Bionano Solve Theory of Operation: Structural Variant Calling

Document Number: 30110
Document Revision: E
# Table of Contents

**Legal Notice** .................................................................................................................................................3  
**Introduction** ..................................................................................................................................................4  
**Structural variant calling with Bionano Solve 3.3** ..........................................................................................4  
  *De novo assembly* ........................................................................................................................................4  
  *SV calling* .......................................................................................................................................................5  
  *Zygosity classification* .................................................................................................................................6  
  *Confidence modeling* ....................................................................................................................................7  
  *Confidence for insertions and deletions* .........................................................................................................7  
  *Confidence for inversion and translocation breakpoints* .............................................................................7  
  *Confidence for duplication breakpoints* ......................................................................................................7  
  *Copy number variant (CNV) calling* .............................................................................................................7  
  *Masking* .........................................................................................................................................................8  
  *SVMerge (recommended only for NLRS data)* ............................................................................................8  
**Structural variant calling performance** .........................................................................................................8  
  *Structural variant calling performance using simulated data* ................................................................8  
    *Performance for simulated insertions and deletions* .................................................................................8  
    *Performance for other simulated events* ..................................................................................................10  
  *Structural variant calling performance using real data* .........................................................................10  
    *Performance for insertions and deletions* ................................................................................................10  
    *Performance for translocation breakpoints* ............................................................................................11  
    *Performance for inversion breakpoints* ..................................................................................................12  
**Copy number variant calling performance** ..................................................................................................13  
  *Copy number variant calling performance using simulated data* ...........................................................13  
  *Copy number variant calling performance using real data* ..................................................................13  
**Bionano Access integration** .........................................................................................................................14  
**SMAP-to-VCF converter support** ................................................................................................................14  
**Additional Considerations** ..........................................................................................................................14  
**FAQs** ..............................................................................................................................................................15  
**Appendix** .......................................................................................................................................................19  
  *Method for assessing SV calling performance* .........................................................................................19  
  *SV calling performance with simulated data* ..........................................................................................19  
  *SV calling performance with real data* ....................................................................................................20  
  *Method for assessing CNV calling performance* .....................................................................................20  
  *CNV calling performance with simulated data* .......................................................................................20  
  *CNV calling performance with real data* ..................................................................................................21  
  *Mask generation* .........................................................................................................................................21  
    *Compiling common translocation breakpoint list* ..................................................................................21  
    *Compiling annotated segmental duplication regions* ...........................................................................21  
**Technical Assistance** .................................................................................................................................22
Introduction

Structural variation (SV) is a common source of sequence variation, involving deletion, insertion, and rearrangement of genomic material. It has the potential to impact large stretches of sequence, disrupting genes and regulatory elements. Structural variants are associated with genetic disorders and are used as disease markers in clinical diagnosis of diseases such as DiGeorge syndrome and cancer.

The latest Bionano Solve® 3.3 release contains a suite of tools that analyze raw molecule data and generate fully annotated SVs. It has been optimized to work with data generated from the new Direct Label and Stain (DLS) chemistry.

Bionano Solve 3.3 features a haplotype-aware assembler specifically designed to detect and differentiate allelic differences, thus enabling effective and comprehensive detection of homozygous and heterozygous SVs. All major SV types are supported, and extensive validation based on simulated and experimental data showed high detection performance.

The variant annotation pipeline (VAP) annotates the SV calls and provides information key to understanding relevance to a biological question or phenotype of interest (see Bionano Solve Theory of Operation: Variant Annotation, PN#30190 for more detail).

Bionano Solve is fully integrated with Bionano Access® 1.3, which provides a user-friendly interface for streamlining analysis. Bionano Access is useful for managing projects, analyzing run results, and visualizing data.

Structural variant calling with Bionano Solve 3.3

De novo assembly

Bionano’s de novo assembly algorithm is built on the overlap-layout-consensus strategy with a maximum likelihood model for scoring alignments. Following pairwise alignment of the input single-molecule maps, an overlap graph is constructed. Spurious edges are removed and redundant edges collapsed. The assembler outputs the longest paths in the graph and constructs a set of draft consensus maps. The consensus maps are further refined, extended, and merged.

In order to optimize assembly of non-homozygous variants, during the extension stages of the assembly, we analyze molecule-to-genome map alignments, identify clusters of molecules with coordinated disrupted alignment and assemble these clusters separately. This is critical for assembling haplotype maps with large differences and for detecting a wide range of variants. Additionally, we implemented haplotype-aware components to optimize assembly of heterozygous variants. In the last refinement stage, molecules are aligned to a given genome map and clustered into two alleles. The allele-specific molecules are used to generate the final set of allele-differentiated consensus genome maps.

