## LUMPY: A probabilistic framework for structural variant discovery

Ryan M Layer1  
Aaron R Quinlan\*2,3 and Ira M Hall\*2,3

1Department of Computer Science, University of Virginia, Charlottesville, VA  
2Department of Biochemistry and Molecular Genetics, University of Virginia, Charlottesville, VA  
3Department of Public Health Sciences and Center for Public Health Genomics, University of Virginia, Charlottesville, VA   
Email: Ryan M Layer - rl6sf@virginia.edu; Aaron R Quinlan\*- arq5x@virginia.edu; Ira M Hall\*- irahall@virginia.edu;  
\*Corresponding author

Abstract \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Background:** Comprehensive discovery of structural variation (SV) in human genomes from DNA sequencing requires the integration of multiple alignment signals including read-pair, split-read and read-depth. In a heterogeneous mixture of diploid genomes, a rare variant may be covered by only a few reads. Since reads typically produce only one alignment signal, any SV detection algorithm that considers a single signal, or considers signals individually, will be unable to identify variants with a diffuse signal. However, owing to inherent technical challenges, most existing SV discovery approaches do not integrate signals and consequently suffer from reduced sensitivity at low sequence coverage and for rare SVs.

**Results:** We present a novel and extremely flexible probabilistic SV discovery framework that is capable of integrating any number of SV detection signals including those generated from read alignments or prior evidence. Our framework facilitates this integration by mapping each SV signal to a common abstract representation, and then performs all SV prediction operations at this higher level. This type of abstraction not only allows for efficient and natural signal integration, it is easily extended to consider new signals.

**Conclusions:** We demonstrate that integrating paired-end and split-read signals results in a marked improvement in sensitivity over extant methods in cases where the SV signal is rare (low coverage, low SV allele frequency, or both). We also show that there is no increase in false discovery rate when integrating signals. SV detection improvements in these instances are important in clinical settings where it is especially important to identify variants present in only a small portion of cells in a sample. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### Background

Differences in chromosome structure are a prominent source of human genetic variation. These differences are collectively known as structural variation (SV), a term that encompasses diverse genomic alterations including deletion, tandem duplication, insertion, inversion, translocation or complex rearrangement of relatively large (e.g., >100 bp) segments. While SVs are considerably less common than smaller-scale forms of genetic variation such as single nucleotide polymorphisms (SNPs), they have much greater functional potential due to their larger size, and they are more likely to alter gene structure or dosage.

Our current understanding of the prevalence and impact of SV has been driven by recent advances in genome sequencing. However, the discovery and genotyping of SV from DNA sequence data has lagged far behind SNPs because it is fundamentally more more complicated. SVs vary considerably in size, architecture and genomic context, and read alignment accuracy is compromised near SVs by the presence of novel junctions (i.e., breakpoints) between the “sample” and reference genomes. Moreover, SVs generate multiple alignment signals including altered sequence coverage within duplications or deletions (read-depth), breakpoint-spanning paired-end reads that align discordantly relative to each other (paired-end), and breakpoint-containing single reads that align in split fashion to discontiguous loci in the reference genome (split-reads). These diverse alignment signals are difficult to integrate and most algorithms use just one. Other methods use two signals, but to our knowledge these limit initial detection to one signal and use the other to add confidence, refine breakpoint intervals, or genotype additional samples  [[1](#Xrausch2012b), [2](#Xsindi2012), [3](#Xhandsaker2011)]. The main consequence of limiting detection to one signal is reduced sensitivity. The impact of this is particularly acute in low coverage datasets or in studies of heterogeneous cancer samples where any given rearrangement may only be present in a small subset of cells.

### Results

Here, we present a novel and general probabilistic SV discovery framework that naturally integrates multiple SV detection signals, including those generated from read alignments or prior evidence, and that can readily adapt to any additional source of evidence that may become available with future technological advances.

#### Overview of the probabilistic framework

Our probabilistic framework is based upon a general probabilistic representation of an SV breakpoint that allows any number of SV alignment signals to be integrated into a single discovery process (Methods). An integrative approach allows for more sensitive SV discovery than methods that examine merely one signal, especially when considering heterogenouse samples and low coverage data, because each individual read generally produces only one signal type (e.g., read-pair or split-read, but not both). Moreover, even with high coverage data, integration of multiple signals can increase specificity by allowing for more stringent criteria for reporting a variant call.

We define a breakpoint as a pair of bases that are adjacent in a sample genome but not in a reference genome. To account for the varying level of noise inherent to different types of alignment evidence, we represent a breakpoint with pair of probability distributions spanning the predicted breakpoint regions (Figure 1, Methods). Each position in the two intervals is assigned a probability that represents the relative likelihood that the given position represents one end of the breakpoint.

