Supplementary Note - Manual

This note provides a guide to help users use TEmarker in a population.

Prerequisites

1. **Python**

   Available at https://www.python.org/downloads/. Developed and tested with version 3.7.1.

2. **Python packages**

   1) biopython: pip install biopython

Required tools

1. **samtools (**v0.1.19-44428cd**)**

   Available at https://github.com/samtools/samtools. Developed and tested with version 0.1.19-44428cd.

2. **hisat2** (v2.1.0**)**

   Available at http://daehwankimlab.github.io/hisat2/download/. Developed and tested with version 2.1.0.

3. **McClintock**

   Available at https://github.com/bergmanlab/mcclintock.

Optional tools

1. **bwa** (v0.7.10-r789)

   Available at https://github.com/lh3/bwa. Tested with version 0.7.10-r789.

2. **CNVnator** (v0.3.2)

   Available at https://github.com/abyzovlab/CNVnator. Tested with version 0.3.2.

3. **Any other polymorphic TE detection tools.**

Example

Four steps should be running to finish the whole process:

Step 0: Prepare input data (TE insertion files; hisat2 mapping files)

Step 1: Create pan-TEs

Step 2: Genotyping

Step 3: Generate output

Here we provide testing data for all the steps users can download following the command lines.

Step 0: Prepare candidate TE insertions in a population

Required files:

1. **Reference genome file**

   ('fasta' format)

2. **TE consensus sequences or TE library file**

   ('fasta' format)

3. **File from paired end reads**

   ('fastq' format)

Download testing required data

Commands (Step 0 data):

- mkdir output_dir
- Download_test_data.py -o output_dir -s0 yes
- tar -xvzf Example_dir_step0.tar.gz
- tar -xvzf Example_dir_step0/SRR800842_1.fastq.tar.gz Example_dir_step0/SRR800842_2.fastq.tar.gz
- mkdir fastq_dir
- mv Example_dir_step0/SRR800842* fastq_dir

Outputs (Step 0 data):

Example_dir_step0 (test_genome.fasta; TE.lib; fastq_dir)

Note: The data are the same test data used in the McClintock tool. The test_genome.fasta is UCSC sacCer2 yeast reference genome. The TE.lib is an annotation of TEs in the yeast reference genome from Martin et al. (2012). fastq_dir that contains multiple paired-end fastq files for each sample.

Tool Installment

Download TEmarker

- git clone https://github.com/yanhaidong1/TEmarker.git.

TEmarker needs at least one polymorphic TE detection tools. Here we provide several tools referred from the relative GitHub download page. Users can choose at least one of them to run. The McClintock tool is suggested to be used.

a. McClintock tool

install conda

install McClintock (https://github.com/bergmanlab/mcclintock)

- git clone git@github.com:bergmanlab/mcclintock.git
- cd mcclintock
- conda env create -f install/envs/mcclintock.yml --name mcclintock
- conda activate mcclintock
- python3 mcclintock.py --install

b. jitterbug tool

install jitterbug tool (https://github.com/elzbth/jitterbug)

git clone https://github.com/elzbth/jitterbug.git

c. TEFLoN tool

install TEFLoN tool (https://github.com/jradrion/TEFLoN)

git clone https://github.com/jradrion/TEFLoN.git

Running

1. McClintock tool

Inputs:

1. test_genome.fasta
2. TE.lib
3. fastq_dir (multiple paird-end fastq files for each sample)

Commands:

- conda activate mcclintock
- mkdir working_dir_step0
- mkdir output_dir_step0
- python TEmarker_bed.py \

```
-d working_dir_step0 \
-o output_dir_step0 \
–mcclintock /path/to/mcclintock.py \
-fastq_d fastq_dir \
-fas_f test_genome.fasta \
-lib_f TE.lib \
-tool ngs_te_mapper,retroseq,temp,te-locate \
-pros_n 10
```

Note: for '-tool', MCclintcok tool provides eight tools that can be run: ngs_te_mapper, relocate, relocate2, temp, retroseq, popoolationte, popoolationte2, te-locate. Here, we only call four of them to be an example. If users want to call all of them, they do not need '-tool'.

Outputs:

1. bed_dir that contains TE location files with 'bed' format.
2. bwa_bam_dir that contains mapping file with 'bam' format generated from bwa tool wrapped in the McClintock tool.

