Supplementary Information for:

Benchmarking Computational Tools for Polymorphic Transposable Element Detection

Lavanya Rishishwar[1,2,3,4], Leonardo Mariño-Ramírez[3,5,*] and I. King Jordan[1,2,3,4,*]

[1]School of Biology, Georgia Institute of Technology, Atlanta, GA, USA
[2]Applied Bioinformatics Laboratory, Atlanta, GA, USA
[3]PanAmerican Bioinformatics Institute, Cali, Valle del Cauca, Colombia
[4]BIOS Centro de Bioinformática y Biología Computacional, Manizales, Caldas, Colombia
[5]National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, Bethesda, MD, USA

*Corresponding Authors:
Leonardo Mariño-Ramírez
8600 Rockville Pike MSC 6075
Bethesda, MD 20894-6075
301-402-3708
marino@ncbi.nlm.nih.gov

I. King Jordan
950 Atlantic Drive
Atlanta, GA 30332-2000
404-385-2224
king.jordan@biology.gatech.edu

LR: lavanya.rishishwar@gatech.edu
LMR: marino@ncbi.nlm.nih.gov
IKJ: king.jordan@biology.gatech.edu

Notes on the general structure of the commands

Reference sequence files used in the following commands:
a) `hg19.fa` is the hg19 genome FASTA file downloaded from UCSC genome browser [1],
b) `hs37d5.fa` is the hg19 genome FASTA file with the decoy chromosome downloaded from the 1000 genomes project FTP site [2, 3].

Commands used for generating simulated BAM files

A custom written Perl script was used to spike polyTEs (AluY, L1 and SVA) in the hg19 reference genome. Following the *in silico* polyTE insertions, the insertions were verified by BLASTing the inserted sequence against the spiked reference genome and offsetting the start position to account for the inserted polyTE. The final command used was:

```
# Create the BLAST database
makeblastdb -in hg19.polyTESpiked.fa -dbtype nucl

# Query each inserted TE sequence (insertedTE.fa) against the BLAST database
# followed by start position correction
blastn -query insertedTE.fa -db hg19.polyTESpiked.fa -max_target_seqs 1 -
outfmt "6 qseqid sseqid sstart qlen" -num_threads 9 -perc_identity 100 -
max_hsps 1 | sed 's/chr//' | sed 's/.*_//' | awk 'BEGIN{offset=0; lastChr =
0; OFS="\t"}{if(lastChr != $2){offset = 0; lastChr = $2}; $3 = $3 - offset;
print $2,$3-1,$3,$1; offset += $4}' > blastOut.txt
```

Once each polyTE insertion was successfully verified, sequence reads were simulated using the ART simulator (version 2.3.7) [4]. ART was run on the Illumina MiSeq profile with the read length of 150 bp, read coverage of 5-30X, fragment length of 500bp and a standard deviation of 10bp. The following is a command for generating the simulated reads on a 5x coverage:

```
art_illumina -sam -na -i hg19.polyTESpiked.fa -l 150 -p -f 5 -s 10 -ss MS -o
simulated5x -m 500
```

The output paired-end sequencing read files (simulated5x1.fq and simulated5x2.fq) were then mapped to the reference hg19 genome using bwa mem aligner [5] followed by sorting and indexing by SAMtools [6].

```
bwa mem -t 20 -I 500,10 hg19.fa simulated5x1.fq simulated5x2.fq >
simulated5x.sam
samtools sort -o simulated5x.bam -@ 10 -m 3G simulated5x.sam && samtools
index simulated5x.bam
```

Commands used for calling polyTE in different tools

The commands listed below are general in nature and were changed for some parameters shown inside `<angular brackets>`. For generality purposes, the input bam file is always called "`map.bam`" and output prefix is "`map`" in all of the commands listed below. Often the output and the messages produced on the error streams were redirected to a log and error file.

