental visualization in a computer science-biology collaboration.

When misunderstanding results in an erroneous visualization, the inherent power of visualization warrants immediate evaluation of one's perception of the problem. Rapid, experimental visualization can facilitate understanding and simplification of complex problems across disciplines, especially when those visualizations undergo critical analysis. If a fundamental simplification of a complex problem can be found, it is likely that visualization strategies can and should change dramatically.

Acknowledgments

The author wishes to thank Miriah Meyer and Nicola J Camp. This work was funded in part by a grant from the Avon Foundation for Women.

References

[1] A map of human genome variation from population-scale sequencing, 10 2010.

[2] M. Bastian, S. Heymann, and M. Jacomy. Gephi: An open source software for exploring and manipulating networks, 2009.

[3] M. R. Berthold, N. Cebron, F. Dill, T. R. Gabriel, T. Kötter, T. Meinl, P. Ohl, C. Sieb, K. Thiel, and B. Wiswedel. KNIME: The Konstanz Information Miner. In *Studies in Classification, Data Analysis, and Knowledge Organization (GfKL 2007)*. Springer, 2007.

[4] C. A. Brewer, G. W. Hatchard, and M. A. Harrower. Colorbrewer in print: A catalog of color schemes for maps, 2003. The color schemes for graph visualizations and the correlation matrix app were chosen from colorbrewer2.org.

[5] N. J. Camp, M. Parry, S. Knight, R. Abo, G. Elliott, S. H. Rigas, S. P. Balasubramanian, M. W. R. Reed, H. McBurney, A. Latif, W. G. Newman, L. A. Cannon-Albright, D. G. Evans, and A. Cox. Fine-mapping casp8 risk variants in breast cancer. *Cancer Epidemiol Biomarkers Prev*, 21(1):176–181, Jan 2012.

[6] GENE-E. [Online; accessed 15-Dec-2011].

[7] hapConstructor. [Online; accessed 10-Jan-2012].

[8] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing In Science & Engineering*, 9(3):90–95, May-Jun 2007.

[9] Medsavant. `http://genomesavant.com/medsavant/`, 2011.

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python . *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[11] P. Shinners. Pygame. `http://pygame.org/`, 2011.

[12] C. Viau. [Online; accessed 23-April-2012].

[13] H. Wickham, D. Cook, H. Hofmann, and A. Buja. Graphical inference for infovis. *IEEE Trans. Vis. Comput. Graph.*, 16(6):973–979, 2010.

[14] Wikipedia. Manhattan plot — Wikipedia, the free encyclopedia, 2011. [Online; accessed 15-Dec-2011].

[15] W. Willett, J. Heer, and M. Agrawala. Scented widgets: Improving navigation cues with embedded visualizations. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*, 13:1129–1136, 2007.

[16] M. Yandell, C. Huff, H. Hu, M. Singleton, B. Moore, J. Xing, L. Jorde, and M. Reese. A probabilistic disease-gene finder for personal genomes. *Genome Research*, 21(9):1529–1542, Sept. 2011.

[17] X. Zhu. Semi-supervised learning literature survey, 2008.

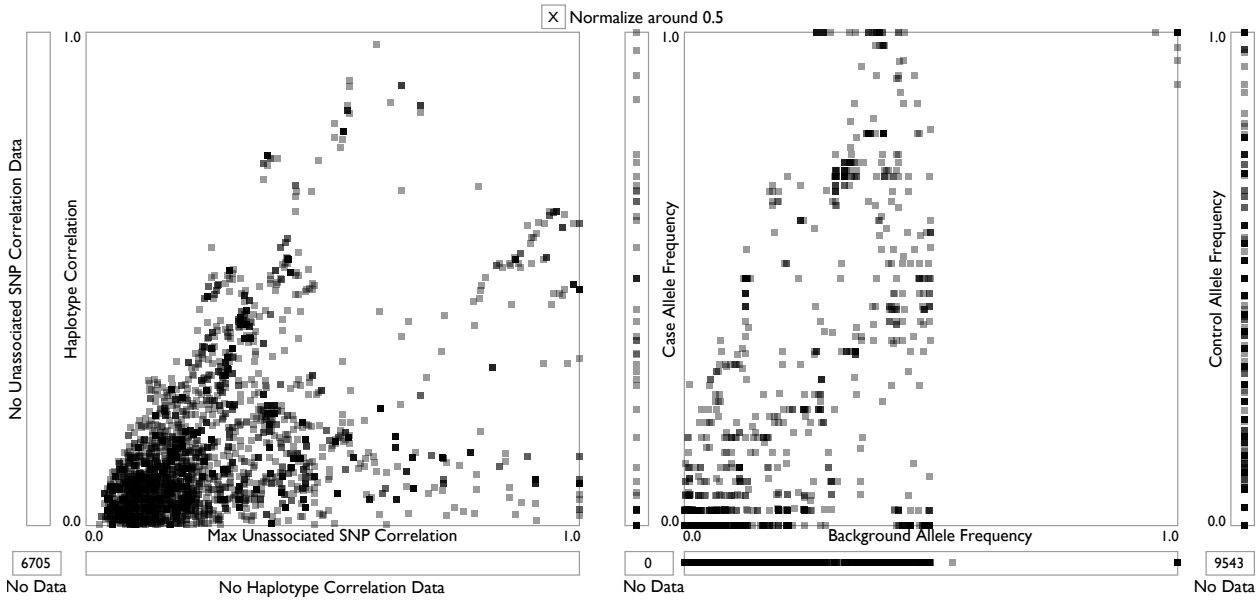


Fig. 10. A first SVG sketch showing actual data, but it did not achieve reasonable interactivity. The regions to the left and bottom of each scatterplot represent points that are null or undefined in the other dimension. The interaction abstraction of the visualization indicated that mousing-over a point would cause the same variant to be highlighted in the other linked views, but this was impossible with SVG event bindings.