During assembly, large non-unique regions (which we call complex multi-path regions, or CMPRs) in the genome maps are recognized and marked. These regions, often associated with large segmental duplications in the genome, create ambiguity in the assembly graph and are prone to mis-assembly. They are detected in a de novo fashion - genome maps are aligned with each other, and maps that shared significant stretches of sequence but are otherwise divergent are identified. If the CMPR is at least 140 kbp, the maps are split (Figure 1) in order to avoid following ambiguous paths. The labels encompassing the CMPR would be marked in the Mask column in the CMAP output (see CMAP File Format Specification Sheet; PN#30039) and highlighted in Bionano Access. Users are also provided the option to not split these maps.
SV calling

SV calls are obtained by aligning consensus genome maps to a reference using a Multiple Local Alignment algorithm and analyzing the alignments for SV signatures. Pairs of alignments within a map are analyzed and inconsistencies representing possible SV events between the genome maps and the reference are identified.

*Insertions and deletions*

An alignment outlier is defined by two well-aligned regions that flank a poorly aligned or unaligned region. An outlier is identified as a deletion if the reference range in the outlier region is larger than the corresponding range on the map, and an insertion if otherwise.

*Translocation breakpoints*

A fusion point between distant regions of the genome is identified as a translocation breakpoint. Intrachromosomal translocation breakpoints involve regions at least 5 Mbp away from each other on the same chromosome. Interchromosomal translocation breakpoints involve regions on different chromosomes.

*Inversion breakpoints*

Inversion breakpoint calls involve neighboring alignments with opposite orientations. Small inversions (whose inverted regions contained at least four labels) are identified by searching in a limited space for potential inverted alignments. They may be spanned by single genome maps and reported as paired inversion breakpoints ("inversion\_paired"). For other inversion breakpoint calls, there are two linked SMAP entries ("inversion" and "inversion\_partial"), specifying six coordinates of interest (Figure 2). Currently, inversions larger than 5 Mbp are called as intra-chromosomal translocation breakpoints.
Figure 2. Schematic of how inversion breakpoints calls are output in SMAP. “Inversion” (highlighted in blue) and “inversion_partial” (highlighted in red) entries are linked, and together, they encode six coordinates of interest.

Duplications

Duplications are detected based on direct or indirect evidence of duplication within single maps. There is direct evidence when two places on the map align to the same place on the reference. However, the map may not span whole copies of the duplicated sequence. In those cases, the alignments may not overlap on the reference. Based on the alignment signature, one could infer duplication events, even though there is no direct evidence of both duplicated copies on the map. They are output as “duplication_split” events. The duplication detection algorithm supports tandem duplications and non-tandem duplications whose copies are sufficiently close to each other on a given map.

Zygosity classification

Zygosity is a classification of an SV call as homozygous, heterozygous, or unknown. It is currently assigned to only insertion, deletion, translocation breakpoint, and inversion breakpoint calls. It is based on the following two criteria: 1) whether the call in question overlaps another call, and 2) whether the call in question overlaps alignment of another genome map that shows no SV.

In the case of overlap with another call, there is a determination of whether the two calls are likely to represent the same allele. An SV call is categorized as homozygous if there is no overlapping alignment or the same SV is called on another genome map. An SV call is categorized as heterozygous if there is overlapping alignment or a different SV is called on another genome map. If both another alignment and a different SV or multiple different SVs are present at the same location, zygosity is set as ‘unknown’. For this purpose, different sets of criteria are applied for different SV types. For insertion and deletion calls, they must have at least 80% overlap and size similarity. For translocation breakpoint calls, the breakpoints must be within 100 kbp and on the same strand and chromosomes. For inversions breakpoints, the breakpoints must be within 50 kbp and on the same strand and chromosomes. Zygosity is not currently assigned to duplication calls.
Confidence modeling

Confidence for insertions and deletions

The confidence of an insertion or deletion call, which ranges from 0 to 1, reflects an estimate of the probability of the call being a true positive. It takes into account the SV size, the non-normalized p-value (log10) of the two well-aligned regions and the non-normalized log-likelihood ratio of the poorly aligned or unaligned region. The recommended cutoff is 0.1 for insertions and deletion calls.

Confidence for insertion and deletion calls <500 bp in size is undefined and set to -1.

Confidence for inversion and translocation breakpoints

Inversion and translocation confidence modeling was performed based on a supervised training approach, anomaly detection. In this case, we trained our classifier to detect false positive (FP) calls as anomalies relative to simulated true positive (TP) calls.

For modeling, we constructed an initial set of predictors likely related to confidence. The predictors included information about the molecule-to-map and map-to-reference alignments. For each of the initial predictors, we performed a t-test to check whether there was a statistically significant difference (p < 0.05) between TP and FP calls. The predictors that showed significant differences were kept, and the highly correlated predictors were removed. The classifier was trained on four assemblies based on simulated data.

The confidence score for a translocation breakpoint call, which ranges from 0 to 1, correlates with the Mahalanobis distance of the call to the mean of the empirical distribution constructed based on predictor values from the set of TP calls. The squared Mahalanobis distance is calculated as:

\[ D^2 = (x - m)^T C^{-1} (x - m) \]

where \( x \) is a vector of predictor values for the query call, \( m \) is a vector of means of predictors from the TP calls used for training, and \( C \) is the variance covariance matrix of the predictors. The recommended cutoffs are 0.1 for translocation and 0.01 for inversion breakpoints calls.