2. jitterbug and TEFLoN tools

Inputs:

1. test_genome.fasta
2. TE.lib
3. fastq_dir
4. bwa_bam_dir (obtain from the mcclintock tool output; users can also prepare by themselves)

Note: These two tools need to be run under the python2 environment.

Commands:

- conda create -n python2 python=2.7
- conda activate python2
- conda install -c bioconda samtools
- conda install -c bioconda bwa
- conda install -c conda-forge perl-text-soundex
- conda install -c anaconda biopython
- mkdir working_dir_step0
- mkdir output_dir_step0
- python TEmarker_bed.py \
  -d working_dir_step0 \
  -o output_dir_step0 \
  –TEFLoN_d /Path/to/TEFLoN/ \
  –jitterbug_d /Path/to/jitterbug/ \
  –repeatmasker /Path/to/RepeatMasker \
  -fastq_d fastq_dir \
  -fas_f test_genome.fasta \
  -lib_f TE.lib \
  -bam_bwa_d bwa_bam_dir \
  -pros_n 10

Outputs:

1. bed_dir that contains TE location files with 'bed' format.

Step 0: Prepare hisat2 mapping files

Inputs:

1. test_genome.fasta
2. fastq_dir
3. material_name.txt [This file is used to change original name (the first column) to a new name (the second column)]

Commands:

- mkdir working_dir_step0
- mkdir output_dir_step0
- TEmarker_tool.py \
  -d working_dir_step0 \
  -o output_dir_step0 \
  -fas_f test_genome.fasta \

```
-fastq_d fastq_dir \
-hst yes \
–hisat2_t /path/to/hisat2 \
–hisat2build /path/to/hisat2build \
-pros_n_hisat2 10 \
–samtools /path/to/samtools \
-m_f material_name.txt \
-clean_temp yes
```

Outputs:

1. hisat2_bam_dir that contains mapping file with ‘bam’ format generated from hisat2 tool.

Note:

1. Users can also use ‘-bwa’ to generate the bwa bam dir if they do not want use mapping files generated from the McClintock tool.
2. Users can use -clean_temp to remove all the sam files generated during the pipeline to save space.

Step 1,2,3: Create pan-TEs; Genotyping; Generate output

Required files:

1. **Reference genome file**

   (test_genome.fasta)
2. **TE consensus sequences or TE library file**

   (TE.lib)
3. **A directory contains TE insertion file (**obtained from **Step 0)**

   (bed_dir)
4. **A directory contains bwa mapping file (**obtained from **Step 0)**

   (bwa_bam_dir)
5. **A directory contains hisat2 mapping file (**obtained from **Step 0)**

   (hisat2_bam_dir)

These five datasets can be downloaded from the following scripts. We will use the downloaded files to be examples.

Download testing required data

- mkdir output_dir
- Download_test_data.py -o output_dir -s123 yes

Outputs:

1. A test genome file (test_genome.fasta; 5 Mb) derived from a part of rice genome (Os-Nipponbare-Reference-IRGSP-1.0).
2. A test TE library file (TE.lib).
3. A test directory including bed files for each sample (bed_dir).
4. A test directory including bam files derived from bwa tool for each sample (bwa_bam_dir).

5. A test directory including bam files derived from hisat2 tool for each sample (hisat2_bam_dir).
6. A test material name file (material_name.txt) that is used to change original name (the first column) to a new name (the second column).

Step 1: Create pan-TEs

Step 1.1: Modify bam files

This step is to remove the PCR duplicates using samtools tool, sort, and index the bam files.

Inputs:

1. bed_dir
2. hisat2_bam_dir
3. bwa_bam_dir

Commands:

- `mkdir working_dir_step1_1`
- `mkdir output_dir_step1_1`

-
  ```
  TEmarker_bam.py \
  -d working_dir_step1_1 \
  -o output_dir_step1_1 \
  -bam_bwa_d bwa_bam_dir \
  -bam_hisat2_d hisat2_bam_dir \
  --samtools /path/to/samtools \
  -pros_n_samtls 10 \
  -clean_temp yes
  ```

Outputs:

1. m_hisat2_bam_dir
2. m_bwa_bam_dir

Step 1.2: Create pan-TEs

Inputs:

1. bed_dir

Commands:

