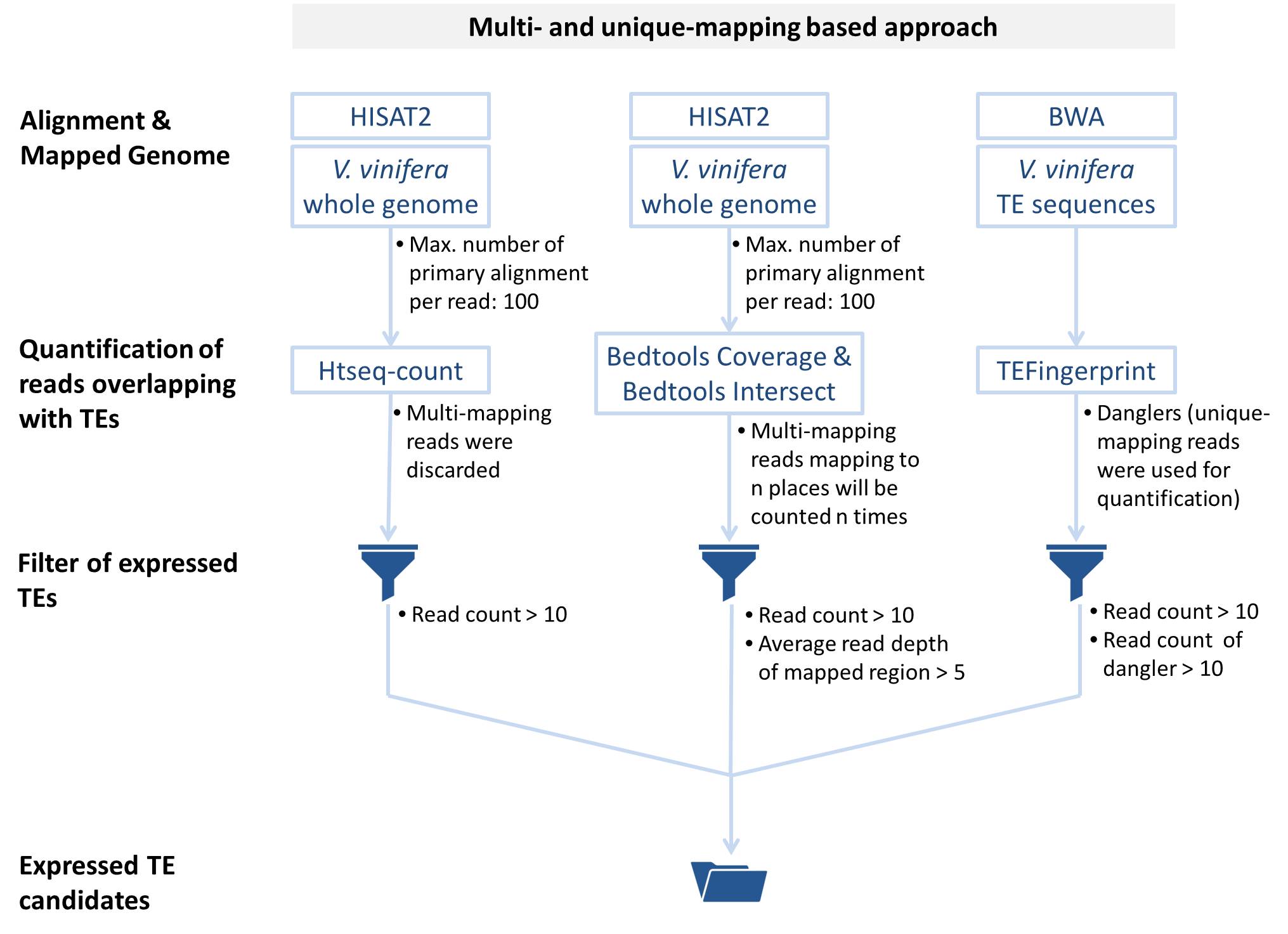
# RNAseq analysis pipeline for the identification of expressed TE candidates



1.