Paired-end sequencing reads extracted from `map.bam` in FASTQ format are r1.fq and r2.fq. For the simulated data set, the original FASTQ files were available and extraction was not required. For NA12878 data sets, FASTQ reads were extracted using PICARD tools' `SamToFastq` utility.

```
java -Xmx40G -jar ~/bin/picard.jar SamToFastq I=map.bam FASTQ=r1.fastq SECOND_END_FASTQ=r2.fastq INCLUDE_NON_PRIMARY_ALIGNMENTS=true
```

Some of the tools used in the study contain two modes, one for detecting polyTE insertions present in the sample but absent in the reference genome and the other for detecting polyTE insertions absent in the sample but present in the reference genome. All the tools were run to detect the former mode, *i.e.*, to detect polyTE insertions absent in the reference genome.

1. MELT (Version: 1.2.20)

Dependencies: Java

```
java -Xmx20G -jar MELT.jar Single -h hs37d5.fa -l ./map.bam -n hg19.genes.bed -t meltTransposonFileList.txt -w output -b hs37d5 -c <coverage>
```

`-Xmx20G` argument controls the maximum memory of 20GB that the program (Java running MELT.jar) can use.
Files used:

1) `hg19.genes.bed` is packaged in MELT's setup add_bed_files/1KGP_Hg19/
2) `meltTransposonFileList.txt` contains:
   `/full/path/to/LINE_MELT.zip`
   `/full/path/to/ALU_MELT.zip`
   `/full/path/to/SVA_MELT.zip`
   Each of the above listed zip files is packaged in MELT's setup in me_refs/1KGP_Hg19/

2. Mobster (Version: 0.1.7c)

Dependencies: Java, Picard tools (bundled with the setup) and MOSAIK

The current available version for Mobster is 0.1.6 and requires the BAM files to be constructed using MOSAIK aligner, which wasn't the case in the data set analyzed here.

```
java -Xmx300G -jar MobileInsertions-1.0-SNAPSHOT.jar -properties map.properties -in map.bam -sn map -out mobster > mobster.log 2> mobster.err
```

The content of the `map.properties` file was mostly the same as the default properties file packaged with the Mobster program. Only three parameters were changed from the default properties file: input file name, output file name and minimum read depth coverage to suit the optimum depth of each tested data set.

Communication with the developer: Djie Tjwan Thung (January 17 – March 10, 2016)

The current released stable version (0.1.6) does not work well for alignments generated using bwa mem. The developer generously provided us the unreleased version 0.1.7c which works well with all alignments. The developer also provided us with the optimum parameters that were used with the NA12878 high coverage data set (as specified in the properties file):

DISCORDANT_CLUSTER_MAX_DISTANCE=600
READS_PER_CLUSTER=1
MINIMUM_CLIP_LENGTH=35
MAXIMUM_CLIP_LENGTH=7
MINIMUM_AVG_QUALITY=20 # Different from default
READ_LENGTH=100 # Different from default

3. RetroSeq (Version: 1.41)

Dependencies: Perl, SAMtools, bedtools and Exonerate

The commands listed below were obtained from the RetroSeq's "1000 Genome CEU Trio Analysis" page.
Website: https://github.com/wtsi-svi/RetroSeq/wiki/1000-Genome-CEU-Trio-Analysis

```
# Discover
retroseq.pl -discover -bam map.bam -output map.bam.candidates.tab -refTEs
ref_types.tab -eref probes.tab -align > log.txt 2> err.txt
# Calling phase
retroseq.pl -call -bam map.bam -input map.candidates.tab -ref hs37d5.fa -
output map -filter ref_types.tab -reads <minimum read depth> -depth <maximum
read depth> >> log.txt 2>> err.txt
bedtools window -b AluY_AluS.bed -a map.PE.vcf -v -w 100 > map.Alu.vcf 2>>
err.txt
bedtools window -b L1HS.bed -a map.vcf -v -w 200 > map.L1.vcf 2>> err.txt
```

Minimum and maximum read depth were changed for each sample depending on the coverage of the data set. `AluY_AluS.bed` and `L1HS.bed` comes packed with the RetroSeq package.