Confidence for duplication breakpoints

Currently, confidence for duplications is undefined and set to -1.

Copy number variant (CNV) calling

Copy number analysis is performed as part of the de novo assembly process; the pipeline is detailed in Introduction to Copy Number analysis (PN#30210). Briefly, a molecule-to-reference alignment is generated as a part of the assembly process and used as input to the copy number analysis pipeline. After scaling the raw molecule coverage profile based on the expected autosome copy number of two and normalizing the profile using control data, the pipeline segments the genome based on detected changes in the underlying copy number state. CNV calls are output and annotated with confidence scores. The normalized coverage profile is plotted in Bionano Access, and the CNV calls can be sorted and filtered in the Copy Number tab.
Masking

Using a BED file provided by Bionano, insertion and deletion calls overlapping N-base gaps in the reference and putative FP translocation breakpoint calls can be masked. The former would be annotated with a suffix "_nbase" in the SV type. The latter would be annotated with a suffix "_common" or "_segdupe" in the SV type, depending on whether they overlap with common FP calls or annotated segmental duplication regions, respectively. For example, it includes selected sub-centromeric and sub-telomeric regions that are prone to generating putative FP translocation breakpoint calls. We also provide BED files for hg38, BssSI, and DLE-1. Details on custom BED generation are described in the Appendix and FAQ sections.

SVMerge (recommended only for NLRS data)

SVMerge helps users to take advantage of having two single-enzyme datasets (for example, BspQI and BssSI). It currently supports merging of insertion, deletion, translocation breakpoint, and inversion breakpoint calls. SVMerge provides several potential benefits. The complementary nature of the two enzymes helps improve sensitivity compared to one-enzyme detection. Cross-confirmation by two independent datasets provides a useful means of validating SV calls. SVMerge also improves SV breakpoint accuracy. For insertions and deletions, SV size estimates are expected to be more accurate. The confidence scores for insertions and deletions reflect whether just one or both enzymes support an SVMerge call.

Briefly, SVMerge examines SV calls from single-enzyme assemblies and evaluates whether the calls are overlapping. The software merges the overlapping calls and outputs refined breakpoint coordinates.

As discussed in the Bionano Solve 3.2.1 Release Notes (PN#30221), the use of SVMerge with DLS data is not recommended (also see the FAQ section below).

Structural variant calling performance

For performance validation, we took two complementary approaches. We used simulated data and real data to assess SV detection performance. Detailed methods are included in the Appendix section.

Structural variant calling performance using simulated data

Performance for simulated insertions and deletions

DLE-1 molecules were simulated at 70X effective coverage from unedited and edited versions of the hg19 reference genome (with 1600 insertions and 1600 deletions from 200 bp to 1 Mbp) and used for assembly and SV calling with the Bionano Solve pipeline (Figure 3).
Figure 3. Heterozygous insertion and deletion calling performance with simulated 70X DLE-1 data. Top: sensitivity data; bottom: positive predictive value (PPV) data.
Performance for other simulated events

Similarly, DLE-1 molecules were simulated from unedited and edited versions of the hg19 reference genome (with translocated fragments, or transposition events, inversions, and duplications of different sizes) and used for assembly and SV calling with the Bionano Solve pipeline. The performance data are shown in Figure 4.

![Sensitivity for other heterozygous SV events](image)

Figure 4. Heterozygous sensitivity of various SV types with simulated 70X DLE-1 data.

Structural variant calling performance using real data

Performance for insertions and deletions

CHM1/13 analysis was performed on an *in silico* mixture of CHM1 and CHM13 DLE-1 datasets. SV calls were compared from pure CHM1 and CHM13 assemblies against SV calls from the mixture. Overall performance is presented for SVs larger than 700 bp (Table 1).

<table>
<thead>
<tr>
<th>Size cut-off</th>
<th>Type</th>
<th>Expected zygosity in mixture</th>
<th>In silico mixture</th>
<th>Individual assemblies</th>
<th>Fraction captured (%)</th>
<th>PPV (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>700 bp</td>
<td>Insertions</td>
<td>Homozygous</td>
<td>1,975</td>
<td>1,963</td>
<td>99.4</td>
<td>97.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Heterozygous</td>
<td>1,737</td>
<td>1,480</td>
<td>85.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Deletions</td>
<td>Homozygous</td>
<td>695</td>
<td>687</td>
<td>98.9</td>
<td>97.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Heterozygous</td>
<td>1,189</td>
<td>1,085</td>
<td>91.3</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Insertion and deletion (> 700 bp) calling performance from CHM1/13 datasets using DLS.
CEPH trio BspQI datasets used in Mak et al\(^1\), were re-analyzed, and the resulting SV lists were compared with the published SV lists (Table 2\(^2\)). Overall, we detected most of the published calls and made a large number of new SV calls. For example, we detected 93\% of the published deletions in NA12878 and made an additional 764 deletion calls.

Table 2: Insertion and deletion calling performance from CEPH trio datasets using NLRS.