10.

9.

8.

7.

6.

4-5.

3.

2.

Contents

[RNAseq analysis pipeline for the identification of expressed TE candidates 2](#_Toc532462995)

[1. Alignment using HISAT2 (sub-pipeline 1 & 2) 2](#_Toc532462996)

[2. Quantification of reads mapping to TEs using htseq-count (sub-pipeline 1) 2](#_Toc532462997)

[3. Filtration of expressed TEs (sub-pipeline 1) 3](#_Toc532462998)

[4. Quantification of reads mapping to TEs using bedtools (sub-pipeline 2) 4](#_Toc532462999)

[5. Calculation of average read depth of mapped region using bedtools (sub-pipeline 2) 5](#_Toc532463000)

[6. Filtration of expressed TEs (sub-pipeline 2) 7](#_Toc532463001)

[7. Alignment using bwa (sub-pipeline 3) 9](#_Toc532463002)

[8. Capturing TE-related danglers and quantifying their read count using TEFingerprint (sub-pipeline 3) 9](#_Toc532463003)

[9. Filtration of expressed TEs (sub-pipeline 3) 10](#_Toc532463004)

[10. Merge of expressed TE candidates from the 3 sub-pipelines 11](#_Toc532463005)

# RNAseq analysis pipeline for the identification of expressed TE candidates

## Alignment using HISAT2 (sub-pipeline 1 & 2)

#!/bin/bash

#unzip fasta.gz files

**for** f **in** **\*.**gz**;** **do**

STEM**=$(basename "${f}" .gz)**

**gunzip** -c "${f}" **>** "${STEM}"

**done**

#concatenate files for each library

**for** i **in** **{**01**..**39**};** **do**

**cat** C9F4BANXX-1959-${i}**-**21**-**01\_**\***\_R1.fastq **>** ${i}\_R1.fastq

**cat** C9F4BANXX-1959-${i}**-**21**-**01\_**\***\_R2.fastq **>** ${i}\_R2.fastq**;**

**done**

#remove adapters by fastq-mcf and check quality of cleaned-up reads

**for** i **in** **{**01**..**39**};** **do**

fastq-mcf -f \

-o ${i}\_clean\_R1.fastq \

-o ${i}\_clean\_R2.fastq \

-l 50 **-**q 15 -t 0**.**1 -C 1000000 \

truseq\_adapters.fasta \

${i}\_R1.fastq \

${i}\_R2.fastq

fastqc ${i}\_clean\_**\*.**fastq -o QC.dir**;**

**done**

#build Hisat rRNA indices

hisat2-build 'AllVv\_tRNA\_rRNA.fasta' AllVv\_tRNA\_rRNA\_hisat2

#build Hisat genome indices

hisat2-build 'Vitis\_genome.fasta' Vitis\_genome\_hisat2

#align reads to tRNA and rRNA, and then filter for unmapped pairs

**for** i **in** **{**01**..**39**};** **do**

hisat2 -x **./**AllVv\_tRNA\_rRNA\_hisat2 \

**-**1 ${i}\_clean\_R1.fastq \

**-**2 ${i}\_clean\_R2.fastq \

-S ${i}\_ncRNA.sam

samtools view **-**Su -f 12 ${i}\_ncRNA.sam **>** ${i}\_ncRNA.bam

bedtools bamtofastq **-**i ${i}\_ncRNA.bam \

**-**fq ${i}\_pairs\_R1.fastq \

**-**fq2 ${i}\_pairs\_R2.fastq**;**

**done**

#map unmapped reads to Vitis vinifera genome

**for** i **in** **{**01**..** 39**};** **do**

hisat2 **--**rna-strandness RF \

**--**dta -k 100 \

-x Vitis\_genome\_hisat2 \

**-**1 ${i}\_pairs\_R1.fastq \

**-**2 ${i}\_pairs\_R2.fastq \

-S ${i}\_k100Mm.sam