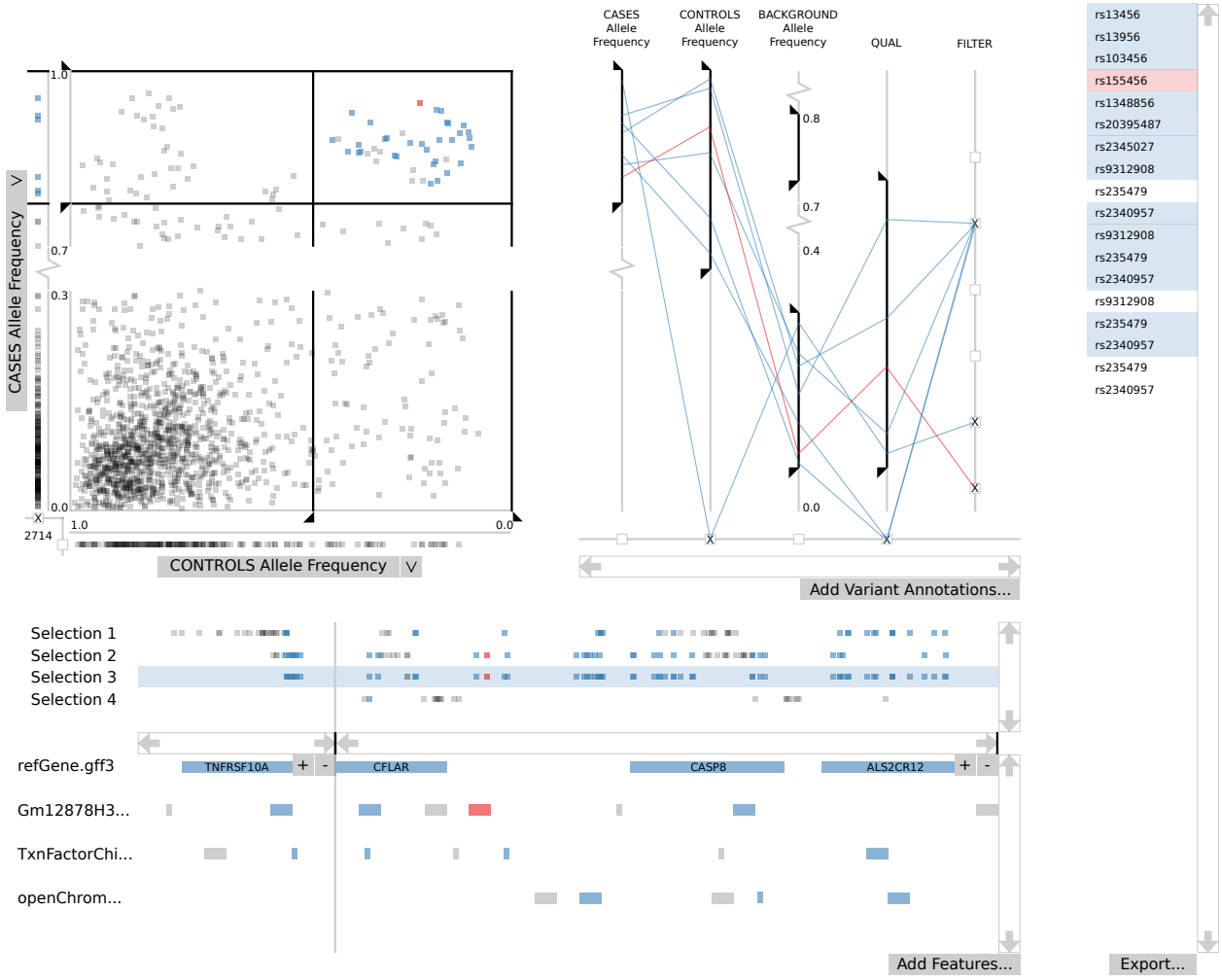


Fig. 11. A final sketch ready for implementation. In addition to linked highlighting, interactive filtering can be achieved via sliders in both the scatterplot and parallel coordinates views. All axes are interchangeable and zoomable (via mouse wheel or contextual menu), and null values are represented by the regions adjacent to scatterplot axes or the baseline of the parallel coordinates view. Additionally, multiple ranges can be viewed via break icons in each axis; each break icon indicates hidden data and a discontinuity in the view. All selected variants are shown in the genome view below; multiple selections can be edited, combined, compared, and deleted via contextual menus. Any features containing a selected variant will be highlighted in the genome view at the bottom. The genome view can show multiple regions/zoom scales via an Excel-style divider. All selected variants are also shown in a table view at the right; specific variants can be manually removed or added via control/command+clicking in this list or shift+lassoing points in the scatterplot view (or alt/option+lassoing points to remove variants). Finally, the list on the right can be exported as a table with any additional attributes that the user has loaded.