<table>
<thead>
<tr>
<th>SV type</th>
<th>Sample</th>
<th>Number (%) of Mak et al. calls detected</th>
<th>Number (%) of calls that were new</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deletions</td>
<td>NA12878</td>
<td>386 (93%)</td>
<td>794 (74%)</td>
</tr>
<tr>
<td></td>
<td>NA12891</td>
<td>337 (81%)</td>
<td>802 (78%)</td>
</tr>
<tr>
<td></td>
<td>NA12892</td>
<td>310 (80%)</td>
<td>721 (78%)</td>
</tr>
<tr>
<td>Insertions</td>
<td>NA12878</td>
<td>382 (89%)</td>
<td>2089 (88%)</td>
</tr>
<tr>
<td></td>
<td>NA12891</td>
<td>355 (81%)</td>
<td>1949 (88%)</td>
</tr>
<tr>
<td></td>
<td>NA12892</td>
<td>336 (78%)</td>
<td>2011 (89%)</td>
</tr>
</tbody>
</table>

Performance for translocation breakpoints\(^2\)

Nine samples with annotated single translocations were tested for translocation calling performance (Table 3). We found the expected translocation breakpoints in eight out of nine samples. For the remaining sample, only 10\% of the cells contained the expected translocation based on karyotyping results. Reciprocal breakpoints were found in six out of eight samples for which expected translocations were found. In addition, putative FP calls were masked and/or filtered (Table 4).

Table 3. Translocation calling performance from real datasets (at close to or less than 70X effective coverage).

<table>
<thead>
<tr>
<th>Sample</th>
<th>Diagnosis</th>
<th>Annotation</th>
<th>Sample prep</th>
<th>Found expected breakpoint?</th>
<th>Found reciprocal breakpoints?</th>
</tr>
</thead>
<tbody>
<tr>
<td>GM16736</td>
<td>Deafness with DNA repair deficiency</td>
<td>FISH: t(9;22)</td>
<td>OptiDNA SVMerge and plug lysis BspQI</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>GM21891</td>
<td>Prader-Willi syndrome</td>
<td>FISH: t(4;15)</td>
<td>OptiDNA SVMerge and plug lysis BspQI</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Sample01</td>
<td>CML</td>
<td>FISH: t(9;22)</td>
<td>Plug lysis BspQI</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Sample02</td>
<td>CLL</td>
<td>FISH: t(11;14)</td>
<td>Plug lysis BspQI</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Sample03</td>
<td>CML</td>
<td>FISH: t(9;22)</td>
<td>Plug lysis BspQI</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Sample04</td>
<td>CML</td>
<td>Kary.: t(9;22)</td>
<td>Plug lysis BspQI and BssSI</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Sample05</td>
<td>AML</td>
<td>Kary.: t(7;11)</td>
<td>Plug lysis BspQI</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Sample06</td>
<td>AML</td>
<td>Kary.: t(8;21)</td>
<td>Plug lysis BspQI</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Sample07</td>
<td>AML</td>
<td>Kary.: t(4;5)</td>
<td>Plug lysis BspQI and BssSI</td>
<td>No</td>
<td>N/A</td>
</tr>
</tbody>
</table>


\(^2\) The data were presented in a previous version of this document and are included here for completeness.
Cell lines GM16736 and GM21891 obtained from the Coriell Institute for Medical Research each contained one known translocation t(9;22) and t(4;15), respectively. Based on 70X effective coverage assemblies, we found the expected translocation breakpoints for both samples and additional translocation breakpoints. The additional breakpoint in GM21891 was filtered out after applying a confidence threshold of 0.1. An additional breakpoint with confidence score of 0.19 remained in the GM16736 BssSI assembly, which may be an unannotated translocation call in the sample. The BssSI masks were generated based on a small control database; thus, they may be less effective than the BspQI masks.

Table 4. Translocation breakpoint call masking and filtering for Coriell samples GM16736 and GM21891.

<table>
<thead>
<tr>
<th>Performance for inversion breakpoints²</th>
<th>Coriell (BspQI)</th>
<th>Coriell (BssSI)</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Translocation calls</td>
<td>GM16736</td>
<td>GM21891</td>
</tr>
<tr>
<td>Known Translocation</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Remaining calls after masking</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>As expected?</td>
<td>Yes</td>
<td>1 add. call</td>
</tr>
<tr>
<td>As expected after applying confidence threshold &gt;= 0.1</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 5. Inversion calling performance from real datasets (at close to or less than 70X effective coverage).

<table>
<thead>
<tr>
<th>Sample</th>
<th>Diagnosis</th>
<th>Annotation</th>
<th>Sample prep</th>
<th>Found expected inversion?</th>
</tr>
</thead>
<tbody>
<tr>
<td>GM19238</td>
<td>Phenotypically normal</td>
<td>Inv(15q13.3)</td>
<td>Plug lysis BspQI</td>
<td>Yes</td>
</tr>
<tr>
<td>GM14266</td>
<td>Micrognathia</td>
<td>Inv(4q34.2-35.2)</td>
<td>OptiDNA SVMerge</td>
<td>Yes*</td>
</tr>
<tr>
<td>GM21074A</td>
<td>Developmental delay</td>
<td>Inv(2p23-q31)</td>
<td>OptiDNA SVMerge</td>
<td>Yes*</td>
</tr>
</tbody>
</table>

*Called as intra-chromosomal translocation breakpoints, since those inversions were larger than 5 Mbp.
Comparison to Valor calls

For the NA12878 sample, we compared Bionano calls to inversion calls predicted by a new NGS based algorithm. Recently, Rasekh et. al.\(^3\) described a novel algorithm, Valor, to detect large inversions based on “split clone” signatures. Out of the 18 inversion calls predicted by Valor, 15 overlapped with Bionano calls. Bionano detected an additional 82 putative inversion regions.