samtools view **-**Su ${i}\_k100Mm.sam \

**|** samtools sort **-** ${i}\_k100Mm

samtools flagstat ${i}\_k100Mm.bam

samtools index ${i}\_k100Mm.bam**;**

**done**

#sort bam file by name

**for** i **in** **{**01**..**39**};** **do**

samtools sort -n **-**m 4G ${i}\_k100Mm.bam ${i}\_k100MmNamesorted.bam**;**

**done**

## Quantification of reads mapping to TEs using htseq-count (sub-pipeline 1)

#htseq-count --stranded=reverse

#for TEs

**for** i **in** **{**01**..**39**};** **do**

htseq-count -f bam \

-t exon \

**-**i transcript\_id \

-s reverse \

**-**m intersection-nonempty \

${i}\_k100MmNamesorted.bam \

AllTEsExpanded\_curated\_V2.gtf **>** ${i}\_k100Mm\_AllSenseTEm\_counttable\_curated.txt**;**

**done**

#for genes

#use version 2 gene annotation and only take the first isoform for each gene

gffread V2.1.gff3 -T -o V2.1.gtf

**grep** **-**E '\.1\b' V2.1.gtf **|** **grep** 'exon' **-** **>** V2.1\_iso1\_exon.gtf

**for** i **in** **{**01**..**39**};** **do**

htseq-count -f bam \

-t exon \

**-**i transcript\_id \

-s reverse \

**-**m intersection-nonempty \

${i}\_k100MmNamesorted.bam \

V2.1\_iso1\_exon.gtf **>** ${i}\_k100Mm\_AllSenseGene\_counttable\_V2.txt**;**

**done**

## Filtration of expressed TEs (sub-pipeline 1)

#R code

#select expr. TEs from htseq-count data

#htseq-count reports “mapped read pairs” when analysing pair-end sequencing data.

#the threshold of the pipeline is set to 10 reads, i.e. at least more than 5 read pairs

RE **=** read.table**(**'k100Mm\_AllSenseTEm\_curatedGTFcounttable.txt', header**=**T, row.names**=**1**)**

RE**=**RE**[!**grepl**(**"\_\_", row.names**(**RE**))**,**]**

RE2**=**RE**[**,1**:**3**]** #ctrl samples (T=0) are from column 1 to column 3

RE2**$**mean**=**rowMeans**(**RE2**[**,1**:**3**])**

RE2**=**subset**(**RE2, RE2**$**mean**>**5**)**

RE2**$**id**=**rownames**(**RE2**)**

RE2**=**RE2**[**,c**(**5,1**:**4**)]**

write.table**(**RE2, "k100Mm\_AllSenseTEm\_curatedGTF\_exprCtrl05.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

RE2**=**RE**[**,4**:**15**]** #mock samples are from column 4 to column 15

RE2**$**meanM01**=**rowMeans**(**RE2**[**,1**:**3**])**

RE2**$**meanM03**=**rowMeans**(**RE2**[**,4**:**6**])**

RE2**$**meanM06**=**rowMeans**(**RE2**[**,7**:**9**])**

RE2**$**meanM12**=**rowMeans**(**RE2**[**,10**:**12**])**

min\_read **=** 5

RE2 **<-** RE2**[**apply**(**RE2**[**,13**:**16**]**,1,**function(**x**){**max**(**x**)})** **>** min\_read,**]**

RE2**$**id**=**rownames**(**RE2**)**

RE2**=**RE2**[**,c**(**17,1**:**16**)]**

write.table**(**RE2, "k100Mm\_AllSenseTEm\_curatedGTF\_exprMock05.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