4. TEMP (Version: 1.04)

Dependencies: Perl, SAMtools v0.1.19, BWA, bedtools, twoBitToFa (Kent Source) and BioPerl

The command and the parameters used were obtained from the example usage in the TEMP's manual.

```
TEMP_Insertion.sh -i map.bam -s ../scripts/ -r
HomoSapienRepbaseTEConcensus.fa -t hg19_rpmk.bed -m 3 -f 500 -c 8 -u >
log.txt 2> err.txt
```

The final output is in `map.insertion.refined.bp.summary`. The resulting file can be further filtered using the following commands:

```
awk 'BEGIN{OFS="\t"}{if($7 >= 5 && $8 >= 0.1){print}}'
NA12878.insertion.refined.bp.summary | cut -f1-3 | sort -k1,1 -k2,2 -V | uniq
> temp.tsv
```

The "`scripts`" folder is the address of the folder containing TEMP scripts.
The files `HomoSapienRepbaseTEConcensus.fa` and `hg19_rpmk.bed` are the RepBase [7] consensus sequence and RepeatMasker [8] annotation file that comes as part of the TEMP package.

Communication with the developer: Jiali Zhuang (November 20, 2105 – March 10, 2016)
One issue with TEMP is that it only works with SAMtools version 0.1.19 or earlier. The author also recommended using the `–u` option to avoid multiple reporting of the same TE and filtering insertions that are supported by less than 5 reads (column 7 of the output) or have an allele frequency of less than 10% (column 8 of the output). For the high coverage data, we decided to use an even more stringent cut-off of 20 minimum reads and 20% allele frequency.

5. Tangram (Version: 0.3.1)

Dependencies: g++ 4.2.0+, zlib, pthread lib

Tangram requires BAM files to be created in a "special" format. That is, the BAM files should contain the reference genome as well as the mobilome and the reads should be mapped to the mobilome before they are mapped to the reference genome. This can be done using MOSAIK. The easiest method of doing this is using the gkno pipeline, developed by the same lab. Installation of the pipeline can be done using:

```
./gkno build
```

Once the pipeline is installed, the alignment can be generated by the following set of commands:

```
# Add the human reference genome and the mobilome
./gkno add-resource human

# Build the mobilome + reference genome index
./gkno pipe build-moblist-reference -ps human

# Align the reference using MOSAIK
/usr/bin/time -v ./gkno mosaik -q tsd5x1.fq -q2 tsd5x2.fq -ps human > log.txt
2> err.txt

# Run Tangram
/usr/bin/time -v ~/data/gkno_launcher/gkno tangram-index -ps human -a
tangram.dat > index.log 2> index.err
```

This step did not finish for us and Tangram was also run separately. Tangram's detection pipeline has multiple steps, a few which we ran parallel. These commands were derived from the manual and the usage from each program. The complete pipeline was run on the default set of parameters. Additionally, the output chromosome wise VCF files were compressed (bgzip), indexed (tabix) and concatenated (VCFtools [9]) to produce the resulting genome wide polyTE callset.

```
tangram_index -ref hs37d5.fa -sp moblist_19Feb2010_sequence_length60.fa -out
tangramIndex
tangram_bam -i map.bam -r moblist_19Feb2010_sequence_length60.fa -o
tangram.bam
samtools sort -@ 10 -m 2G tangram.bam tangramSorted
echo tangramSorted.bam > list.txt
tangram_scan -in list.txt -dir tangramScan
seq 1 22 | xargs -I CHR -P 22 sh -c 'tangram_detect -lb
tangramScan/lib_table.dat -ht tangramScan/hist.dat -in list.txt -ref
tangramIndex -rg CHR > chrCHR.vcf'
seq 1 22 | xargs -I CHR -P 22 sh -c 'bgzip chrCHR.vcf; tabix -p vcf
chrCHR.vcf.gz'
vcf-concat chr*.vcf.gz > tangram.vcf
```

The file `moblist_19Feb2010_sequence_length60.fa` was included with the Tangram package.