Copy number variant calling performance

For performance validation, we took two complementary approaches. We used simulated data and real data to assess SV detection performance. Detailed methods are included in the Appendix section.

Copy number variant calling performance using simulated data

High sensitivity and PPV were observed for events > 500 kbp. Generally, increasing coverage increases the signal-to-noise ratio; however, sensitivity for simulated events did not significantly increase beyond 70X (Figure 5). Higher PPV was observed at lower coverage levels, but sensitivity was significantly lower at those coverage levels.

Copy number variant calling performance using real data

Copy number events of different sizes and zygosity states were expected in five well-characterized samples. The smallest expected event (160 kbp) was not detected in either BspQI or BssSI data; all other events were detected.

---

Table 6. CNV calling performance from real datasets (at close to or less than 70X effective coverage).

<table>
<thead>
<tr>
<th>Chr</th>
<th>Size</th>
<th>Type</th>
<th>Zygosity</th>
<th>Detected in BspQI?</th>
<th>BssSI?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>chr2</td>
<td>160 kbp</td>
<td>Duplication</td>
<td>Heterozygous</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>chr2</td>
<td>380 kbp</td>
<td>Duplication</td>
<td>Heterozygous</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Sample 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>chr3</td>
<td>540 kbp</td>
<td>Duplication</td>
<td>Heterozygous</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Sample 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>chr6</td>
<td>1 Mbp</td>
<td>Deletion</td>
<td>Heterozygous</td>
<td>Yes</td>
<td>N/A</td>
</tr>
<tr>
<td>Sample 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>chr17</td>
<td>18 Mbp</td>
<td>Duplication</td>
<td>Homozygous</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>chr8</td>
<td>75 Mbp</td>
<td>Duplication</td>
<td>Homozygous</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Sample 5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>chr6</td>
<td>46 Mbp</td>
<td>Deletion</td>
<td>Heterozygous</td>
<td>Yes</td>
<td>N/A</td>
</tr>
<tr>
<td>chr18</td>
<td>65 Mbp</td>
<td>Duplication</td>
<td>Heterozygous</td>
<td>Yes</td>
<td>N/A</td>
</tr>
<tr>
<td>chr11</td>
<td>67 Mbp</td>
<td>Duplication</td>
<td>Heterozygous</td>
<td>Yes</td>
<td>N/A</td>
</tr>
<tr>
<td>chr1</td>
<td>106 Mbp</td>
<td>Duplication</td>
<td>Heterozygous</td>
<td>Yes</td>
<td>N/A</td>
</tr>
</tbody>
</table>

**Bionano Access integration**

With Bionano Access, users are able to select from the dropdown menus the set of assembly parameters for de novo assembly, depending on the application. Bionano Access enables import, filtering, and visualization of SV and copy number variant calls. Users can import a mask file to annotate insertion and deletion calls and filter out likely false positive translocation calls. We provide six versions of the mask to support the (hg19 and hg38) human reference assemblies and the enzymes recommended for human analysis (BspQI, BssSI, and DLE-1). Users can also import custom gap files.

**SMAP-to-VCF converter support**

The python-based SMAP-to-VCF converter supports conversion of insertion, deletion, translocation breakpoint, and inversion breakpoint calls. The resulting VCF output is dbVar-compliant. The SV breakpoints are output as the midpoint between the last aligned label and the next label. The uncertain breakpoint is indicated in the CIEND and CIPOS fields.

Consistent with the VCF v4.2 format, the VCF file output passed preliminary testing with vcf-tools/vcf-validator, and the VCF headers have been updated accordingly. The information that is added to the output includes the reference accession, orientation information for translocations, and additional dbVar-specific tags.

**Additional Considerations**

SV sensitivity generally improves with depth of coverage. The minimum recommended effective coverage (defined as the product of raw coverage and molecule alignment rate) is 70X, assuming that the quality and average length of the input molecules are good. More coverage will yield small additional sensitivity with diminishing returns beyond 100X.
In order to resolve the alleles, haplotype-aware assemblies require more processing time, relative to non-haplotype-aware assemblies. With typical good-quality molecules, one should expect 2X to 2.5X increase in processing time.

The Bionano Solve pipeline was validated on human diploid samples. When running more complex samples (e.g., cancer) or non-human samples with the haplotype-aware assembler (with haplotype-aware assembly parameters selected), we recommend that the SV calls be validated.

