TEBreak: Generalised insertion detection

Adam D. Ewing (adam.ewing@mater.uq.edu.au)

April 10, 2017

1 Introduction

1.1 Software Dependencies and Installation

TEBreak requires the following software packages be available:

1. samtools (`http://samtools.sourceforge.net/`)
2. bwa (`http://bio-bwa.sourceforge.net/`)
3. LAST (`http://last.cbrc.jp/`)
4. minia (`http://minia.genouest.org/`)
5. exonerate (`https://github.com/adamewing/exonerate.git`)

Please run the included setup.py to check that external dependencies are installed properly and to install the required python libraries:

```
python setup.py build
python setup.py install
```

2 Testing / example run

2.1 Run tebreak.py

TEBreak includes a small example intended to test whether the software is installed properly. Starting in the TEBreak root directory, try the following:

```
tebreak/tebreak.py \
-b test/data/example.ins.bam \
-r test/data/Homo_sapiens_chr4_50000000-60000000_assembly19.fasta \
-i test/data/example.region.bed \
--pickle test/example.pickle \
--detail_out test/example.tebreak.detail.out
```

This will yield two files, in this case test/example.pickle and test/example.tebreak.detail.out. If not specified, the output filenames will be based on the input BAM file name (or the first BAM in a comma delimited list of BAMs). The "pickle" file contains information that will be passed on to subsequent steps in TEBreak, the "detail" output contains raw putative insertion sites which generally require further refinement, annotation, and filtering, but can be useful in troubleshooting when a true positive site is missed.

2.2 Run resolve.py

The next step is to resolve putative insertions, in this case human transposable element insertions, using resolve.py:

```
tebreak/resolve.py \
-p test/example.pickle \
-i lib/teref.human.fa \
--detail_out test/example.resolve.detail.out \
-o test/example.table.txt
```

If all went well, test/example.table.txt should contain evidence for 5 transposable element insertions (2 L1s and 3 Alus). If it does not, please double check that all prerequisites are installed, try again, and e-mail me at adam.ewing@mater.uq.edu.au with error messages if you are still not having any luck.