Communication with the developer:

Jiantao Wu (March 7, 2016): The first author and developer of the program Jiantao Wu was contacted regarding the error message "ERROR: Cannot read the number of anchors from the library file". We were not able to get any response from the developer. The same set of commands works on the low coverage data but fails on all other data sets.

Alistair N Ward (June 4, 2016): Contacted Dr. Alistair Ward regarding the error with the gkno pipeline and Tangram. Dr. Ward informed us that Tangrams hasn't undergone any recent development and is probably not the best tool to use for polyTE detection. Recommended MELT for such task.

6. ITIS (Download date: 1st March 2015)

Dependencies: Perl, SAMtools v0.1.19, bwa v0.7.7

The ITIS script was run on the default set of parameters.

```
itis.pl -g hg19.fa -t ./te.fasta -l 500 -N sampleName -1 r1.fq -2 r2.fq –e Y
> log.txt 2> err.txt
```

The `te.fasta` file is the FASTA consensus sequence of set of TEs expected to be polymorphic in the genome, viz., AluY, L1 and SVA. These sequences were obtained from the RepBase database [7].

Communication with the developer: Chuan Jiang (February 29 – March 11, 2016)

We were having difficulties in obtaining any predictions on any data set – actual or simulated. The developer recommended adding the `–e Y` option to the command which masks all the homologous sequences in the genome. This enabled prediction of polyTEs from the genome sequence data.

7. T-lex2 (Version: 2.2.2)

Dependencies: Perl, RepeatMasker, MAQ, SHRIMP2, BLAT
Additional dependencies for TSD: Phrap, FastaGrep

The basic command using default parameter set was selected to run T-lex2. The input files required by T-lex2 are:

1) TE list (polyTE.id; AluY, L1 and SVA)
2) TE annotations (polyTE.coord; gene coordinates for the TEs derived from RepeatMasker)
3) Reference genome (hs37d5.fa)
4) Sequencing data directory (fqDir)

```
tlex-open-v2.2.2.pl -T polyTE.id -M polyTE.coord -G hs37d5.fa -R fqDir
```

The fqDir contained a subdirectory named after the data set being analyzed. The subdirectory contained r1.fq and r2.fq, the paired end sequence files for the respective data set.

The program took ~4 weeks to run on the low coverage human data set but did generate appropriate log messages indicating that the program was running. After ~4 weeks, the tool predicted >300,000 human polyTE insertions and was thus deemed unreliable for these particular data sets.

References

1. Speir ML, Zweig AS, Rosenbloom KR et al. The UCSC Genome Browser database: 2016 update, Nucleic Acids Res 2016;44:D717-725.
2. Genomes Project C, Auton A, Brooks LD et al. A global reference for human genetic variation, Nature 2015;526:68-74.
3. Sudmant PH, Rausch T, Gardner EJ et al. An integrated map of structural variation in 2,504 human genomes, Nature 2015;526:75-81.
4. Huang W, Li L, Myers JR et al. ART: a next-generation sequencing read simulator, Bioinformatics 2012;28:593-594.
5. Li H, Durbin R. Fast and accurate short read alignment with Burrows-Wheeler transform, Bioinformatics 2009;25:1754-1760.
6. Li H, Handsaker B, Wysoker A et al. The Sequence Alignment/Map format and SAMtools, Bioinformatics 2009;25:2078-2079.
7. Bao W, Kojima KK, Kohany O. Repbase Update, a database of repetitive elements in eukaryotic genomes, Mob DNA 2015;6:11.
8. Smit AFA, Hubley R, Green P. RepeatMasker Open-4.0. http://www.repeatmasker.org.
9. Danecek P, Auton A, Abecasis G et al. The variant call format and VCFtools, Bioinformatics 2011;27:2156-2158.

Supplementary Table 1. **Rationale for the selection or omission of computational polyTE detection tools for this benchmark study and their relevance to human next-generation sequencing (NGS) data**. Extensive benchmarking was done on seven tools that were selected based on the criteria adopted in this study (see "Polymorphic TE detection tools" section). Additionally, four more previously not included polyTE detection tools were tested on the low coverage dataset. Other existing polyTE detection tools that were omitted from the benchmark are also listed along with the rationale of their omission. Briefly, tools that are not specialized for polyTE detection or requires specific TSDs were not included in the benchmark.

