SUPPLEMENTARY

1 Identifying transpositions

Both the HX1 and the GIAB projects provide a list of insertions, detected using long-read approaches. Each insertion has an insertion site and an insertion sequence. We identified those insertions that are caused by a transposition event using the following algorithm:

1. We used TRF (2) on the insertion sequences to identify their period. Given an insertion sequence S, its period is a sequence P such that S can be obtained by concatenating P n times (where $n \geq 1$). For example, T is the period of the sequence TTTTTTT, AG is the period of AGAGAGAG and AGCTCGA is the period of AGCTCGA.

2. For a given insertion, let l be the length of its period P. We extracted from the reference a genomic region surrounding the insertion site; in particular, let i be the genomic coordinate of the insertion site, we extracted the region between $i-2l$ and $i+2l$. We call the extracted sequence R.

3. We computed the local alignment between P and R. The scoring scheme assigns 1 point to matches, -2 to mismatches, -3 to gap open and -1 to gap extension. If the alignment score is at least 90% of the length of P, we classified the insertion as a tandem duplication.

4. To classify the remaining insertions into transpositions and novel insertions, we mapped their original insertion sequences to the reference using Blast 2.5.0 (1), default parameters and soft masking disabled. We required the hits to cover at least 90% of the query. When no hits were found, we classified the insertion as novel, otherwise we classified it as a transposition.

2 Post-processing of results

HX1 does not report SVs on chrY, although the individual is male; on the other hand, GIAB does not report alternative chromosomes. For consistency, we only compare on chr1-22, and we remove predictions on all the other contigs. Furthermore, for all tools we filter centromeric regions as they are notoriously difficult and leading to false positives.

3 Comparing predicted transpositions to the benchmark

Since not all the methods report the source of the transposition, we have two different criteria for comparing a benchmark insertion and a predicted transposition. Let μ and σ be the mean and the standard deviation of the insert size of all the read pairs in the dataset we are considering. Let d be $\mu+3\sigma$. We say that two coordinates are close if they are in the same chromosome and within d bps from each other.

Our first criteria, applicable to all methods, is the following. When only the insertion site is predicted, it must be within $maxFS$ of to the actual insertion site. When the two ends of the transposition event are given, one of them must within $maxFS$ of the insertion site.

Our second criteria, not applicable to DD_DETECTION and the tested database-based methods, requires both ends to be reported. As previously, one end within $maxFS$ of the insertion site. Furthermore, we locally align the insertion sequence S reported in the benchmark to the $2 \cdot maxFS$ bp-long region centred at the other end of the event. We assign a score of 2 to matches, -2 to mismatches, -4 to gap open and -1 to gap extension. We accept the transposition if the alignment score is greater than the length of S.

We consider a benchmark transposition as correctly predicted if it is matched by any called insertion, whether a tandem duplication, transposition or novel insertion. Note that this is only relevant to the general SV callers (Delly, Lumpy, Socrates), since all the other tools specialise in predicting transpositions. We consider a call a true positive when it matches an insertion in the benchmark.

Table 1. For HX1 and HG001, we report the percentage of discordant pairs supporting a benchmark transposition where the stable read was correctly identified.

Strategy	HX1	HG001
Highest MQ	97.6%	93.7%
Discordant count	84.5%	71.4%
Different Chromosome Count	82.2%	69.9%

4 Determining the stable end

A crucial part of the algorithm is determining the stable end given a discordant pair of reads. We tested three simple strategies:

Highest MQ: selects the read with the highest map quality as the stable end;

Discordant count: given a read r, counts the number of discordant pairs having a read within a distance of $maxFS$ of r. In the pair, it selects as stable the mate with the highest such number;

Different Chromosome Count: given a read r, extracts the set D of discordant pairs having a read within a distance of $maxFS$ of r, and counts the number of unique different chromosomes the mates in D are mapped to. In the pair, it selects as stable the mate with the highest such number.

In the case of a tie, the algorithm duplicates the pair and both reads are selected as stable and unstable once. For this reason, we consider the pair as correctly classified.

Table 1 reports the percentages of success of the methods. Highest MQ is not only the lighter strategy computationally and the easiest to implement, but it is also the best performing strategy.