3 Insertion site discovery (tebreak.py)

3.1 Usage

The following output can be obtained by running tebreak.py -h:

```
usage: tebreak.py [-h] -b BAM -r BWAREF [-p PROCESSES] [-c CHUNKS]
                  [-i INTERVAL_BED] [-d DISCO_TARGET] [--minMWP MINMWP]
                  [--min_minclip MIN_MINCLIP] [--min_maxclip MIN_MAXCLIP]
                  [--min_sr_per_break MIN_SR_PER_BREAK]
                  [--min_consensus_score MIN_CONSENSUS_SCORE] [-m MASK]
                  [--rpkm_bam RPKM_BAM] [--max_fold_rpkm MAX_FOLD_RPKM]
                  [--max_ins_reads MAX_INS_READS]
                  [--min_split_reads MIN_SPLIT_READS]
                  [--min_prox_mapq MIN_PROX_MAPQ]
                  [--max_N_consensus MAX_N_CONSENSUS]
                  [--exclude_bam EXCLUDE_BAM]
                  [--exclude_readgroup EXCLUDE_READGROUP]
                  [--max_bam_count MAX_BAM_COUNT] [--map_tabix MAP_TABIX]
                  [--min_mappability MIN_MAPPABILITY]
                  [--max_disc_fetch MAX_DISC_FETCH]
                  [--min_disc_reads MIN_DISC_READS] [--bigcluster BIGCLUSTER]
                  [--skipbig] [--tmpdir TMPDIR] [--pickle PICKLE]
                  [--detail_out DETAIL_OUT] [--debug]

Find inserted sequences vs. reference

optional arguments:
  -h, --help            show this help message and exit
  -b BAM, --bam BAM     target BAM(s): can be comma-delimited list or .txt
                        file with bam locations
  -r BWAREF, --bwaref BWAREF
                        bwa/samtools indexed reference genome
```

```
-p PROCESSES, --processes PROCESSES
                      split work across multiple processes
-c CHUNKS, --chunks CHUNKS
                      split genome into chunks (default = # processes),
                      helps control memory usage
-i INTERVAL_BED, --interval_bed INTERVAL_BED
                      BED file with intervals to scan
-d DISCO_TARGET, --disco_target DISCO_TARGET
                      limit breakpoint search to discordant mate-linked
                      targets (e.g. generated with
                      /lib/make_discref_hg19.sh)
--minMWP MINMWP       minimum Mann-Whitney P-value for split qualities
                      (default = 0.01)
--min_minclip MIN_MINCLIP
                      min. shortest clipped bases per cluster (default = 3)
--min_maxclip MIN_MAXCLIP
                      min. longest clipped bases per cluster (default = 10)
--min_sr_per_break MIN_SR_PER_BREAK
                      minimum split reads per breakend (default = 1)
--min_consensus_score MIN_CONSENSUS_SCORE
                      quality of consensus alignment (default = 0.9)
-m MASK, --mask MASK  BED file of masked regions
--rpkm_bam RPKM_BAM   use alternate BAM(s) for RPKM calculation: use
                      original BAMs if using reduced BAM(s) for -b/--bam
--max_fold_rpkm MAX_FOLD_RPKM
                      ignore insertions supported by rpkm*max_fold_rpkm
                      reads (default = None (no filter))
--max_ins_reads MAX_INS_READS
                      maximum number of reads to use per insertion call
                      (default = 100000)
--min_split_reads MIN_SPLIT_READS
                      minimum total split reads per insertion call (default
                      = 4)
--min_prox_mapq MIN_PROX_MAPQ
                      minimum map quality for proximal subread (default =
                      10)
--max_N_consensus MAX_N_CONSENSUS
                      exclude breakend seqs with > this number of N bases
                      (default = 4)
--exclude_bam EXCLUDE_BAM
                      may be comma delimited
--exclude_readgroup EXCLUDE_READGROUP
                      may be comma delimited
--max_bam_count MAX_BAM_COUNT
                      maximum number of bams supporting per insertion
--map_tabix MAP_TABIX
                      tabix-indexed BED of mappability scores
--min_mappability MIN_MAPPABILITY
```

```
                      minimum mappability (default = 0.1; only matters with
                      --map_tabix)
--max_disc_fetch MAX_DISC_FETCH
                      maximum number of discordant reads to fetch per
                      insertion site per BAM (default = 50)
--min_disc_reads MIN_DISC_READS
                      if using -d/--disco_target, minimum number of
                      discordant reads to trigger a call (default = 4)
--bigcluster BIGCLUSTER
                      set big cluster warning threshold (default = 1000)
--skipbig             drop clusters over size set by --bigcluster
--tmpdir TMPDIR       temporary directory (default = /tmp)
--pickle PICKLE       pickle output name
--detail_out DETAIL_OUT
                      file to write detailed output
--debug
```

3.2 Description

Insertion sites are discovered through clustering and scaffolding of clipped reads. Additional support is obtained through local assembly of discordant read pairs, if applicable. Input requirements are minimal, consisting of one of more indexed BAM files and the reference genome corresponding to the alignments in the BAM files(s). Many additional options are available and recommended to improve performance and/or sensitivity.

3.3 Input

BAM Alignment input (-b/–bam) BAMs ideally should adhere to SAM specification i.e. they should validate via picardś ValidateSamFile. BAMs should be sorted in coordinate order and indexed. BAMs may consist of either paired-end reads, fragment (single end) reads, or both. Multiple BAM files can be input in a comma-delimited list.

Reference genome (-r/–bwaref) The reference genome should be the **same as that used to create the target BAM file**, specifically the chromosome names and lengths in the reference FASTA must be the same as in the BAM header. The reference must be indexed for bwa (`bwa index`) and indexed with samtools (`samtools faidx`).

Pickled output (–pickle) Output data in python's pickle format, meant for input to other scripts including resolve.py and picklescreen.py (in /scripts). Default is the basename of the input BAM with a .pickle extension.