RE2**=**RE**[**,16**:**27**]** #yeast samples are from column 16 to column 27

RE2**$**meanY01**=**rowMeans**(**RE2**[**,1**:**3**])**

RE2**$**meanY03**=**rowMeans**(**RE2**[**,4**:**6**])**

RE2**$**meanY06**=**rowMeans**(**RE2**[**,7**:**9**])**

RE2**$**meanY12**=**rowMeans**(**RE2**[**,10**:**12**])**

min\_read **=** 5

RE2 **<-** RE2**[**apply**(**RE2**[**,13**:**16**]**,1,**function(**x**){**max**(**x**)})** **>** min\_read,**]**

RE2**$**id**=**rownames**(**RE2**)**

RE2**=**RE2**[**,c**(**17,1**:**16**)]**

write.table**(**RE2, "k100Mm\_AllSenseTEm\_curatedGTF\_exprYeast05.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

RE2**=**RE**[**,28**:**39**]** #botrytis samples are from column 28 to column 39

RE2**$**meanB01**=**rowMeans**(**RE2**[**,1**:**3**])**

RE2**$**meanB03**=**rowMeans**(**RE2**[**,4**:**6**])**

RE2**$**meanB06**=**rowMeans**(**RE2**[**,7**:**9**])**

RE2**$**meanB12**=**rowMeans**(**RE2**[**,10**:**12**])**

min\_read **=** 5

RE2 **<-** RE2**[**apply**(**RE2**[**,13**:**16**]**,1,**function(**x**){**max**(**x**)})** **>** min\_read,**]**

RE2**$**id**=**rownames**(**RE2**)**

RE2**=**RE2**[**,c**(**17,1**:**16**)]**

write.table**(**RE2, "k100Mm\_AllSenseTEm\_curatedGTF\_exprBotrytis05.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

#and then, use a text editor to do the followings:

#replace "\_ClassI;" and "\_ClassII;" to "\_" ;

#replace ":" to "\_" ;

#processed files are as follows:

#k100Mm\_AllSenseTEm\_curatedGTF\_exprCtrl05.txt -> k100Mm\_AllSenseTEm\_curatedGTF\_exprCtrl05\_V4.txt

#k100Mm\_AllSenseTEm\_curatedGTF\_exprMock05.txt -> k100Mm\_AllSenseTEm\_curatedGTF\_exprMock05\_V4.txt

#k100Mm\_AllSenseTEm\_curatedGTF\_exprYeast05.txt -> k100Mm\_AllSenseTEm\_curatedGTF\_exprYeast05\_V4.txt

#k100Mm\_AllSenseTEm\_curatedGTF\_exprBotrytis05.txt -> k100Mm\_AllSenseTEm\_curatedGTF\_exprBotrytis05\_V4.txt

## Quantification of reads mapping to TEs using bedtools (sub-pipeline 2)

#separate forward and reverse TEs into different gtf file

#R code

df**=**read.table**(**"AllTEsExpanded\_curated\_V2.gtf", header**=**F, sep**=**"\t"**)**

gtf\_foward**=**subset**(**df, df**$**V7**==**"+"**)**

gtf\_reverse**=**subset**(**df, df**$**V7**==**"-"**)**

write.table**(**gtf\_foward, "AllTEsExpanded\_curated\_Forward\_V2.gtf", col.names**=**F, row.names**=**F, sep**=**"\t", quote**=**F**)**

write.table**(**gtf\_reverse, "AllTEsExpanded\_curated\_Reverse\_V2.gtf", col.names**=**F, row.names**=**F, sep**=**"\t", quote**=**F**)**

#!/bin/bash

#capture F1R2 read pairs

**for** i **in** **{**01**..**39**};** **do**

bamtools filter **-**script filter\_F1R2.txt **-**in ${i}\_k100Mm.bam **-**out ${i}\_k100Mm\_F1R2.bam

samtools index ${i}\_k100Mm\_F1R2.bam**;**

**done**

#capture F2R1 read pairs

**for** i **in** **{**01**..**39**};** **do**

bamtools filter **-**script filter\_F2R1.txt **-**in ${i}\_k100Mm.bam **-**out ${i}\_k100Mm\_F2R1.bam

samtools index ${i}\_k100Mm\_F2R1.bam**;**

**done**

#script of “filter\_F1R2.txt” (JSON syntax)