PolyTE detection tools selected for benchmarking			
Tool name	Rationale for selection	Tool's success	Relevance to human NGS data
MELT	All criteria	Success	High
Mobster	All criteria	Success	High
RetroSeq	All criteria	Success	High
Tangram	All criteria	Failure	High
TEMP	All criteria	Success	High
ITIS	Criterion #1 and #4	Success	Medium
T-lex/T-lex2	Criterion #1 and #4	Aborted	Medium
DD_DETECTION	Expanded set	Failure	High
Jitterbug	Expanded set	Failure	High
TE-Locate	Expanded set	Failure	Medium
TE-Tracker	Expanded set	Failure	Medium
PolyTE detection tools omitted from the benchmarking			
Tool name	Rationale for omission		Relevance to human NGS data
GRIPper	Detects non-reference gene copy insertion		High
TIGRA	Breakpoint assembler, not an SV caller		High
TranspoSeq	Requires paired tumor/normal WGS data		High
Tea	Requires paired tumor/normal WGS data		High
TraFiC	Requires paired tumor/normal WGS data		High
VariationHunter	General purpose SV detection tool		High
HYDRA-SV	General purpose SV detection tool		High
MetaSV	General purpose SV detection tool		High
ngs_te_mapper	Requires TSDs to be provided		Medium
RelocaTE	Requires TSDs to be provided		Low

Supplementary Table 2. **Summary of algorithmic differences between the computational polyTE detection tools benchmarked in this study**. More detailed differences are listed in Supp. Table 3.

	Read mapping				Breakpoint estimation			Filtering criteria				Output features		
Tool	All DP reads	Treats SR & DP independently	SR searched after DP	Mobilome Aligner	Fragment size distribution	SR dependent	Holistic	Read depth - flanking	Read depth - site	Known TEs	Mapping quality	VCF file	Predicts TSD	Predicts zygosity
MELT	✓	×	✓	Bowtie2	?	×	✓	✓	✓	✓	?	✓	✓	✓
Mobster	✓	✓	×	MOSAIK	✓	×	✓	×	✓	✓	×	×	×	×
RetroSeq	✓	×	✓	Exonerate	×	×	✓	✓	✓	✓	×	✓	×	✓
Tangram	✓	✓	×	MOSAIK	✓	×	✓	×	✓	✓	×	✓	×	×
TEMP	✓	×	✓	BWA	✓	✓	×	×	✓	×	×	×	×	NA
ITIS	×	✓	×	BWA	×	✓	×	✓	✓	✓	✓	×	×	×

Supplementary Table 3. **Detailed Algorithmic differences between the computational polyTE detection tools benchmarked in this study**.