Detailed human-readable output (–detail_out) This is a file containing detailed information about consensus reads, aligned segments, and statistics for each putative insertion site detected. Note that this is done with minimal filtering, so these should not be used blindly. Default filename is tebreak.out

Options important for optimal runtime

- -p/–processes : Split work across multiple processes. Parallelism is accomplished through python's multiprocessing module. If specific regions are input via -i/–interval_bed, these intervals will be distributed one per process. If a whole genome is to be analysed (no -i/–interval_bed), the genome is split into chunks, one per process, unless a specific number of chunks is specified via -c/–chunks.

- -i/–interval_bed : BED file specifying intervals to be scanned for insertion evidence, the first three columns must be chromosome, start, end. A number of intervals equal to the value specified by -p/–processes will be searched in parallel. Overrides -c/–chunks.

- -c/–chunks : if no input BED is specified via -i/–interval_bed and no set of discordant targets is specified by -d/–disco_target, split the genome up into some number of "chunks" (default = number of processes). For whole-genome sequencing data split across 32-64 cores, 30000-40000 chunks often yields reasonable runtimes.

- -m/–mask : BED file of regions to mask. Reads will not be considered if they fall into regions in this file. Usually this file would include regions likely to have poor unique alignability and a tendency to generate large pileups that leads to slow parsing such as centromeres, telomeres, recent repeats, and extended homopolymers and simple sequence repeats. An example is included for hg19/GRCh37 (lib/hg19.centromere_telomere.bed).

- -d/–disco_target : if paired reads are present and the sequencing scheme permits, you can use discordant read pair mapping to narrow down regions to search for split reads as evidence for insertion breakpoints. Most Illumina WGS/WES sequencing approaches support this. This can result in a dramatic speedup with the tradeoff of a slightly reduced sensitivity in some cases.

Additional options

- –minMWP: Minimum value of Mann-Whitney P-value used to check similarity between the distribution of mapped base qualities versus clipped base qualities for a soft-clipped read (default = 0.01).

- –min_minclip : the shortest amount of soft clipping that will be considered (default = 3, minimum = 2).

- –max_minclip : For a given cluster of clipped reads, the greatest number of bases clipped form any read in the cluster must be at least this amount (default = 10).

- –min_sr_per_break : minimum number of split (clipped) reads required to form a cluster (default = 1)

- –min_consensus_score : minimum quality score for the scaffold created from clipped reads (default = 0.95)

- -m/–mask : BED file of regions to mask. Reads will not be considered if they fall into regions in this file.

- –rpkm_bam : use alternate BAM file(s) for RPKM calculation for avoiding over-aligned regions (useful for subsetted BAMs).

- –max_fold_rpkm : reject cluster if RPKM for clustered region is greater than the mean RPKM by this factor (default = None (no filter))
- –max_ins_reads : maximum number of reads per insertion call (default = 100000)
- –min_split_reads : minimum total split (clipped) read count per insertion (default = 4)
- –min_prox_mapq : minimum mapping quality for proximal (within-cluster) alignments (default = 10)
- –max_N_consensus : exclude reads and consensus breakends with greater than this number of N (undefined) bases (default = 4)
- –exclude_bam : only consider clusters that do not include reads from these BAM(s) (may be comma-delimited list)
- –exclude_readgroup : only consider clusters that to not include reads from these readgroup(s) (may be comma-delimited list)
- –max_bam_count : set maximum number of BAMs involved per insertion
- –insertion_library : pre-select insertions containing sequence from specified FASTA file (not generally recommended but may improve running time in some instances)
- –map_tabix : tabix-indexed BED of mappability scores. Generate for human with script in lib/human_mappability.sh.
- –min_mappability : minimum mappability for cluster (default = 0.5; only effective if –map_tabix is also specified)
- –max_disc_fetch : maximum number of discordant mates to fetch per insertion site per BAM. Sites with more than this number of discordant reads associated will be downsampled. This helps with runtime as fetching discordant rates is time-consuming (default = 50).
- –min_disc_reads : sets the threshold for calling a cluster of discordant reads when using -d/–disco_target (default = 4)
- –tmpdir : directory for temporary files (default = /tmp)