Our framework provides distinct modules that map signals from each alignment evidence type to our common probability interval pair. For example, paired-end sequence alignments are projected to a pair of intervals upstream or downstream (depending on orientation) of the mapped ends (Figure 1). The size of the intervals and the likelihood at each position is based on the empirical size distribution of the sample’s DNA fragment library. The distinct advantage of this approach is that any type of evidence can be considered, as long there exists a direct mapping from the alignment signal to breakpoint likelihoods. Here we provide three modules for converting SV alignment signals to breakpoint likelihood intervals: paired-end, split-read, and generic. We emphasize that our framework is extensible to possible new alignment signals from forthcoming DNA sequencing technologies  [[4](#Xclarke2009)]. The paired-end module maps the output of a paired-end sequence alignment algorithm (e.g., novoalign  [[5](#Xhercus2013)] or BWA  [[6](#Xli2009a)]), the split-read module maps the output of a split-read sequence alignment algorithm (e.g., YAHA [[7](#Xfaust2012)] or BWA-SW [[8](#Xli2010)]), and the generic module allows users to include SV signal types that do not have a specific module implemented (e.g., a priori knowledge such as known SV, and/or output from copy-number variation discovery tools).

Once all of the evidence from the different classes is mapped to breakpoint intervals, all breakpoints with overlapping intervals are clustered and the probability intervals are integrated to refine the evidence for rearrangement and the predicted breakpoint interval (see Methods for details). Any clustered breakpoint region that contains sufficient evidence (based on user-defined arguments) is returned as predicted SV. Similar to the breakpoint probability, the clustered probabilities give the relative likelihood of a breakpoint. The resolution of the predicted breakpoint regions is improved by trimming the positions with probabilities in the lower (e.g., the lowest 5 percent) percentile of the distribution.

We have implemented this framework into an open source C++ software package (LUMPY, available at https://github.com/arq5x/lumpy-sv) that is capable of detecting SV from multiple alignment signals in BAM alignment  [[9](#Xli2009b)] files from one or more samples.

#### Comparison of discovery performance on simulated datasets

We compared LUMPY’s performance to two other SV discovery packages GASVPro  [[2](#Xsindi2012)] and DELLY  [[1](#Xrausch2012b)]. GASVPro and DELLY were selected because both tools are widely used (DELLY is part of the 1000 Genomes Project analysis), both consider a secondary SV signal along with paired-end alignments (coverage and unmapped/split-reads, respectively), and both have demonstrated improvement over the popular SV tools like Breakdancer  [[10](#Xchen2009)] and HYDRA  [[11](#Xquinlan2010b)].

Detection performance was measured in two different simulated experiments. The first experiment tested each tool’s basic detection capability by simulating 1000 homozygous variants at random locations across human chromosome 10. The second experiment measured the ability of each tool to detect rare variants in a more realistic heterogeneous whole-genome cancer sample. To simulate a cancer sample, reads from an “abnormal” genome that included variants identified by the 1000 Genomes Projects were mixed with reads from an unaltered “normal” genome at varying rates.

Improved sensitivity is crucial for comprehensive studies of genome variation, yet high sensitivity at the cost of an inflated false discovery rate (FDR) is undesirable given the time and cost associated with pursuing the putative biological impact of spurious variation. We measured performance in terms of sensitivity and FDR by comparing the predicted SV breakpoints to simulated breakpoints. Predictions that overlapped simulated variants where considered true positives, and all other prediction were considered false positives.

##### Homozygous variants on a single chromosome simulation

To assess the impact of coverage, SV type, and SV size on the performance of our framework, we simulated a set of experimental genomes using SVsim that included 1000 deletions, duplications, insertions, and inversions randomly placed on chromosome 10 of the human genome (build 37). For each SV type, the variant size ranged from 100bp to 10kb. We used WGSIM to “sequence” each simulated genome at 2X, 5X, 10X, 20X, and 50X haploid coverage.

