Using the command line tool

Tim Millar

2017-07-31

Contents

TEF

The basic 'tef' tool is used as a wrapper for included programs: `preprocess`, `fingerprint`, `compare` and `filter-gff`. The help text can be displayed for `tef` or any of the wrapped tols with the `--help` or `-h` flag e.g:

```
tef -h
tef preprocess -h
tef fingerprint -h
tef compare -h
tef filter-gff -h
```

Preprocess Pipeline

The preprocessing pipeline requires a paired-end reads in fastq format, a library of tranposable elements in fasta format and a reference genome in fasta format. Fasta files should be pre indexed using BWA. This pipeline also requires that BWA and Samtools are installed and on the users `$PATH`.

Paired end reads are initially aligned to the library of transposable elements manually with bwa mem. The preprocess tool is then used to identify and extract unmapped reads with mapped mates (dangler reads) which are mapped against the reference genome and tagged with the ID of their mates transposable element.

You should have the following four files:

- `reads1.fastq` and `reads2.fastq`: paired-end reads in fastq format.
- `reference.fasta` a reference genome in fasta format.
- `repeats.fasta` a library of repeat elements in fasta format.

Initial alignment

Index fasta files with bwa:

```
bwa index -a bwtsw reference.fasta
bwa index -a is repeats.fasta
```

Map paired end reads to repeat elements using `bwa mem` and convert to a sorted bam file with `samtools`:

```
bwa mem -t 8 \
    repeats.fasta \
    reads1.fastq \
    reads2.fastq \
    | samtools view -Su - \
    | samtools sort - \
    -o reads_mapped_to_repeats.bam
```

Index the resulting bam file with `samtools`:

```
samtools index reads_mapped_to_repeats.bam
```

Preprocess tool

Runing the preprocess program on the indexed bam file:

```
tef preprocess \
    reads_mapped_to_repeats.bam \
    --reference reference.fasta \
    --output danglers.bam \
    --threads 8
```

Arguments:

- Input bam file
- `-r/--reference` reference genome to align informative reads to.
- `-o/--output` the name of the output (indexed) bam file.

- `-t/--threads` specifies how many threads to use for alignment in bwa (python components of the pipeline are single threaded).

Additional arguments:

- `--exclude-tails` by default the soft-clipped tails of pairs properly mapping to a single repeat element are included as an additional source of information. This option will exclude soft-clipped tails from the resulting bam file.
- `--tail-minimum-length` the minimum length allowed for soft-clipped tails to be included (defaults to `38`).
- `--tempdir` by default, the intermediate files are written to a temporary directory that is automatically removed when the pipeline is completed. These files can be saved by manually specifying a directory with theis option.
- `--mate-element-tag` by default, the same tag used to store repeat element names associated with each read is `ME` (Mate Element). This can be changed with the option.

The output file `danglers.bam` contains reads mapped to the reference genome. Each of these reads is tagged with the repeat element that their pair was mapped to.

Fingerprint

Example usage:

```
tef fingerprint danglers.bam \
    -f family1 family2 ... \
    -m 20 \
    -e 500 \
    -q 30 \
    -t 4 \
    > fingerprint.gff
```

Where `danglers.bam` is the bam file being fingerprinted and `fingerprint.gff` is the output gff file.

Arguments:

- A bam file to be fingerprinted
- `-r/--references` may optionally be used to specify a subset of chromosomes to fingerprint. By default all reference chromosomes are fingerprinted (based on the bam header).
- `-f/--families` specifies the (super) families or grouping of repeated elements to fingerprint. These names are matched against the start of the

mate element name i.e. the name `Gypsy` would treat reads with tagged with a mate element called `Gypsy3`, `Gypsy27` or `GypsyX` as the same.
- `-m/--minimum-reads` specifies the minimum number of read (tips) required to form a cluster.
- `-e/--epsilon` specifies the maximum allowable distance among a set of read tips to be considered a cluster.
- `-q/--mapping-quality` specifies the minimum mapping quality allowed for reads (defaults to `30`).
- `-t/--threads` specifies the number of CPU threads to use. The maximum number of threads that may be used is the same as the number of references specified.