4 Resolution of specific insertion types (resolve.py)

4.1 Usage

```
usage: resolve.py [-h] -p PICKLE [-t PROCESSES] -i INSLIB_FASTA
                  [-m FILTER_BED] [-v] [-o OUT] [-r REF]
                  [--max_bam_count MAX_BAM_COUNT]
                  [--min_ins_match MIN_INS_MATCH]
                  [--min_ref_match MIN_REF_MATCH]
                  [--min_cons_len MIN_CONS_LEN] [--min_discord MIN_DISCORD]
                  [--min_split MIN_SPLIT] [--ignore_filters]
                  [-a ANNOTATION_TABIX] [--refoutdir REFOUTDIR] [--use_rg]
                  [--keep_all_tmp_bams] [--detail_out DETAIL_OUT] [--unmapped]
```

```
                  [--usecachedLAST] [--uuid_list UUID_LIST] [--callmuts]
                  [--tmpdir TMPDIR]

Resolve insertions from TEbreak data

optional arguments:
  -h, --help            show this help message and exit
  -p PICKLE, --pickle PICKLE
                        pickle file output from tebreak.py
  -t PROCESSES, --processes PROCESSES
                        split work across multiple processes
  -i INSLIB_FASTA, --inslib_fasta INSLIB_FASTA
                        reference for insertions (not genome)
  -m FILTER_BED, --filter_bed FILTER_BED
                        BED file of regions to mask
  -v, --verbose         output detailed status information
  -o OUT, --out OUT     output table
  -r REF, --ref REF     reference genome fasta, expect bwa index, triggers
                        transduction calling
  --max_bam_count MAX_BAM_COUNT
                        skip sites with more than this number of BAMs (default
                        = no limit)
  --min_ins_match MIN_INS_MATCH
                        minumum match to insertion library (default 0.95)
  --min_ref_match MIN_REF_MATCH
                        minimum match to reference genome (default 0.98)
  --min_cons_len MIN_CONS_LEN
                        min total consensus length (default=250)
  --min_discord MIN_DISCORD
                        minimum mapped discordant read count (default = 8)
  --min_split MIN_SPLIT
                        minimum split read count (default = 8)
  --ignore_filters
  -a ANNOTATION_TABIX, --annotation_tabix ANNOTATION_TABIX
                        can be comma-delimited list
  --refoutdir REFOUTDIR
                        output directory for generating tebreak references
                        (default=tebreak_refs)
  --use_rg              use RG instead of BAM filename for samples
  --keep_all_tmp_bams   leave ALL temporary BAMs (warning: lots of files!)
  --detail_out DETAIL_OUT
                        file to write detailed output
  --unmapped            report insertions that do not match insertion library
  --usecachedLAST       try to used cached LAST db, if found
  --uuid_list UUID_LIST
                        limit resolution to UUIDs in first column of input
                        list (can be tabular output from previous resolve.py
                        run)
```

```
--callmuts            detect changes in inserted seq. vs ref. (requires
                      bcftools)
--tmpdir TMPDIR       directory for temporary files
```

4.2 Description

This script is the second step in insertion analysis via TEBreak, it is separated from the initial insertion discovery script (tebreak.py) to facilitate running multiple different analyses on the same set of putative insertion sites (e.g. detecting transposable elements, viral insertions, processed transcript insertions, and novel sequence insertions from the same WGS data).

4.3 Input

Insertion call input (-p/–pickle) This is the 'pickle' containing information about putative insertion sites derived from tebreak.py (filename specified by –pickle).