LUMPY was consistently more sensitive than other approaches in all cases, had a marked improvement at lower coverage, and was able to detect all SV types (although GASVPro does detect duplications or insertions and DELLY does not detect insertions, our tests still ran the tools in those cases and consider all calls that were made). For example, LUMPY detected 31% and 79% of all deletions at 2X and 5X coverage, whereas GASVPro and DELLY detected 4.3%/35% and 3.4%/38.4%, respectively. At lower coverage (i.e., 2X and 5X), LUMPY’s sensitivity was greater than all other approaches across all SV types. At most, LUMPY was 9 times more sensitive than the second most sensitive approach at low coverage (LUMPY 31% vs. DELLY 3.4% for deletions at 2X coverage). At worst, LUMPY was 1.01 times more sensitive for inversions at 5X coverage (LUMPY 90% vs. GASVPRO 82%). At higher (20X and 50X) coverage, LUMPY’s sensitivity advantage persisted; it ranged from 90% to 98.7% across all SV types, whereas GASVPRO and DELLY ranged from 67% to 93% and from 82% to 95%, respectively (excluding the SV types which GASVPRO and DELLY are incapable of detecting).

LUMPY’s FDR was comparable to both GASVPro and DELLY in all but the highest coverage cases. At 2X, 5X, and 10X coverage LUMPY’s FDR did not exceed 0.015% across all SV types. For the same coverage levels and for the SV types that GASVPro and DELLY were able to make predictions, the max FDR was 0.002% and 0.009%, respectively. For duplications LUMPY’s FDR ranged from 0% to 0.06% and DELLY’s FDR ranged from 0% to 0.02%. Neither GASVPro nor DELLY report the ability to detect insertions, but in both cases calls where made. Not surprisingly, the FDR in these cases was much higher, while the FDR for LUMPY was consistent with the other SV types, ranging from 0% to 0.07%. For deletions LUMPY’ FDR ranged from 0% to 0.11%, GASVPro’s FDR ranged from 0% to 0.002%, and DELLY’s FDR ranged from 0% to 0.05%.

##### Heterogeneous tumor simulation

Purely homogeneous DNA samples are rare, especially in a clinical setting where biopsied samples include a mixture of both abnormal and normal tissue. To assess the performance of our framework in this realistic scenario, we simulated heterogeneous samples by pooling reads from an “abnormal” genome and a “normal” genome at varying ratios. The source of the simulated abnormal genome was the human reference genome (build 37) modified (using SVsim  [[12](#Xfaustunpub)]) with 5516 non-overlapping deletions identified by the 1000 Genomes Project. The source of the simulated normal genome the was unmodified human reference genome (build 37). Each genome was “sequenced” using the paired-end read simulator WGSIM  [[13](#Xliunpub)], and the reads from the two genomes were combined to create a single heterogeneous sample. The ratio of the reads in a heterogeneous sample that were from the abnormal genome (SV allele frequency) varied between 5% and 50%, and the total coverage ranged from 10X and 80X. For example, to simulate a sample with a 5% SV allele frequency at 10X coverage, the affected genome was sequenced at 0.5X coverage and the unaffected genome at 9.5X coverage. The two sets of reads were then pooled into a single 10X coverage heterogeneous sample.

LUMPY was more sensitive than GASVPro and DELLY in all cases, and had marked improvement when the coverage of the abnormal genome is low (either lower coverage, low SV allele frequency, or both). For example, at 10X coverage and 20% SV allele frequency LUMPY detects 30% of the SVs, whereas GASVPro and DELLY detect only 9% and 5% of the SVs, respectively. This was an 11-fold increase in sensitivity over the next best method. In general, to achieve the same level of sensitivity as LUMPY, GASVPro and DELLY required evidence from the abnormal genome to occur at twice the rate that LUMPY required. At 40X coverage LUMPY detected 30% of variants when the SV allele frequency was 5%, while GASVPro and DELLY did not reach that level of sensitivity (36% and 33%, respectively) until the SV allele frequency was 10%.

In the heterogeneous simulation the FDR for all tools was low, ranging from 0% to 0.03% (excluding the 10X coverage/0.05 SV allele frequency case where GASVPro made very few calls). While DELLY had the lowest overall FDR, ranging from 0% to 0.004%, LUMPY either matched or beat DELLY’s FDR in cases of low SV allele frequency. Outside of the highest coverage levels and SV allele frequencies, LUMPY maintained an FDR of less than 0.01%. At worst LUMPY’s FDR was 0.03%.

These results indicate that LUMPY’s probabilistic framework afford substantial improvements in discovery sensitivity while maintaining low false discovery rates.

#### Benefits of integrating all signals for SV discovery

LUMPY’s increased sensitivity is driven by the fact that both paired-end and split-read signals are combined during SV discovery. More generally, our framework is capable of pooling any number of signals in order to further increase sensitivity. To our knowledge, while other tools exploit multiple SV signals, they first exploit one signal to drive discovery and then refine candidates with a second signal. An intrinsic limitation of such stepwise approaches is other available signals cannot increase the number of true positive SV calls beyond those candidates identified by the signal used for initial discovery.