FAQs

1. How does coverage affect SV calling performance?

Based on consideration of SV calling performance, runtime, and system throughput, we recommend a minimum of 70X effective coverage for haplotype-sensitive assembly for all SV types. Sensitivity does increase with coverage, but PPV is comparable across coverage levels (the lowest coverage level tested was 40X effective coverage).

2. What is the sensitivity to small insertions and deletions?

We recommend that users focus on insertions and deletions larger than 500 bp.

2. Why do alignment boundaries appear to be off sometimes?

Based on visual inspection, we noticed cases where there was room for improvement in the accuracy of the alignment boundaries. Even though we correctly detected that there was an event, there was an impact on the SV boundaries for these calls. We have optimized parameters to improve the accuracy of SV boundaries and maintain overall performance.

3. Do we classify translocations?

A translocation can be balanced or unbalanced, and reciprocal or non-reciprocal. We currently detect single translocation breakpoints and do not attempt to further classify them. Also, orientation information is not currently encoded in SMAP. Users may infer the orientation when visualizing the calls in Access. Additional custom secondary analysis may aid pairing and classification of translocation breakpoints.

4. What are the different versions of cluster parameters and assembly parameters files?

Different versions of cluster parameters and assembly parameters files were created for supporting different platforms and different applications. Cluster parameter files are distributed with the python assembly pipeline; assembly parameter files are distributed with RefAligner. To facilitate selection, dropdown menus are available in Access when an assembly is set up.

5. How should label SNR be thresholded?

For human assemblies, we recommend label SNR-thresholding based on alignment to the reference. This is performed during the auto-noise stage in the assembly process. For non-human assemblies, we recommend histogram-based SNR-thresholding. Both methods are implemented as part of the Bionano Access workflow and do not need to be manually run.

6. How are reciprocal translocation breakpoints detected?

Each translocation breakpoint of a reciprocal translocation is independently detected. We currently do not pair potential reciprocal translocation breakpoints.
7. How does the assembler handle ambiguity associated with segmental duplication regions?

Large segmental duplication regions, a type of CPMRs, appear at least twice in the genome and are connected to different sequences (potentially on different chromosomes). If a given segmental duplication region is larger than an average molecule’s length, the assembler likely does not have enough information to resolve the inherent ambiguity in the connectivity. Maps containing CPMRs larger than 140 kbp are split by default; there is also the option to not split those maps.

8. How are masked translocation breakpoint calls annotated?

A suffix (“_common” or “segdupe”) would be appended to any masked translocation breakpoint call in the SMAP SV detection output. A suffix of “_common” indicates that the call overlaps with translocation breakpoints detected in genomes not known to contain translocations. A suffix of “_segdupe” indicates that the call overlaps annotated segmental duplication regions. If a breakpoint overlaps with both types of masked regions, “_common” would be appended.

9. How to create a custom mask?

The masks are in BED format (https://genome.ucsc.edu/FAQ/FAQformat#format1). They include three components: 1) N-base gaps, 2) annotated segmental duplication regions, and 3) translocation breakpoints detected in genomes not known to contain translocations. The list of N-base gaps can be obtained from analyzing a given reference fasta file and outputting locations of the N bases (this is output by the in silico digestion tool included in Bionano Access). The list of annotated segmental duplication regions (> 50 kbp) for human was obtained from the UCSC Genome Browser database. We further verified map-level similarity for these regions. Using a database of control samples, we defined a list of translocation breakpoints that we detected in more than a specified number of genomes.

10. How are translocation and inversion confidence scores different from insertion and deletion confidence scores?

Confidence scores for insertions and deletions are computed as PPV estimates, or how likely a given call is a TP. Confidence scores for translocation and inversion breakpoints are computed as p-values, indicating how likely a given call is different (in Mahalanobis distance) from known TP calls. They require different thresholds.

11. What types of SV calls does SVMerge merge?

SVMerge currently supports merging of insertion, deletion, translocation breakpoint, and inversion breakpoint calls from NLRS data. SVMerge does not merge duplication calls but does carry forward the single-enzyme calls in the output. SVMerge is not recommended for DLS data.

12. What does SVMerge do if a call from one assembly can be merged with more than one call from another assembly?

SVMerge attempts to merge all calls that satisfy the merging criteria. For example, if a call from the BspQI assembly could be merged with two separate calls from the BssSI assembly, there would be two merged calls in the output involving the same BspQI call. Also, if there are duplicate single-enzyme calls, SVMerge may produce additional duplicate merged calls.

13. Why are certain SV calls not merged by SVMerge, even though they appear to be overlapping?

SVMerge attempts to make a determination of whether two calls are “similar” enough to be merged, and there are SV type specific criteria. For example, for insertions and deletions, SVMerge requires that there be both overlap and size concordance between two calls. For inversions, the boundaries and the difference in the inversion ranges are required to be within a certain cutoff (20 kbp by default).
14. Does the SMAP-to-VCF converter support conversion of SVMerge output?

There is a separate SVMerge-to-VCF converter packaged with Bionano Solve; it can be run on the command-line. The SMAP-to-VCF converter expects that the input be in the standard SMAP format.