**{**

"filters" **:** **[**

**{**"id" **:** "FirstinPair"**,**

"isFirstMate" **:** "true"

**},**

**{**"id" **:** "F"**,**

"isReverseStrand" **:** "false"

**},**

**{**"id" **:** "R"**,**

"isReverseStrand" **:** "true"

**},**

**{**"id" **:** "SecondinPair"**,**

"isFirstMate" **:** "false"

**},**

**{**"id" **:** "P"**,**

"isPaired" **:** "true"

**}**

**],**

"rule"**:** "(P & FirstinPair & F) | (P & SecondinPair & R)"

**}**

#script of “filter\_F2R1.txt” (JSON syntax)

**{**

"filters" **:** **[**

**{**"id" **:** "FirstinPair"**,**

"isFirstMate" **:** "true"

**},**

**{**"id" **:** "F"**,**

"isReverseStrand" **:** "false"

**},**

**{**"id" **:** "R"**,**

"isReverseStrand" **:** "true"

**},**

**{**"id" **:** "SecondinPair"**,**

"isFirstMate" **:** "false"

**},**

**{**"id" **:** "P"**,**

"isPaired" **:** "true"

**}**

**],**

"rule"**:** "( P & FirstinPair & R) | ( P & SecondinPair & F)"

**}**

#!/bin/bash

#count reads mapping to TEs in sense direction using “bedtools coverage”:

#for TEs in forward orientation, use F2R1 read pairs;

#for TEs in reverse orientation, use F1R2 read pairs.

**for** i **in** **{**01**..**39**};** **do**

bedtools bamtobed **-**split **-**i ${i}\_k100Mm\_F2R1.bam **>** ${i}\_k100Mm\_F2R1\_split.bed

bedtools bamtobed **-**split **-**i ${i}\_k100Mm\_F1R2.bam **>** ${i}\_k100Mm\_F1R2\_split.bed

bedtools coverage -a AllTEsExpanded\_curated\_Forward\_V2.gtf \

-b ${i}\_k100Mm\_F2R1\_split.bed \

**>** ${i}\_BedCov\_forwardTE\_sense.txt

bedtools coverage -a AllTEsExpanded\_curated\_Reverse\_V2.gtf \

-b ${i}\_k100Mm\_F1R2\_split.bed \

**>** ${i}\_BedCov\_reversedTE\_sense.txt**;**

**Done**

**for** i **in** **{**01**..**39**};** **do**

**cat** ${i}\_BedCov\_forwardTE\_sense.txt ${i}\_BedCov\_reversedTE\_sense.txt \

**|** **sort** **-**k1 \

**>** ${i}\_BedCov\_TE\_sense.txt**;**

**done**

## Calculation of average read depth of mapped region using bedtools (sub-pipeline 2)

#average read depth = ( Sum of MapReadBase )/( MapTEbase )

#Sum of MapReadBase = sum of mapped bases of the TE-mapping reads

#MapTEbase = sum of the mapped bases of the corresponding TEs

#!/bin/bash

#calculate MapReadBase of each read overlapping with each TE using “bedtools intersect”