- `mkdir working_dir_step1_2`
- `mkdir output_dir_step1_2`

-
  ```
  TEmarker_panTEs.py \
  -d working_dir_step1_2 \
  -o output_dir_step1_2 \
  -b_d bed_dir
  ```

Outputs:

1. opt_pan_TEs.txt

Step 2: Genotyping

Inputs:

1. opt_pan_TEs.txt
2. m_hisat2_bam_dir
3. m_bwa_bam_dir
4. TE.lib

Commands:

- `mkdir working_dir_step2`
- `mkdir output_dir_step2`

-
  ```
  TEmarker_genotyping.py \
  -d working_dir_step2 \
  -o output_dir_step2 \
  -panTEs_f output_dir_step1_2/opt_pan_TEs.txt \
  -bam_bwa_d output_dir_step1_1/m_bwa_bam_dir \
  -bam_hisat_d output_dir_step1_1/m_bwa_bam_dir \
  -lib_f TE.lib \
  -pros_n 20
  ```

Outputs:

1. opt_te_genotype.txt

Note:

1. By setting '-pros_n', users can speed up the genotyping.

Optional step: Modify genotyping

Since TEs may locate at regions with abundant copy number variations, we provide 'TEmarker_tool.py' to generate copy number variation files based on **CNVnator** tool.

Inputs:

1. m_bwa_bam_dir
2. test_genome.fasta

Commands:

- `mkdir working_dir_cnv`
- `mkdir output_dir_cnv`

-
  ```
  TEmarker_tool.py \
  -d working_dir_cnv \
  -o output_dir_cnv \
  ```

```
-cnv yes \
-cnvnator_t /path/to/cnvnator \
-fas_f test_genome.fasta \
-bam_d output_dir_step1_1/m_bwa_bam_dir
```

Outputs:

1. output_dir_cnv

After we obtain output_dir_cnv, we can run the TEmarker_genotyping.py by calling -cnv_d.

Inputs:

1. opt_pan_TEs.txt
2. m_hisat2_bam_dir
3. m_bwa_bam_dir
4. TE.lib
5. output_dir_cnv

Commands:

- `mkdir working_dir_step2`
- `mkdir output_dir_step2`

-
  ```
  TEmarker_genotyping.py \
  -d working_dir_step2 -o output_dir_step2 \
  -canTE_f output_dir_step1_2/opt_pan_TEs.txt \
  -bam_bwa_d output_dir_step1_1/m_bwa_bam_dir \
  -bam_hisat_d output_dir_step1_1/m_bwa_bam_dir \
  -lib TE.lib \
  -pros_n 10 \
  -modify yes \
  -cnv_d output_dir_cnv
  ```

Outputs:

1. opt_modified_te_genotype.txt

Step 3: Generate output

Inputs:

1. opt_te_genotype.txt or opt_modified_te_genotype.txt
2. test_genome.fasta
3. material_name.txt

Commands:

- `mkdir working_dir_step3`
- `mkdir output_dir_step3`

```
• TEmarker_output.py \
  -d working_dir_step3 -o output_dir_step3 \
  -genos_f output_dir_step2/opt_te_genotype.txt \
  -m_f material_name.txt \
  -fas_f test_genome.fasta
```

Outputs:

1. vcf_output_dir
    1) opt.vcf (no filtration)
    2) opt_bi.vcf (transfer ‘0/1’ to ‘1/1’)
    3) opt_fltmissing_fltmaf.vcf (remove loci based on missing sample and maf thresholds)
3. genos_output_dir
    1) opt_genos.txt
    2) opt_genos_fltmissing_fltmaf.txt
4. summary of TE families
    1) opt_summary_te_family.txt

Users can use **TEmarker_pipeline.py** to run **Step 1, 2, 3**, but need to complete **Step 0** first.

Commands:

```
• mkdir working_dir
• mkdir output_dir
• TEmarker_pipeline.py \
  -d working_dir \
  -o output_dir \
  -fas_f test_genome.fasta \
  -lib_f TE.lib \
  -b_d bed_dir \
  -bam_bwa_d bwa_bam_dir \
  -bam_hisat2_d hisat2_bam_dir \
  -m_f material_name.txt \
  -TEmarker_p /Path/to/TEmarker \
  --samtools /Path/to/samtools \
  -pros_n 20
```

Outputs:

2. vcf_output_dir
    2) opt.vcf (no filtration)
    3) opt_bi.vcf (transfer ‘0/1’ to ‘1/1’)

    4) opt_fltmissing_fltmaf.vcf (remove loci based on missing sample and maf thresholds)

5. genos_output_dir
    1) opt_genos.txt
    2) opt_genos_fltmissing_fltmaf.txt
6. summary of TE families
    1) opt_summary_te_family.txt

An overview of all parameters

Table all parameters information of TEmarker scripts.