Additional arguments:

- `--minimum-epsilon` the minimum value of epsilon to be used in hierarchical clustering (defaults to `0`).
- `--non-hierarchical` by default a hierarchical clustering algorithm is used. This flag will switch to the non-hierarchical version.
- `--mate-element-tag` the sam tag used to specify the name of each reads mate element (defaults to `ME`).
- `--feature_csv` optionally specify the name of a CSV file to output containing feature data.

Compare

Example usage:

```
tef compare danglers1.bam danglers2.bam ... \
    -f family1 family2 ... \
    -m 20 \
    -e 500 \
    -b 50 \
    -t 4 \
    > comparison.gff
```

Where `danglers1.bam ...` are the bam files being compared and `comparison.gff` is the output gff file.

Arguments:

- At least two bam files to be compared.
- `-r/--references` may optionally be used to specify a subset of chromosomes to fingerprint. By default all reference chromosomes are fingerprinted (based on the bam header).
- `-f/--families` specifies the (super) families or grouping of repeated elements to fingerprint. These names are matched against the start of the

mate element name i.e. the name `Gypsy` would treat reads with tagged with a mate element called `Gypsy3`, `Gypsy27` or `GypsyX` as the same.

- `-m/--minimum-reads` specifies the minimum number of read (tips) required to form a cluster.
- `-e/--epsilon` specifies the maximum allowable distance among a set of read tips to be considered a cluster.
- `-q/--mapping-quality` specifies the minimum mapping quality allowed for reads (defaults to `30`).
- `-b/--buffer-fingerprints` specifies a distance (in base pairs) to buffer fingerprints by before combining them into comparative bins (defaults to `0`). This is used to ensure that small clusters, that are slightly offset in different samples, are treated as a single comparative bin. It also improves the robustness of comparisons by allowing more reads to be included in each bin.
- `-t/--threads` specifies the number of CPU threads to use. The maximum number of threads that may be used is the same as the number of references specified.

Additional arguments:

- `--long-form-gff` optional flag to produce a GFF file in which each comparative bin is duplicated for each input bam file to avoid nested lists of counts or source names.
- `--minimum-epsilon` the minimum value of epsilon to be used in hierarchical clustering (defaults to `0`).
- `--non-hierarchical` by default a hierarchical clustering algorithm is used. This flag will switch to the non-hierarchical version.
- `--mate-element-tag` the sam tag used to specify the name of each reads mate element (defaults to `ME`).
- `--buffer-comparative-bins` similar to `--buffer-fingerprints` but buffering is performed after fingerprints are combined, therefore less likely to combine slightly offset clusters (defaults to `0`).
- `--feature-csv` optionally specify the name of a CSV file to output containing (long-form) feature data. This produces one row of data per sample per feature.
- `--character-csv` optionally specify the name of a CSV file to output containing a matrix of read counts. This produces one column per feature by one row per sample.

Filter GFF

This script can be used to filter down the results of `fingerprint` or `compare`. Filters can be applied to attributes in the attribute column or to the first 8 standard gff3 columns.

Multiple filters may be combined, in which case a feature must pass all of them to be kept.

If an attribute contains a comma separated list of values e.g. `proportions=0.9,0.1,0.0` only one of the values must pass the filter for the feature to be retained.

Filters take the form `'<column/attribute><operator><value>'` where:

- `<column/attribute>` is the name of the column or attribute that the filter is applied to.
- `<operator>` is one of the following operators `=`, `==`, `!=`, `< >`, `>=`, `<=` that describes the comparason being performed.
- `<value>` is the value the each feature is compared to.

Filters should be contained within quotes `''` so that the operator is not interpreted as a shell command.

The following operators are only used for numerical comparisons: `< >`, `>=`, `<=`.

The operators `=`, `==` and `!=` will try to compare values as numerical (floating points) but will also check for equivalence or non-equivalence of string values. Note that `=`, `==` are identical.

Example usage with one column filter and two attribute filters:

```
tef filter_gff comparison.gff \
    -c 'seqid=chr1' \
    -a 'category=Gypsy' 'proportions>=0.95' \
    > comparison_filtered.gff
```

Where `comparison.gff` is a gff file and `comparison_filtered.gff` is a filtered version of that file.

Arguments:

- `-c/--column-filters` filters to apply to the first 8 standard gff3 columns. These should take the form `'<column><operator><value>'`
- `-a/--attribute-filters` filters to apply to the attributes column. These should take the form `'<attribute><operator><value>'`