It is well-known that variant calling is improved by integrating data from multiple samples  [[3](#Xhandsaker2011), [14](#Xmckenna2010), [15](#Xhormozdiari2011), [16](#Xquinlan2011)], especially when searching for mutations that are rare or private to a single sample. The LUMPY framework naturally handles multiple samples by tracking the sample origin of each probability distribution during clustering. Given that LUMPY can analyze a single human dataset (HG00262 from the 1000 Genomes Project) comprising 104 million read pairs in less than an hour with one thread, we anticipate that simultaneous analysis of tens to hundreds of genomes will be possible with LUMPY using commodity hardware.

### Discussion

We have developed a general probabilistic framework for accurate SV discovery, and have demonstrated that our framework is more sensitive than existing discovery tools across all SV types and coverage levels. Importantly, the increased sensitivity does not come at the cost of excessive spurious SV predictions.

Our framework represents an important technological advance, especially in the context of cancer genomics where sensitivity is crucial to understanding tumor evolution. While highly sensitive methods have been developed for point mutations, similar sensitivity has been challenging for structural variation owing to the technical challenges inherent to characterizing genomic rearrangements from DNA sequence alignments. Our approach greatly simplifies the problem by providing a common framework for representing and integrating breakpoint likelihoods from any number of SV alignment signals. Any signal can be integrated into our framework so long as a breakpoint likelihood can be assigned to each base pair in a candidate breakpoint region. The result is a dramatic increase in SV discovery sensitivity and a corresponding increase in the resolution of the predicted breakpoint interval.

It has not escaped our notice that this general approach can be used to perform probabilistic set theory operations on diverse genomic interval datasets. One immediate application of this framework is interpretation of splicing patterns from RNA-seq data, where sensitivity for low abundance transcripts is paramount, and where there is generally prior evidence for breakpoint positions (i.e., exons). ChIP-seq is another attractive application, as different ChIP-seq datasets are typically analyzed through binary comparisons of peak intervals: that is, do the peaks overlap or not? However, were peaks converted to probability distributions, multiple datasets could be integrated in a probabilistic fashion analogous to how LUMPY interprets SV signals, thus preserving both the spatial and quantitative information underlying the experiment. As a more powerful alternative to traditional peak finding, we envision multi-sample data integration using whole-genome probability distributions, perhaps through extension of existing interval-based software such as our own BEDTools  [[17](#Xquinlan2010a)]. Such toolsets will empower sophisticated probabilistic analyses of inherently complex and nuanced datasets such as ENCODE  [[18](#Xencode2012)]. In general, our framework applies to any data type that can be represented as a probability distribution across genome space.

### Methods

We propose a breakpoint prediction framework that can accommodate multiple classes of evidence from multiple sources in the same analysis. To accomplish this, we define a high-level breakpoint type that represents the consensus breakpoint location from different pieces of evidence. Our framework makes use of an abstract breakpoint evidence type to define a set of functions that serve as an interface between specific evidence subtypes (e.g., paired-end sequence alignments and split-read mappings) and the breakpoint type. Any class of evidence for which these functions can be defined may be included in our framework. To demonstrate the applicability of this abstraction, we defined three breakpoint evidence subtypes: paired-end sequencing, split-read mapping, and a general breakpoint interface.

Since our framework combines evidence from multiple classes, it extends naturally to include evidence from multiple sources. The sources that can be considered in a single analysis may be any combination of evidence from different samples, different evidence subclasses from the same samples, or data sets from known genomic features. We refer to a given data set as a breakpoint evidence instance, and assume that each instance contains only one evidence subtype and is from a single sample. To help organize the results of analysis with multiple samples or multiple instances for a single sample, each instance is assigned an id that can be shared across instances.

#### Breakpoint

A breakpoint is a pair of genomic sequences that are adjacent in a sample genome but not in a reference genome. Breakpoints can be detected, and their locations predicted by various evidence classes (e.g., paired-end sequence alignments and split-read mappings). To support the inclusion of different evidence classes into a single analysis, we define a high-level breakpoint type as a collection of the evidence that corroborates the location and variety of a particular breakpoint. Since many evidence classes provide a range of possible breakpoint locations, we represent the breakpoint’s location with a pair of breakpoint intervals where each interval has a a start position, an end position, and a probability array that represents the likelihood that a given position in the interval is one end of the breakpoint. More formally, a breakpoint is a tuple b = ⟨E,l,r,v⟩ where: E is the set of evidence that corroborates the location and variety of a particular breakpoint; l and r are left and right breakpoint intervals each with values s and e that are the start and end genomic coordinates and p is a probability array where |p| = e - s and p[i] is the relative probability that position s + i is one end of the breakpoint; and v is the breakpoint variety (e.g., Deletion, Duplication, etc.)