Download_test_data.py	
-o [output directory]	Output directory stores the output files.
-s0 [s0_yes]	If users initiate this argument, they will download testing data for step 0.
-s123 [s123_yes]	If users initiate this argument, they will download all the testing data for step123.
TEmarker_bed.py	
Required parameters	
-d [working directory]	Working directory stores intermediate files of each step. Attention: please provide the absolute path of working directory.
-o [output directory]	Output directory stores the output files.
--mcclintock [mcclintock tool]	Specify an absolute path to mcclintock executable.
-fastq_d [fastq directory]	Specify an absolute path to a fastq directory that contains all the fastq files.
-fas_f [genome fasta file]	Specify an absolute path to a reference genome file.
-lib_f [TE library]	Specify an absolute path to a TE library file.
Optional parameters	
-m_f [material file]	Specify a path to a material file that includes two columns. The first column is original sample name and the second name is a new sample name that users want to replace.
-tool [tool list]	Specify tools users want to use in the mcclintock tool. Default: all. Alternative: ngs_te_mapper,relocate,relocate2,temp,retroseq,popoolationte,popoolationte2,te-locate.
-clean [yes]	If users initiate this argument, they will delete the files in the working directory to save space.
-p [processor number]	Specify processor number. Default: 12.
--TEFLoN_d [TEFLoN tool folder]	Specify TEFLoN tool folder and use python2 conda environment to run (see examples to use this argument).
--jitterbug_d [jitterbug tool folder]	Specify jitterbug tool folder and use python2 conda environment to run (see examples to use this argument).
--repeatmasker [RepeatMasker]	Specify a path to RepeatMasker executable once initiating **--TEFLoN_d** and **--jitterbug_d**.
-bam_bwa_d [mapping files folder from bwa tool]	Specify mapping files directory once initiating **--jitterbug_d**. **TEmarker_tool.py** can generate bam files.
--bwa [bwa]	Specify a path to bwa tool once initiating **--TEFLoN_d**.
--samtools [samtools]	Specify a path to samtools tool once initiating **--TEFLoN_d**.

TEmarker_tool.py	
Required parameters	
-d [working directory]	Working directory stores intermediate files of each step.
-o [output directory]	Output directory stores the output files.
-fas_f [genome fasta file]	Specify a path to a reference genome file.
Optional parameters	
##Use hisat2 to generate bam files	
-hst [yes]	If users initiate this argument, they need to initiate **--hisat2_t** and **--hisat2build** to generate bam files. Meanwhile, users need to provide a path of samtools using **--samtools** and fastq directory using **-fastq_d**.
--hisat2 [hisat2 tool]	Specify a path to hisat2 executable once initiating the **-hst**.
--hisat2build [hisat2-build]	Specify a path to hisat2-build executable once initiating the **-hst**.
-pros_n_hisat2 [processor number]	Specify processor number. Default: 1.
##Use bwa to generate bam files	
-bwa [yes]	If users initiate this argument, they need to initiate **--bwa_t** to generate bam files. Meanwhile, users need to provide a path of samtools using **--samtools** and fastq directory using **-fastq_d**.
--bwa_t [bwa tool]	Specify a path to bwa executable once initiating the **-bwa**.
-pros_n_bwa [processor number]	Specify processor number. Default: 1.
##Dependences of hisat2 and bwa tools	
--samtools [samtools tool]	Once initiating **-hst** or **-bwa**, users need to specify a path to samtools executable.
-fastq_d [fastq directory]	Once initiating **-hst** or **-bwa**, users need to specify a path to a fastq directory that contains all the fastq files.
##Use cnvnator to generate copy number variation files	
-cnv [yes]	If users initiate this argument, they need to initiate **--cnvnator_t** to generate copy number variation for each individual sample. Meanwhile, users need to provide a path of bam directory derived the bwa tool using **-bam_d**.
--cnvnator_t [cnvnator tool]	Specify a path to cnvnator executable once initiating the **-cnv**.
-bam_d [bwa bam directory]	Specify a path to a directory that stores bam files derived from bwa tool for each indivdual sample once initiating the **-cnv**.
##Dependences for all three tools: hisat2, bwa, and cnvnator	
-m_f [material file]	Specify a path to a material file that includes two columns. The first column is original sample name and the secon name is a new sample name that users want to replace.
-clean [yes]	If users initiate this argument, they will delete the files in the working directory to save space. In this case, it will delete sam files generated from hisat2 and bwa tools.
TEmarker_pipeline.py	
Required parameters	
-d [working directory]	Working directory stores intermediate files of each step.
-o [output directory]	Output directory stores the output files.