15. What is the expected runtime performance?

Below are representative runtime data for IrysSolve and Saphyr Compute running the haplotype-aware pipeline through to SV calling.

<table>
<thead>
<tr>
<th>SERVER CONFIGURATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enzyme</td>
</tr>
<tr>
<td>Sample 1</td>
</tr>
<tr>
<td>Sample 2</td>
</tr>
<tr>
<td>Sample 3</td>
</tr>
</tbody>
</table>

16. Why do I see fewer maps in regions where I’ve seen many more previously? Am I losing any information?

We have optimized parameters for allele separation. We expect that homozygous regions are now more likely to be represented by a single allele map. Heterozygous regions are expected to be represented by allele maps containing allele-specific SVs.

17. Could I take advantage of the new Solve release with my existing assemblies?

Yes, we recommend that SV detection be re-run on the command line using the new Solve tools in order to take advantage of improved SV calling sensitivity. After SV detection, one would replace the existing SV output directory with the new SV output directory, compress the entire assembly output, and import into Access for SV visualization and downstream analysis.

The computation of the inversion and translocation breakpoint confidence relies on assembly quality scores that are only available in assemblies generated with 2.0 or later releases of the Solve pipeline. If assembly quality scores were not available, “-1” would be output for inversion and translocation breakpoint confidence.

18. How do I take advantage of having both NLRS and DLS data?

Data generated with a second enzyme can provide a very valuable, independent validation of SVs of interest. Although Bionano has not developed and tested a particular workflow, one common workflow is to first discover relevant SVs using DLS and the Variant Annotation Pipeline, as well as, potentially, other data and tools. Then, generate a second dataset and de novo assembly for any samples that require validation. It should not be necessary to pass the data through the discovery pipeline. Simply query the SV list from the confirmation sample to find a match. Key considerations are that the SV coordinates may not match exactly because the coordinates are based on the enzyme recognition sites and have a median accuracy of ~3-5 kbp. A second key consideration is that because of coverage dropout from NLRS-produced “fragile sites,” confirmation could fail, i.e. be negative. That result should not be considered a refutation of the SV being tested. Users are recommended to use Bionano Access to inspect the region of interest to determine whether there is sufficient coverage and whether there is any SV “signal” that was not called but could partially confirm the SV (i.e. the map diverging from the reference at the SV location).
19. How close are the detected breakpoint coordinates to the actual coordinates?

The detected breakpoints are typically around 3 kbp from the actual breakpoints (with the 90th percentile at around 11 kbp).

20. How are molecule alignment confidence scores calculated?

Confidence scores are useful for evaluating alignments. We estimate the probability that the labels on a map match the labels on the reference purely by chance and that the maps are unrelated. The scores are calculated as -\log_{10} of the probability. Generally, alignments with higher scores are of higher confidence.

21. How do I interpret SV counts in the informatics reports?

The informatics report displayed in Access contains counts of clustered SV calls. Sometimes, the same SV may be present in more than one map. For example, a homozygous SV may be present in both allelic maps. We cluster similar SV calls in order to more accurately estimate the number of SVs in the assembly. In the full informatics report (part of the compressed assembly output), SV counts for both pre- and post-clustering are available.

22. How do I run the SMAP-to-VCF converter?

The SMAP-to-VCF converter is now packaged with Bionano Solve. It can be run on the command-line, and it can also be run within Bionano Access. The SVMerge-to-VCF converter is also available with Solve; currently, we only support command-line usage of the SVMerge-to-VCF converter.

23. Does the SMAP-to-VCF converter output masked calls?

Currently, the SMAP-to-VCF converter only outputs non-masked calls.
Appendix

Method for assessing SV calling performance

SV calling performance with simulated data

We simulated random SV events so that we could estimate our genome-wide SV calling performance accurately. The human reference assembly hg19 was used as an “SV-free” base genome (in our performance analyses, SVs were called against hg19).

Simulation of insertions and deletions

We randomly introduced 1600 insertions and 1600 deletions into an in-silico map of hg19. The insertions involved addition of new simulated material (random sequences of defined sizes) and deletions involved removing material; we did not simulate replacement of sequence (substitutions). The simulated events were at least 500 kbp from each other or N-base gaps. They ranged from 200 bp to 1 Mbp, with smaller SVs more frequent than larger ones.

Based on the edited hg19, molecules were simulated to resemble actual molecules collected on a Bionano system. This entailed adding sizing error in accordance with the model used in RefAligner, outliers created by stitching and DNA knots and folds, and fragile sites. In addition, we simulated molecules from the unedited hg19 and mixed with molecules from the edited hg19 such that all events would be heterozygous. Four such datasets with effective coverage levels from 87X-110X were generated. The simulated molecules were used as input for the haplotype-aware assembly pipeline, and SV calls were compared to the ground truth.

Simulation of translocations

Random translocation events were simulated to form an edited genome (Figure 6). 918 segments were randomly selected across hg19 and randomly inserted elsewhere into the genome. We avoided N-base gaps, and the size of the translocation fragments ranged from 50 kbp to 1 Mbp. Breakpoints were at least 500 kbp away from each other. For simulated intrachromosomal translocations, the breakpoints were at least 5 Mbp away. Each translocated fragment is expected to generate two translocation breakpoints. These can also be considered transpositions.