Reference genome (-i/–inslib A FASTA file containing template insertion sequences (e.g. reference transposable elements, viral sequences, mRNAs, etc.). For transposable elements, sequence superfamilies and subfamilies can be specified by separating with a colon (:) as follows:

```
>ALU:AluYa5
GGCCGGGCGCGGTGGCTCACGCCTGTAATCCCAGCACTTTGGGAGGCCGAGGCGGGCGGATCACGAGGTCAGGAGATCG
AGACCATCCCGGCTAAAACGGTGAAACCCCGTCTCTACTAAAAATACAAAAAATTAGCCGGGCGTAGTGGCGGGCGCCT
GTAGTCCCAGCTACTTGGGAGGCTGAGGCAGGAGAATGGCGTGAACCCGGGAGGCGGAGCTTGCAGTGAGCCGAGATCC
CGCCACTGCACTCCAGCCTGGGCGACAGAGCGAGACTCCGTCTCAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

Detailed human-readable output (–detail_out) This is a file containing detailed information about consensus reads, aligned segments, and statistics for each putative insertion site detected. Note that this is done with minimal filtering, so these should not be used blindly. Default filename is the input BAM filename minus the .bam extension plus .resolve.out

Tabular output (-o/–out) Filename for the output table which is usually the primary means to assess insertion detection. This format is described in detail later in this document. Default filename is the input BAM filename minus the .bam extension plus .table.txt

Additional options

- -t/–threads : Split work across multiple processes.
- -m/–filter_bed : BED file of regions to mask (will not be output)
- -v/–verbose : Output status information.
- –max_bam_count : do not analyse insertions represented by more than this number of BAMs (default = no filter)
- –min_ins_match : minimum percent match to insertion library
- –min_ref_match : minimum percent match to reference genome

- –min_discord : minimum number of discordant reads associated with insertion sites (default = 8)
- –min_cons_len : minimum consensus length (sum of 5p and 3p insertion consensus contigs)
- –min_split : minimum number of split reads associated with insertion sites (default = 8)
- –annotation_tabix : tabix-indexed file, entries overlapping insertions will be annotated in output
- –refoutdir : non-temporary output directory for various references generated by resolve.py. This includes LAST references for the insertion library and insertion-specific BAMs if –keep_ins_bams is enabled.
- –use_rg : use the readgroup name instead of the BAM name to count and annotate samples
- –keep_all_tmp_bams : retain all temporary BAMs in temporary directory (warning: this can easily be in excess of 100000 files for WGS data and may lead to unhappy filesystems).
- –unmapped : also report insertions that do not match the insertion library
- –usecachedLAST : useful if -i/–inslib FASTA is large, you can use a pre-built LAST reference (in –refoutdir) e.g. if it was generated on a previous run.
- –uuid_list : limit analysis to set of UUIDs in the first column of specified file (generally, this is the table output by a previous run of resolve.py) - this is useful for changing annotations, altering parameters, debugging, etc.
- –callmuts : reports changes in inserted sequence vs insertion reference in the 'Variants' column of the tabular output
- –tmpdir : directory for temporary files (default is /tmp)

4.4 Output

A table (tab-delimited) is output (with a header) is written to the file specified by -o/–out in resolve.py. The columns are as follows:

- UUID : Unique identifier
- Chromosome : Chromosome
- Left_Extreme : Coordinate of the left-most reference base mapped to
- –te : enable additional filtering specific to transposable element insertions a read supporting the insertion
- Right_Extreme : Coordinate of the right-most reference base mapped to a read supporting the insertion
- 5_Prime_End : Breakend corresponding to the 5' end of the inserted sequence (where 5' is can be defined e.g. for transposable elements)
- 3_Prime_End : Breakend corresponding to the 3' end of the inserted sequence (where 3' is can be defined e.g. for transposable elements)

- Superfamily : Superfamily of insertions of superfamily is defined (see option -i/–inslib in resolve.py)
- Subfamily : Subfamily of insertions of subfamily is defined (see option -i/–inslib in resolve.py)
- TE_Align_Start : alignment start position within inserted sequence relative to reference
- TE_Align_End : alignment end position within inserted sequence relative to reference
- Orient_5p : Orientation derived from the 5' end of the insertion (if 5' is defined), assuming insertion reference is read 5' to 3'
- Orient_3p : Orientation derived from the 3' end of the insertion (if 3' is defined), assuming insertion reference is read 5' to 3'
- Inversion : If Orient_5p and Orient_3p disagree, there may be an inversion in the inserted sequence relative to the reference sequence for the insertion
- 5p_Elt_Match : Fraction of bases matched to reference for inserted sequence on insertion segment of 5' supporting contig