If there exists two breakpoints b and c in the set of all breakpoints B where b and c intersect (b.r intersects c.r, b.l intersects c.r, and b.v = c.v), then b and c are merged into interval m, b and c are removed from B, and m is placed into B. The evidence set m.E is the union of the evidence sets b.E and c.E.

A straight-forward method to define breakpoint intervals m.l and m.r would would be to let m.l.s = max(b.l.s,c.l.s), m.l.e = min(b.l.e,c.l.e), similar for m.r. However, if a spurious alignment is merged into a set genuine breakpoints, the resulting breakpoint interval can be “pulled” away from the actual breakpoint. The impact of an outlier can be minimized or eliminated once the full set of agreeing alignments is collected for a given breakpoint, but collecting the full set is complicated by the fact that alignments are considered in-order and outliers typically occur first. To account for this, we define a liberal merge process where m.l.s is the mean start position for the left intervals in m.E, and m.l.e is the mean end position for the left intervals in m.E, similar for m.r.

Once all the evidence has been considered, an SV call s is make for each breakpoint b ∈ B that meets a user-defined minimum evidence threshold (e.g., four pieces of evidence). The boundaries of the breakpoint intervals s.l and s.r are the trimmed mixture distributions of the left and right intervals in b.E. Let s.l.s = min({e.l.s|e ∈ b.E}), s.l.e = min({e.l.e|e ∈ b.E}), and s.l.p[i] = ∑ e∈b.Ee.l.p[i-o] where o is the offset value e.l.s-s.l.s. Similar for s.r. The value at s.l.p[i] (or s.r.p[i]) represents the level of agreement among the evidence in b.E that position i is one end of the breakpoint. The intervals s.l and s.r are then trimmed to include only those positions that are in the top percentile (e.g., top 99.9 percent of values). An outlier in b.E will extend the interval s.l, but the extended region will have little support from other elements in b.E and values of s.p in that region will relatively small and are likely to be removed in the trimming process.

#### Breakpoint Evidence

To combine the distinct SV alignment signals like paired-end and split-read alignments to the general breakpoint type defined above, we define an abstract breakpoint evidence type. This abstract type defines an interface that allows for the inclusion of any data that can provide the following functions: is\_bp determines if a particular instance of the data contains evidence of a break point, get\_v determines the breakpoint variety (e.g., deletion, duplication, inversion, etc.), and get\_bpi maps the data to a pair of breakpoint intervals.

To demonstrate the applicability of this abstraction, we defined three breakpoint evidence instances: paired-end sequencing alignments, split-read mapping, and a general breakpoint interface. Paired-end sequencing and split read mapping are among the most frequently used data types for breakpoint detection, and the general interface provides a mechanism to include any other breakpoint information such as known breakpoints or output from other analysis pipelines. As technologies evolve and our understanding of structural variations improves, other instances can be easily added.

##### Paired-End Sequencing Alignments

Paired-end sequencing involves fragmenting genomic DNA into roughly uniformly sized segments, and sequencing both ends of each segment to produce the sequence pair ⟨x,y⟩. The ends of the pair are aligned to a reference genome R(x) =< o,s,e >, where o = +|- indicates the alignment orientation, and s and e delineate the start and end positions of the matching sequence in the reference genome. To simply the explanation, we let the genome be one contiguous interval of concatenated chromosomes so that all sequences can be referred to by offset only. Translocations can still be identified in this model since the positions on different chromosomes will be far apart. We also assume that both x and y align uniquely to the reference and that R(x).s < R(x).e < R(y).s < R(y).e. While it is often not possible find the exact position of a sequence in the sample genome, it is useful to refer to S(x) =< o,s,e > as the alignment of x with respect to the originating sample’s genome.

Assuming the reads were made on an Illumina platform, pairs are expected to align to the reference genome with a R(x).o = +,R(y).o = - orientation, and at distance R(y).e - R(x).s roughly equivalent to the fragmentation length from the sample preparation step. Any pair that aligns with an unexpected configuration can be evidence of a breakpoint. These unexpected configurations include matching orientation R(x).o = R(y).o, alignments with switched orientation R(x).o = -,R(y).o = +, and an apparent fragment length (R(y).e - R(x).s) that is either shorter or longer than expected. We estimated the expected fragment length to be the sample mean l fragment length, and the fragment length standard deviation to be the sample standard deviation s from the set of properly mapped pairs (as defined by the SAM spec) in the sample data set. Considering the variability in the sequencing process, we extend the expected fragment length to include sizes l \_ vls, where vl is a tuning parameter that reflects spread in the data.