Figure 6. Simulation of translocated fragments. Random fragments are removed from a donor chromosome and inserted into an acceptor chromosome.
Simulation of inversions

About 900 intervals of 5 kbp to 1 Mbp in size were randomly sampled across hg19 and inverted to create an edited genome with simulated inversions. The inversion events were at least 500 kbp away from each other and were required to not overlap with N-base gaps.

Simulation of duplications

About 900 intervals of 5 kbp to 1 Mbp in size were randomly sampled across hg19. For each sampled interval, an extra copy of the sequence was inserted in tandem next to the original segment. The new copy could either be in the same or opposite orientation such that performance for detecting tandem and inverted duplications could be assessed. The duplication events were at least 500 kbp away from each other and were required to not overlap with N-base gaps.

Simulation of molecules

Molecules were simulated from edited genomes according to empirically derived error and size characteristics. Error-free molecules were simulated; then, errors (such as sizing errors and FP and FN labels) were added. Simulated molecules were used as input for de novo assembly and SV detection.

SV calling performance with real data

CHM1/13 analysis

Data was generated from homozygous CHM1 and CHM13 cell lines initially derived from hydatidiform moles. Single-molecule maps from CHM1 and CHM13 were evenly sampled and combined in silico to simulate a diploid genome at 80X effective coverage. We also downsampled the dataset to different coverage levels (50X and 70X). The CHM1, CHM13, and the CHM1/13 mixture molecule sets were assembled separately.

A three-way SV comparison analysis was performed for each trio set of CHM1, CHM13, and CHM1/13 mixture SV calls. The SV calls from the CHM1 and CHM13 pure assemblies were considered as the (conditional) ground truth. The sensitivity and positive predicted value (PPV) at different coverage levels were analyzed. Sensitivity was defined as fraction of SV calls in pure assemblies that were called in the mixture assembly, and PPV was defined as fraction of calls in mixture assembly that were called in the pure assemblies. We reported SV calling performance based on the 70X dataset (Table1).

CEPH trio analysis

Using genome mapping, Mak et al. analyzed a Caucasian trio from the 1000 Genomes Project (the parents NA12891 and NA12892, and the daughter NA12878) and published an expert-curated SV list. We reanalyzed the same starting data and compared the resulting SV list from the automated pipeline against the curated list.

Method for assessing CNV calling performance

CNV calling performance with simulated data

The simulations are used for performance validation and for constructing confident tables. The methods are described in Introduction to Copy Number Analysis (PN#30210). Briefly, we randomly introduced copy number events across the genome. Starting from a molecule-to-reference alignment, molecules that overlap with simulated copy number events are sampled accordingly based on their simulated copy number states. The sampled molecules were used as input to the copy number analysis pipeline, and the detected events were compared with the simulated ground truth to derive sensitivity and PPV data.
CNV calling performance with real data

Samples with known copy number events were analyzed with BspQI, BssSI, and/or DLE-1. The datasets are analyzed using the Solve pipeline, and the copy number analysis output were assessed based on coordinates where CNV events were expected.

Mask generation

Compiling common translocation breakpoint list

Genome map assemblies for control human samples were used as input. Given that they were phenotypically normal samples, we assumed that the control samples did not contain translocations. We performed SV detection on each assembly, and detected translocation breakpoints were compiled.

Common translocation breakpoint calls were merged and included into the masks. Two neighboring breakpoints were merged if they were within 25 kbp of each other. After merging, for each breakpoint or merged region, a 10-kbp buffer was added on both sides. These common breakpoint regions were labeled as “common” in the masks.

Compiling annotated segmental duplication regions

Annotated segmental duplication regions of at least 50 kbp were compiled from the UCSC Genome Browser database. Sequences from the annotated segmental duplication regions were extracted and converted into in silico maps. For each pair of segmental duplication regions, we checked for map-based similarity. If the in silico maps aligned with each other with a p-value of less than 1E-4, the segmental duplication region would be included in the mask and labeled as “segdupe”.
## Technical Assistance

For technical assistance, contact Bionano Genomics Technical Support. You can retrieve documentation on Bionano products, SDS’s, certificates of analysis, frequently asked questions, and other related documents from the Support website or by request through e-mail and telephone.

<table>
<thead>
<tr>
<th>Type</th>
<th>Contact</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email</td>
<td><a href="mailto:support@bionanogenomics.com">support@bionanogenomics.com</a></td>
</tr>
<tr>
<td>Phone</td>
<td><strong>Hours of Operation:</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Monday through Friday, 9:00 a.m. to 5:00 p.m., PST</strong></td>
</tr>
<tr>
<td></td>
<td><strong>US: +1 (858) 888-7663</strong></td>
</tr>
<tr>
<td>Website</td>
<td><a href="http://www.bionanogenomics.com/support">www.bionanogenomics.com/support</a></td>
</tr>
</tbody>
</table>