- 3p_Elt_Match : Fraction of bases matched to reference for inserted sequence on insertion segment of 3' supporting contig
- 5p_Genome_Match : Fraction of bases matched to reference genome on genomic segment of 5' supporting contig
- 3p_Genome_Match : Fraction of bases matched to reference genome on genomic segment of 3' supporting contig
- Split_reads_5prime : Number of split (clipped) reads supporting 5' end of the insertion
- Split_reads_3prime : Number of split (clipped) reads supporting 5' end of the insertion
- Remapped_Discordant : Number of discordant read ends re-mappable to insertion reference sequence
- Remap_Disc_Fraction : The proportion of remapped discordant reads mapping to the reference insertion sequence
- Remapped_Splitreads : Number of split reads re-mappable to insertion reference sequence
- Remap_Split_Fraction : The proportion of remapped split reads mapping to the reference insertion sequence
- 5p_Cons_Len : Length of 5' consensus sequence (Column Consensus_5p)
- 3p_Cons_Len : Length of 3' consensus sequence (Column Consensus_3p)
- 5p_Improved : Whether the 5' consensus sequence was able to be improved through local assembly in tebreak.py
- 3p_Improved : Whether the 3' consensus sequence was able to be improved through local assembly in tebreak.py

- TSD_3prime : Target site duplication (if present) based on 5' overlap on consensus sequence
- TSD_5prime : Target site duplication (if present) based on 3' overlap on consensus sequence (can be different from 5' end, but in many cases it is advisable to filter out putative insertions that disagree on TSDs)
- Sample_count : Number of samples (based on BAM files or readgroups, depending on –use_rg option in resolve.py)
- Sample_support : Per-sample split read count supporting insertion presence (Sample|Count)
- Genomic_Consensus_5p : Consensus sequence for 5' supporting contig assembled from genome side of breakpoint
- Genomic_Consensus_3p : Consensus sequence for 3' supporting contig assembled from genome side of breakpoint
- Insert_Consensus_5p : Consensus sequence for 5' supporting contig assembled from insertion side of breakpoint
- Insert_Consensus_3p : Consensus sequence for 3' supporting contig assembled from insertion side of breakpoint
- Variants : if –callmuts option given to resolve.py, this is a list of changes detected between inserted sequence and insertion reference sequence
- Genotypes : per-sample read depth information useful for determining genotype

5 Filtering

5.1 Pre-filtering

In some cases it is not necessary to analyse the entire genome, or only specific regions are of interest. Regions of interest in BED format may be input to tebreak.py via the -i/–inverval_bed option.

5.2 Post-filtering

The insertion resolution script (resolve.py) includes a number of options for filtering insertions, with default parameters generally being tuned towards high sensitivity for high-quality WGS samples at greater than 30x coverage. In many instances, additional filtering may be necessary. For example, false positives may arise due to homopolymer expansions and contrations around reference transposable elements, as is particularly the case in mismatch-repair defective tumour genomes. False positives may also arise from sample preparation methods that induce PCR artefacts such as chimeric reads. Generally, when searching for rare events, the decreased signal-to-noise ratio necessitates a low false positive rate. To perform additional filtering, a script is included in scripts/general_filter.py which should be regarded as a starting point for obtaining an optimal analysis.

5.3 Miscellaneous methods for improving runtime

- For whole genome sequencing (WGS) data, it is only the soft-clipped reads and reads in discordant pairs that are informative; the rest of the mapped reads can be discarded. This can be accomplished using the script 'reduce_bam.py', located in the scripts directory. Preprocessing with this script may yield lower runtime due to less disk access.

- For sequence derived from targeted sequencing methods (e.g. whole exome sequencing), it may be useful to only analyse regions covered to at least a certain read depth. This can be accomplished using the script 'covered_segments.py' in the scripts directory.

- To improve the runtime of resolve.py over large datasets (e.g. WGS) the 'pickle' generated by tebreak.py can be reduced using picklescreen.py (in the scripts directory). This will pre-align all junction reads to an insertion reference library, reducing the number of candidates for further analysis by resolve.py.

- For larger cohorts, splitting the 'pickle' file up into per-chromosome files may be required to reduce memory usage. This can be accomplished with the 'picklesplit.py' script in the scripts directory.