The breakpoint variety for ⟨x,y⟩ can be inferred from the orientation that x and y align to in the reference. If the orientations match, then the breakpoint was caused by an inversion event, and if the R(x).o = - and R(y).o = + then there was a duplication event. When R(x).o = + and R(y).o = -, the breakpoint variety is ambiguous between an insertion and a deletion. This ambiguity is also true for other types of evidence types (e.g., split-read mappings). While it may be possible to determine which event caused the breakpoint in a post-processing step, breakpoint correlation is a complex process and is beyond the scope of this framework. Since we cannot distinguish between the two varieties, any pair with a +∕- orientation configuration is marked as a deletion.

To map ⟨x,y⟩ to breakpoint intervals l and r, the ranges of possible breakpoint locations must be determined and probabilities assigned to each position in those ranges. By convention, x maps to l and y to r, and for the sake of brevity we will focus on x and l since the same process applies to y and r. Assuming that a single breakpoint exists between x and y, then the sign of x determines if l will be upstream or downstream of x. If the R(x).s = +, then the breakpoint interval begins after R(x).e (downstream), otherwise the interval ends before R(x).s (upstream).

The length of each breakpoint interval is proportional to the expected fragment length l and standard deviation s. Since we assume that only one breakpoint exists is between x and y, and that it is unlikely that the distance between the ends of a pair in the sample genome (S(y).e-S(x).s) is greater than l, then it is also unlikely that one end of the breakpoint is at a position greater than R(x).s + l, assuming that R(x).o = +. If R(x).o = -, then it is unlikely that a breakpoint is at a position less than R(x).e -l. To account for variability in the fragmentation process, we extend the breakpoint to R(x).e + (l + vfs) when R(x).o = +, and R(x).s - (l + vfs) when R(x).o = -, where vf is a tuning parameter that, like vl, reflects the spread in the data.

The probability that a particular position i in the breakpoint interval l is part of the actual breakpoint can be estimated by the probability that x and y span that position in the sample. For x and y to span i, the fragment that produced ⟨x,y⟩ must be longer than then distance from the start of x to i, otherwise y would occur before i and x andy would not span i (contradiction). The resulting probability is P(S(y).e-S(x).s > i-R(x).s) if R(x).o = +, and P(S(y).e - S(x).s > R(x).e - i) if R(x).o = -. While we cannot directly measure the sample fragment length (S(y).e - S(x).s), we can estimate its distribution by constructing a frequency-based cumulative distribution D of fragment lengths from the same sample that was used to find l and s, where D(j) gives the proportion of the sample with fragment length greater than j (Appendix  Algorithm [1](#x1-280021) and Algorithm [2](#x1-280052)).

##### Split-Read Alignments

A split-read alignment is a single DNA fragment X that does not uniquely align to the reference genome, but contains a contiguous ordered set of substrings (x1,x2,…,xn) where X = x1x2…xn, each substring aligns uniquely to the reference R(xi) = ⟨o,s,e⟩, and adjacent substrings align to non-adjacent location in the reference genome R(xi).e≠R(xi+1).s + 1 for 1 ≤ i ≤ n - 1. A single split-read alignment maps to a set of adjacent split-read sequence pairs (⟨x1,x2⟩,⟨x2,x3⟩,…,⟨xn-1,xn⟩), and each pair ⟨xi,xi+1⟩ is considered individually.

By definition, a split-read mapping is evidence of a breakpoint and therefore the function is\_bp trivially returns true.

Both orientation and mapping location must be considered to infer the breakpoint variety for ⟨xi,xi+1⟩. When the orientations match R(xi).o = R(xi+1).o, the event was either a deletion or a duplication. Assuming the R(xi).o = R(xi+1).o = +, R(xi).s < R(xi+1).s indicates a gap caused by a deletion and R(xi).s > R(xi+1).s indicated a repeated sequenced caused by a duplication. These observations are flipped when orientations R(xi).o = R(xi+1).o = -. Similar to paired-end alignments, we do not mark breakpoints as insertions since we cannot distinguish between deletions and insertions. When the orientations do not match R(xi).o≠R(xi+1).o, the event was an inversion and the mapping locations do not need to be considered.

The possibility of errors in the sequencing and alignment processes create some ambiguity in the exact location of the breakpoint associated with a split-read sequence pair. To account for this, each pair ⟨xi,xi+1⟩ maps to two breakpoint intervals l and r centered at the split. The probability vectors l.p and r.p are highest at the midpoint and exponentially decreasing toward their edges. The size of this interval is a configurable parameter vs and is based on the quality of the sample under consideration and the specificity of the alignment algorithm used to map the sequences to the reference.