5 Tested programs

All programs were tested on a cluster running CentOS 6.5. The C++ compiler was g++ 4.9.3. For Mobster and Socrates, Java 8 was used. In the commands, two recurring variables will be $reference , which is the location of the reference genome, and $bam, which is the location of the input BAM file. The other variables will be named with self-explanatory names.

DD_DETECTION's source code was cloned from the repository at https://bitbucket.org/mkroon/dd_detection/overview. The command used was:

```
$ dd_detection -f $reference -i config.txt
```

```
  -c ALL -o $output_dir
```

config.txt contains the location of the BAM file, the mean insert size of the dataset and a sample name.

Delly's source code was cloned from the repository at https://github.com/dellytools/delly. The command used was:

```
  $ delly call -t TRA -o $output_file
  -g $reference $bam
```

Lumpy's source code was cloned from the repository at https://github.com/arq5x/lumpy-sv. The command used was:

```
  $ lumpyexpress -B $bam -o $output_file
```

MELT was downloaded from its official page http://melt.igs.umaryland.edu. The archive contains a runnable jar and databases for humans (hg19 and hg38) as well as for chimpanzee. The command used was:

```
java -jar MELT.jar Single -bamfile $bam
-c 60 -h $reference -n $provided_bed_file
-w $workdir -t transposon_file_list
```

Mobster's source code was cloned from https://github.com/jyhehir/mobster, and was installed using the provided script. The parameters provided in the default Mobster.properties file were used. Mobster comes with two databases, one for hg19 and one for hg38. The appropriate one was used in all datasets. The command used was:

```
  $ java -jar MobileInsertions.jar
  -properties Mobster.properties
  -in $bam -sn $sample_name -out $output_prefix
```

Retroseq was downloaded from https://github.com/tk2/RetroSeq. It works in two consecutive steps, and commands used were:

```
./retroseq.pl -discover -bam $bam
-output $workdir/discover.txt
 -refTEs $refTE-file

./retroseq.pl -call -bam $bam
-input $workdir/discover.txt
-ref  $reference
-output $workdir/calls.vcf -soft
```

Since Retroseq does not come with a database of transposable elements, we built one from Mobster's database. We divided the entries into ALU, L1 and Others.

Socrates was downloaded as a jar, version 1.13 (latest currently available). The command used was:

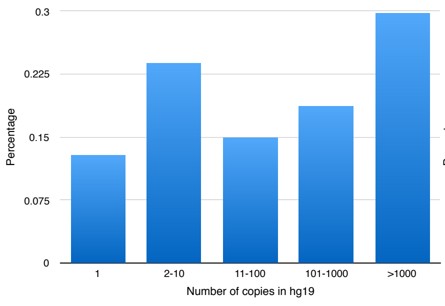
```
  $ java -jar socrates.jar $bowtie2_idx $bam
```

TranSurVeyor, the command used was:

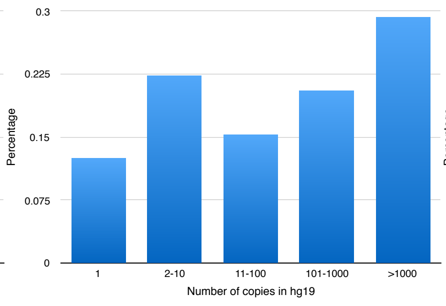
```
  $ python surveyor.py $bam $workdir $reference
  --threads 8 --samtools /path/to/samtools
  --bwa /path/to/bwa
```

REFERENCES

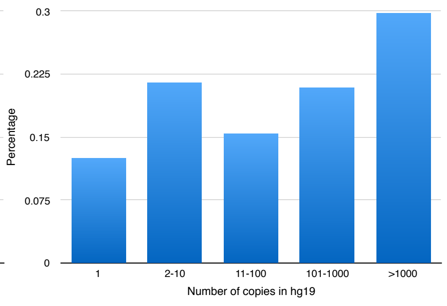
1. S F Altschul, W Gish, W Miller, E W Myers, and D J Lipman. Basic local alignment search tool. Journal of molecular biology, 215:403–410, October 1990.
2. G Benson. Tandem repeats finder: a program to analyze dna sequences. Nucleic acids research, 27:573–580, January 1999.



(a) HG002



(b) HG003



(c) HG004

Figure S1. Transpositions in the benchmark of (a) HG002, (b) HG003 and (c) HG004 were partitioned into categories according to how many times the transposed (i.e. inserted) sequence appears in the reference. Most of them appear multiple times.