Tool	DP definition	SR definition	DP and SR search paradigm	Mobilome alignment tool	Cluster definition	Merging clusters	Breakpoint estimation	Filtering criteria
MELT	Information not available	Information not available	DPs were used to identify potential/ candidate TE sites SRs were used to identify breakpoints and TSDs	Bowtie2 with default parameters	Sites with at least 4 DP anchors clustered within 500bp of each other	Merges all DP and SR clusters from all BAM files (from 1KG project)	Unspecified type of model was built containing all available information for the candidate site. This model was then used to predict precise insertion site, strand, TSD, insertion sequence and length.	Based on: 1) minimum 4 supporting DPs 2) proximity to a reference TE 3) filter sites with depth of coverage outside 70-130% of the 100bp flanking region
Mobster	1) Orientation different from expectation or 2) Distance between pairs significantly different or 3) Reads mapping to different chromosome or 4) One read mapped, other not DP will have at least one uniquely mapping read referred to as the anchor read	Reads that map partially (clipped); will have one uniquely mapping anchor read and uniquely mapping unclipped part (anchor for SR)	DP and SR are searched independently Anchor reads tagged as unmapped or by the TE family their mate/clipping maps to	MOSAIK (hash size = 9; max mismatches = 10%, min length = 20 bp)	<u>DP clusters</u> 1) Anchors map to same strand 2) support the same TE family 3) have start position in proximity to each other <u>SR clusters</u> 1) Anchors belong to the same TE family or polyA/T stretch 2) same side clipping 3) clipped within a few bp of each other	Merge same family (or homopolymer) forward and reverse strand clusters First merge DP and SR independently, then proceed to merge the two Confidence assigned based on the number of clusters and orientations (5' and 3') that were merged	Breakpoints are estimated based on the inner borders of 5' and 3'clipped If clipped reads not available, inner borders of DP clusters are used for breakpoint estimation Else, estimated from insert size distribution and cluster length	Based on: 1) proximity (within 90bp) to a reference TE 2) user controlled read depth based filtering
RetroSeq	1) SAM flag 0x0002 unset, i.e., reads that are not proper pairs, or 2) One mate of the pair is unmapped Proper pairs are defined as reads whose pair maps within the expected distance	Partially mapped reads	Extracts DP in the beginning SR are only searched for breakpoint estimation step	Exonerate (80% min identity, 36 bp min length, mapping quality 30, local alignment with affine gap penalty, report best 5 results)	Forward and reverse orientation clusters created by the start position of the anchor reads Max gap 120bp between reads in a cluster	Uses bedtools window command to merge forward and reverse clusters	Excludes clusters with average read depth surrounding the cluster above a cutoff (def 200). Estimates using a set of parameters: 1) read depth of DP on both strands, 2) forward to reverse reads ratio at 5' and 3' of the putative breakpoint and 3) distance between last 5' and first 3' read.	Based on: 1) proximity (within 100bp of an Alu or within 200bp of an L1) to a reference TE Confidence for each genotype provided in the output VCF file

Supplementary Table 3. (Continued)

Tool	DP definition	SR definition	DP and SR search paradigm	Mobilome alignment tool	Cluster definition	Merging clusters	Breakpoint estimation	Filtering criteria
Tangram	Utilizes customized BAM format that contains both the genome and TE reference sequence alignment (No instruction provided on generating this alignment) DP are read pairs with one read mapping uniquely to the reference genome and the other mapping to the TE reference sequence	SR have one mate mapping uniquely while the other is either soft-clipped or unmapped The unaligned or soft-clipped reads are then realigned to both reference genome and reference TE sequences	DP and SR are searched independently	MOSAIK (Done before the process begins)	Clusters candidate read pairs using fragment center position; applies a customized nearest-neighbor algorithm for clustering Utilizes fragment length distribution Capable of handling multiple different libraries		DP – identifies pair of clusters spanning on the insertion from 3' and 5'. Leftmost position candidate insertion position SR – Performs fast local alignment to identify the breakpoint	Based on 1) supporting reads per insertion, 2) additional filtering if only DP support and 3) proximity to a reference TE
TEMP	One uniquely mapping read (anchor read) and the other read that maps to multiple distant locations or is unmappable	Reads that start in genomic sequence but are interrupted by transposon or non-contiguous genomic sequence Clipped portion which maps to the TE should be at least 7bp long	SR are looked for after DP DP identifies insertion regions, SR helps in breakpoint estimation	BWA-aln and BWA-sampe	Defines intervals such as they contain TE "junction in the beginning (and) at the end of the anchor read, and extending into the genome by the length of the average insert size" Reads supporting same TE type, same orientation and intervals that overlap by at least 1nt are clustered		Extends intervals in both directions to find overlapping SRs. Estimates breakpoint based on the clipped portion. In case of multiple locations, selects the one with highest support. When base estimate is not available, interval midpoint is taken as insertion position	Based on 1) Read depth
ITIS	One end mapped to the reference genome, other mapped to the TE	At least one end covers both reference and TE	DP and SR are searched independently but SR determines the genomic location	BWA	DP and SR that are in close proximity		Based on the SR	Based on 1) MAQ > 0 2) 2 < RD < 300 – around insertion 3) RD (DP/SR) > 2