-fas_f [genome fasta file]	Specify a path to a reference genome file.
-lib_f [TE library]	Specify a path to a TE library file.
-b_d [bed directory]	Specify a path to a directory that stores bed files containing TE locations for each individual sample.
-m_f [material file]	Specify a path to a material file that includes two columns. The first column is original sample name and the second name is a new sample name that users want to replace.
-bam_bwa_d [bwa bam directory]	Specify a path to a directory that stores bam files derived from bwa tool for each individual sample.
-bam_hisat2_d [hisat2 bam directory]	Specify a path to a directory that stores bam files derived from hisat2 tool for each individual sample.
-TEmarker_p [TEmarker directory]	Specify a path to the TEmarker directory.
--samtools [samtools tool]	Specify a path to samtools executable.
Optional parameters	
-modify [yes]	If users initiate this argument, they need to provide a directory that includes outputs derived from cnvnator for each individual sample. Users can generate this directory using TEmarker_tool.py in the TEmarker directory.
-cnv_d	If users initiate the **-modify**, they need to provide a path to cnv_dir generated from TEmarker_tool.py in the TEmarker directory.
-pros_n [processor number]	Specify processor number. Default: 1. We suggest users set as many as processor number they can since genotyping step wrapped in this pipeline needs a greater number of processors to increase speed.
-clean [yes]	If users initiate this argument, they will delete the files in the working directory to save space.
TEmarker_bam.py	
Required parameters	
-d [working directory]	Working directory stores intermediate files of each step.
-o [output directory]	Output directory stores the output files.
-bam_bwa_d [bwa bam directory]	Specify a path to a directory that stores bam files derived from bwa tool for each individual sample.
-bam_hisat2_d [hisat2 bam directory]	Specify a path to a directory that stores bam files derived from hisat2 tool for each individual sample.
--samtools [samtools tool]	Specify a path to samtools executable.
Optional parameters	
-pros_n_samtls [processor number]	Specify processor number for samtools tool. Default: 1.
-clean [yes]	If users initiate this argument, they will delete the files in the working directory to save space.
TEmarker_panTEs.py	
Required parameters	
-d [working directory]	Working directory stores intermediate files of each step.
-o [output directory]	Output directory stores the output files.
-b_d [bed directory]	Specify a path to a directory that stores bed files containing TE locations for each individual sample.
Optional parameters	