Depending the breakpoint variety, the intervals l and r are centered on either the start or the end of R(xi) and R(xi+1). When the breakpoint is a deletion l is centered at R(xi).e and r at R(xi+1).s, and when the breakpoint is a duplication l is centered at R(xi).s and r at R(xi+1).e. If the breakpoint is an inversion, l and r are both centered either at the start positions or end positions of R(xi) and R(xi+1), respectively. Assuming that R(xi).s < R(xi+1).s, if R(xi).o = + then l and r are centered at R(xi).e and R(xi+1).e, otherwise they are centered at R(xi).s and R(xi+1).s. If R(xi).s > R(xi+1).s, then the conditions are swapped (Appendix  Algorithm [3](#x1-290023)).

##### Generic Evidence

The generic evidence subclass provides a mechanism to directly encode breakpoint intervals using the BEDPE format  [[17](#Xquinlan2010a)]. BEDPE is an extension of the popular BED format that provides a means to specify a pair of genomic coordinates; in this case the pair is a breakpoint. This subclass extends our framework to include SV signal types that do not have a specific subclass implemented yet. For example, a copy number variation prediction algorithm may report segments of the genome that are either duplicated or deleted. This signal can be included in the analysis by expanding the edges of the predicted intervals to create a breakpoints, and encoding that breakpoints in BEDPE format. Each BEDPE entry is assumed to be real breakpoint(is\_bp), the variety is encoded in the auxiliary fields in BEDPE (get\_v), and the intervals are directly encoded in BEDPE (get\_bpi).

##### Simulation

Simulated data was used to compare the sensitivity and false discovery rate of LUMPY to other SV detection algorithms that use multiple signals (GASVPRO and DELLY). Two types of simulations were performed, a homozygous simulation that included reads from a modified chromosome 10 from the human reference genome, and a full-genome heterogeneous simulation that mixed reads from a modified human reference genome and an unaffected human reference genome.

The seed sequence for all the homozygous simulations was chromosome 10 from the human reference genome (build 37). For each SV variety considered (deletions, duplications, insertions, and inversions), we used SVsim to simulate a new version of the seed that contained 1000 randomly placed, non-overlapping variants ranging between 100 bp and 10000 bp. Each simulated genome was sampled to 40X, 20X, 10X, 5X, and 2X coverage.

The seed sequences for the heterogeneous simulation included a modified version of the human reference genome (build 37) and an unmodified version of the human reference genome (build 37). The modified genome was created using SVsim, and included 5516 non-overlapping deletions identified by the 1000 Genomes Project. Each simulation included reads from both genomes at varying rates. We refer to the proportion of reads that were from the modified genome as the SV allele frequency. The simulated SV allele frequencies were 5%, 10%, 20% and 50%, and the simulated coverages were 10X, 20X, 40X, and 80X. For example, in the simulation with 5% SV allele frequency and 10X coverage, the modified genome was sampled at 0.5X coverage and the unmodified genome was sampled at 9,5X coverage. The two sets of reads are then pooled into a single 10X coverage sample.

For all simulations, WGSIM was used to sample pair-end reads with a 150 bp read length, 500 bp mean outer distance with a 50 bp standard deviation, and default error rate settings. Paired-end reads were mapped to the seed sequence with novoalign using default parameters. From the novoalign output, all split-reads and unmapped reads were realigned with the split-read aligner YAHA using a word length of 11 and a minimum match of 15. The novoalign output was used as input to DELLY and GASVPRO, and both novoalign and YAHA output were used as input to LUMPY. In all algorithms, the minimum evidence threshold was four. For LUMPY, the turning parameters min\_non\_overlap was set to 150, discordant\_z was set to 4, back\_distance was set to 20, weight was set to 1, and min\_mapping\_threshold was set to 1. For GASVPro, LIBRARY\_SEPARATED was set to all, CUTOFF\_LMINLMAX was set to SD=4, WRITE\_CONCORDANT was set to true, and WRITE\_LOWQ was set to true. For DELLY, map-qual was set to 1, and the inc-map flag was set.

The reads predicted by each algorithm were compared to the events produced by SVsim. A true positive was a predicted breakpoint that intersected within 50 bp of both ends of a simulated breakpoint, all other predictions were considered to be false positives, and all other missed simulated events were false negatives. Since the output of DELLY is a single interval, we took the 100 bp regions flanking the ends of the predicted interval as the predicted breakpoint.