-c_bp [length of bp that is needed to be combined]	Specify a range that connects insertion. For example, sample 1 has a TE insertion with location 1_100 in chr01, and sample 2 has a TE insertion with location of 1_105 in chr01. There are 5 bp distance between these two locations in the same chromosome 1. If **-c_bp** is set as 5, these two insertions will be combined. If users do not want to combine these two insertions, they can set this value to be 0. Default: 5.
-f_th [filter threshold]	Specify a filter threshold. For example, **-f_th** is set to 0.05. If sample proportion is 0.04, this location with this sample proportion will be filtered out. Sample proportion could be calculated as follows: If a TE is inserted in a location, we calculate number of samples (A) that have this insertion and number of samples (B) without this insertion. If A is smaller than B, sample proportion = A/(A+B). Otherwise, sample proportion = B/(A+B).
TEmarker_genotyping.py	
Required parameters	
-d [working directory]	Working directory stores intermediate files of each step.
-o [output directory]	Output directory stores the output files.
-panTEs_f [candidate TE file]	Specify a path to opt_pan_TEs.txt generated from TEmarker_panTEs.py.
-bam_bwa_d [bwa bam directory]	Specify a path to rm_pcr_bwa_bam_dir generated from TEmarker_bam.py.
-bam_hisat2_d [hisat2 bam directory]	Specify a path to rm_pcr_hisat2_bam_dir generated from TEmarker_bam.py.
-lib_f [TE library]	Specify a path to a TE library file.
Optional parameters	
-acc_thr [an accuracy threshold]	Specify an accuracy rate to decide if reads support a TE deletion. For example, if accuracy threshold is set to 0.8, and reads with higher or equal value to 0.8 are considered to support the TE deletion. Default: 0.8.
-heter_thr [a heterozygous genotype threshold]	Specify a heterozygous genotype threshold to decide genotypes of candidate TE markers in the opt_pan_TEs.txt. For example, if we define a genotype of a TE marker that shows insertion in individual samples comparing with a reference genome. Next, we calculate proportion of reads that support the TE insertion (reads_in_pro). If we set **-heter_thr** to be 0.7, we define the genotypes as follows: 0/0: 0 <= reads_in_pro < 0.3; 1/1: 0.7 < reads_in_pro <=1; 0/1: 0.3 < reads_in_pro <= 0.7. Default: 0.7.
-t_cover_thr	Specify a threshold to remove candidate insertion loci covering number of reads less than the total_cover_thr. Default: 3
-clrd_thr	Specify a threshold to estimate average of clipped reads for each sample. Default: 2

-TSD_thr	Specify a threshold to estimate the TSD length. If the TSD length is larger than the threshold, if these two locations at the two ends of the TSD is larger than the threshold, these two locations will be regarded as two independent candidate TE insertions. Default: 15.
-comb_thr	Specify a threshold to combine very close insertions. If two close insertions have less or equal to 3 bp distance. They will be combined. Default: 3.
-antgap_thr	Specify a threshold to combine close insertions when these two insertions are annotated as the same TE family name during annotation step. If two close insertions with same TE family name have less or equal to 15 bp distance. They will be combined. Default: 15.
-antclgap_thr	Specify a threshold to combine close insertions when these two insertions are annotated as the different TE family names during annotation step. If two close insertions with different TE family names have less or equal to 5 bp distance. They will be combined. Default: 5.
-s_rg_thr	Specify an extended region (bp) at two ends of panTEs (from output of **TEmarker_panTEs.py**) to be the new panTEs used for the genotyping. Default: 50.
-miss_thr	Specify a threshold to remove candidate insertion loci based on the sample number. For example, if thr_miss is equal to 0.7, and the sample number threshold is equal to 0.7*total_sample_number. If sample number with genotype information is over than the 0.7*total_sample_number, this location will be added to compare to choose the best location in the annotation modifcation step. Default: 0.7.
-pros_n [processor number]	Specify processor number. Default: 1. Note: We suggest users set as many as processor number they can since genotyping step wrapped in this pipeline needs a greater number of processors to increase speed.
-modify [yes]	If users initiate this argument, they need to provide a directory that includes outputs derived from cnvnator for each individual sample. Users can generate this directory using TEmarker_tool.py in the TEmarker directory.
-cnv_d	If users initiate the **-modify**, they need to provide a path to cnv_dir generated from TEmarker_tool.py in the TEmarker directory.
-disable_g	Do not conduct genotyping.
-disable_a	Do not conduct annotation.
TEmarker_output.py	
Required parameters	
-d [working directory]	Working directory stores intermediate files of each step.
-o [output directory]	Output directory stores the output files.
-genos_f [genotype file]	Specify a path to opt_te_genotype.txt/opt_modified_te_genotype.txt generated from TEmarker_genotyping.py.
-m_f [material file]	Specify a path to a material file that includes two columns. The first column is original sample name and the second name is a new sample name that users want to replace.

-fas_f [genome fasta file]	Specify a path to a reference genome file.
Optional parameters	
-miss_thr [filter out missing loci threshold]	Specify a threshold to filter out loci without genotyping information in a population. For example, if **-miss_thr** is set to 0.8 for a locus, and the proportion of the samples with missing genotyping in the locus is over 0.2, and this locus will be filtered out. Default: 0.8.
-maf_thr	Specify a MAF (Minor Allele Frequency) filtration threshold. Default: 0.05.