### Authors contributions

R. Layer developed the algorithms, wrote the all the source code, and performed the all the experiments. A. Quinlan developed the heterogeneous simulation and made substantial contributions to this manuscript. I. Hall conceived the framework, helped devise the experiments, and contributed to this manuscript.

### Acknowledgments

We thank G. Faust for sharing unpublished SV simulation algorithms, M. Lindberg for code development, A. Ritz and S. Sindi for help with GASVPRO, and T. Rausch for help with DELLY.

Funding: This work was supported by an NIH/NHGRI award to AQ (1R01HG006693-01) and an NIH New Innovator Award to IH (DP2OD006493-01)

### References

1.    Rausch T, et al.: DELLY: structural variant discovery by integrated paired-end and split-read analysis. Cell 2012, 28:333–339.

2.    Sindi SS, et al.: An integrative probabilistic model for identification of structural variation in sequencing data. Genome Biology 2012, 13:R22.

3.    Handsaker R, Korn J, et al.: Discovery and genotyping of genome structural polymorphism by sequencing on a population scale. Nature Genetics 2011, 43(3):269–276.

4.    Clarke J, Wu H, et al.: Continuous base identification for single-molecule nanopore DNA sequencing. Nature Nanotechnology 2009, 4:265–270.

5.    novoalign [[http://www.novocraft.com]](%5bhttp://www.novocraft.com%5d).

6.    Li H, Durbin RD: Fast and accurate short read alignment with Burrows–Wheeler transform. Bioinformatics 2009, 25(14):1754–1760.

7.    Faust GG, Hall IM: YAHA: fast and flexible long-read alignment with optimal breakpoint detection. Bioinformatics 2012, 28(19):2417–2424.

8.    Li H, Durbin RD: Fast and accurate long read alignment with Burrows-Wheeler transform. Bioinformatics 2010, 26(5):589–595.

9.    Li H, et al.: The Sequence alignment/map (SAM) format and SAMtools. Bioinformatics 2009, 25:2078–2049.

10.    Chen K, et al.: BreakDancer: an algorithm for high-resolution mapping of genomic structural variation. Nature Methods 2009, 6:677–681.

11.    Quinlan AR, et al.: Genome-wide mapping and assembly of structural variant breakpoints in the mouse genome. Genome Research 2010, 20:623–635.

12.    Faust G: SVsim. [Unpublished].

13.    Li H: WGSIM. [Unpublished].

14.    McKenna A, et al.: The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. Genome Research 2010, 20(9):1297–1303.

15.    Hormozdiari F, et al.: Simultaneous structural variation discovery among multiple paired-end sequenced genomes. Genome Research 2011, 21(12):2203–12.

16.    Quinlan A, et al.: Genome sequencing of mouse induced pluripotent stem cells reveals retroelement stability and infrequent DNA rearrangement during reprogramming. Cell Stem Cell 2011, 9(4):366–73.

17.    Quinlan AR, Hall IM: BEDTools: a flexible suite of utilities for comparing genomic features. Bioinformatics 2011, 26(6):841–842.

18.    ENCODE Project Consortium: An integrated encyclopedia of DNA elements in the human genome. Nature 2012, 489:57–74.

### Figures

#### Figure 1 - LUMPY Workflow

The LUMPY probabilistic SV discovery framework with two example workflows are presented. One workflow (left) uses three different signals (paired-end, split-read, and read-depth) from one sample, as well as prior knowledge regarding known variant sites. The second workflow (right) integrates a single signal type (in this case, paired-end) from three different samples to improve discovery among sensitivity among all three samples.

#### Figure 2 - Sensitivity and false discovery rate in a whole-genome heterogeneous sample

SV discovery sensitivity and false discovery rate (FDR) for LUMPY, GASVPRO, and DELLY for different ratios of abnormal to normal reads (SV allele frequency) at different coverage levels. In all cases, LUMPY was more sensitive than GASVPro and DELLY, and had marked improvement when the coverage of the abnormal genome is low (either lower coverage, low SV allele frequency, or both). In general, to achieve the same level of sensitivity as LUMPY, GASVPro and DELLY required evidence from the abnormal genome to occur at twice the rate that LUMPY required.

#### Figure 3 - Sensitivity and false discovery rate in a homozygous sample on chromosome 10

SV discovery sensitivity and false discovery rate (FDR) for LUMPY, GASVPRO, and DELLY for different SV types across multiple genome coverage levels. LUMPY was consistently more sensitive than other approaches in all cases, had a marked improvement at lower coverage, and was able to detect all SV types. Although GASVPro does detect duplications or insertions and DELLY does not detect insertions, our tests still ran the tools in those cases and consider all calls that were made.