**for** i **in** **{**01**..**39**};** **do**

bedtools intersect -a AllTEsExpanded\_curated\_Forward\_V2.gtf \

-b ${i}\_k100Mm\_F2R1\_split.bed \

**-**wo **>** ${i}\_BedIntersect\_forwardTE\_sense.txt

bedtools intersect -a AllTEsExpanded\_curated\_Reverse\_V2.gtf \

-b ${i}\_k100Mm\_F1R2\_split.bed \

**-**wo **>** ${i}\_BedIntersect\_reversedTE\_sense.txt**;**

**Done**

**for** i **in** **{**01**..**39**};** **do**

**cat** ${i}\_BedIntersect\_forwardTE\_sense.txt ${i}\_BedIntersect\_reversedTE\_sense.txt \

**|** **sort** **-**k1 **>** ${i}\_BedIntersect\_TE\_sense.txt**;**

**Done**

#establish the lists of TEs having sense reads and TEs not having sense reads

**awk** '{gsub("\"", "");print}' **$refdir/**AllTEsExpanded\_curated\_V2.gtf **>** temp.gtf

**for** i **in** **{**01**..**39**};** **do**

**awk** **-**F"\t" '{print $9}' ${i}\_BedIntersect\_TE\_sense.txt \

**|** **uniq** **>** ${i}\_TEhaveSenseRead.txt

**awk** **-**F'\t' 'NR==FNR {c[$1]++;next}; c[$9]==0' ${i}\_TEhaveSenseRead.txt temp.gtf \

**|** **awk** **-**F"\t" '{print}' **>** ${i}\_TEhaveNoSenseRead.txt**;**

**done**

**rm** -f temp.gtf

#calculate “Sum of MapReadBase” for each TEs having sense reads

#required information in the file “${i}\_BedIntersect\_TE\_sense.txt”:

#MapReadBase (number of mapped bases of each TE-mapping read): the last column

**for** i **in** **{**01**..**39**};** **do** **touch** ${i}temp01.txt**;** **done**

**for** i **in** **{**01**..**39**};** **do**

**cat** ${i}\_TEhaveSenseRead.txt **|** **while** **read** line**;** **do**

**grep** "$line" ${i}\_BedIntersect\_TE\_sense.txt **>** ${i}temp02.txt

**awk** '{sum+=$NF}END{print sum}' ${i}temp02.txt **>>** ${i}temp01.txt**;** **done;**

**done**

# calculate “average read depth”

#required information:

#(1) TE id in the file “${i}\_BedCov\_TE\_sense.txt”: 9th column

#(2) TE id in the file “${i}\_OverlapBasePair\_Read.txt”: 1st column

#(3) Sum of MapReadBase: will be in the 14th column of “${i}temp03.txt”

#(4) MapTEbase: will be in the 11th column of “${i}temp03.txt”

**for** i **in** **{**01**..**39**};** **do**

**paste** -d "\t" ${i}\_TEhaveSenseRead.txt ${i}temp01.txt **>** ${i}\_OverlapBasePair\_Read.txt

**join** **-**1 9 **-**2 1 -t $'\t' \

**<(sort** **-**k9 ${i}\_BedCov\_TE\_sense.txt**)** \

**<(sort** **-**k1 ${i}\_OverlapBasePair\_Read.txt**)** \

**|** **awk** **-**F'\t' '{OFS="\t"; print $2,$3,$4,$5,$6,$7,$8,$9,$1,$10,$11,$12,$13,$14}' **-** \

**>** ${i}temp03.txt

**awk** **-**F'\t' '{depth=$14/$11; print depth}' ${i}temp03.txt \

**|** **paste** ${i}temp03.txt **-** \

**>** ${i}\_BedCov\_TEhaveSenseRead.txt

**awk** **-**F"\t" '{OFS="\t"; TElength=$5-$4+1; print $1,$2,$3,$4,$5,$6,$7,$8,$9,0,0,

TElength,0,0,0}' ${i}\_TEhaveNoSenseRead.txt \

**>** ${i}\_BedCov\_TEhaveNoSenseRead.txt

**cat** ${i}\_BedCov\_TEhaveSenseRead.txt ${i}\_BedCov\_TEhaveNoSenseRead.txt \

**|** **sort** **-**k1 \

**>** ${i}\_BedCov\_OverlapBasePair\_senseRead.txt**;**

**done**

**rm** -f ${i}temp**\*.**txt

**rm** -f ${i}\_TEhave**\*.**bed**;**

#content of “${i}\_BedCov\_OverlapBasePair\_senseRead.txt”:

#column 1: chr

#column 2: source

#column 3: feature

#column 4: start

#column 5: end

#column 6: score

#column 7: str (strand)

#column 8: dot

#column 9: TE\_id

#column 10: count (read count)

#column 11: coveredTEbp (MapTEbase, sum of the mapped bases of the TE)

#column 12: TElength (length of TE)

#column 13: breadthCov (breadth of coverage)

#column 14: readbpsum (Sum of MapReadBase)

#column 15: depth (average read depth)

#combined all sample

#R code

df01**=**read.table**(**"01\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df02**=**read.table**(**"02\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df03**=**read.table**(**"03\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df04**=**read.table**(**"04\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df05**=**read.table**(**"05\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df06**=**read.table**(**"06\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df07**=**read.table**(**"07\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df08**=**read.table**(**"08\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df09**=**read.table**(**"09\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df10**=**read.table**(**"10\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df11**=**read.table**(**"11\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df12**=**read.table**(**"12\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df13**=**read.table**(**"13\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df14**=**read.table**(**"14\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df15**=**read.table**(**"15\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df16**=**read.table**(**"16\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df17**=**read.table**(**"17\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df18**=**read.table**(**"18\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df19**=**read.table**(**"19\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df20**=**read.table**(**"20\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df21**=**read.table**(**"21\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df22**=**read.table**(**"22\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df23**=**read.table**(**"23\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df24**=**read.table**(**"24\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df25**=**read.table**(**"25\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df26**=**read.table**(**"26\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df27**=**read.table**(**"27\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df28**=**read.table**(**"28\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df29**=**read.table**(**"29\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df30**=**read.table**(**"30\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df31**=**read.table**(**"31\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df32**=**read.table**(**"32\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df33**=**read.table**(**"33\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df34**=**read.table**(**"34\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df35**=**read.table**(**"35\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df36**=**read.table**(**"36\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df37**=**read.table**(**"37\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df38**=**read.table**(**"38\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df39**=**read.table**(**"39\_BedCov\_OverlapBasePair\_senseRead.txt", header**=**F, sep**=**"\t"**)**

df01**=**df01**[**order**(**df01**$**V9**)**,**]**

df02**=**df02**[**order**(**df02**$**V9**)**,**]**

df03**=**df03**[**order**(**df03**$**V9**)**,**]**

df04**=**df04**[**order**(**df04**$**V9**)**,**]**

df05**=**df05**[**order**(**df05**$**V9**)**,**]**

df06**=**df06**[**order**(**df06**$**V9**)**,**]**

df07**=**df07**[**order**(**df07**$**V9**)**,**]**

df08**=**df08**[**order**(**df08**$**V9**)**,**]**

df09**=**df09**[**order**(**df09**$**V9**)**,**]**

df10**=**df10**[**order**(**df10**$**V9**)**,**]**

df11**=**df11**[**order**(**df11**$**V9**)**,**]**

df12**=**df12**[**order**(**df12**$**V9**)**,**]**

df13**=**df13**[**order**(**df13**$**V9**)**,**]**

df14**=**df14**[**order**(**df14**$**V9**)**,**]**

df15**=**df15**[**order**(**df15**$**V9**)**,**]**

df16**=**df16**[**order**(**df16**$**V9**)**,**]**

df17**=**df17**[**order**(**df17**$**V9**)**,**]**

df18**=**df18**[**order**(**df18**$**V9**)**,**]**

df19**=**df19**[**order**(**df19**$**V9**)**,**]**

df20**=**df20**[**order**(**df20**$**V9**)**,**]**

df21**=**df21**[**order**(**df21**$**V9**)**,**]**

df22**=**df22**[**order**(**df22**$**V9**)**,**]**

df23**=**df23**[**order**(**df23**$**V9**)**,**]**

df24**=**df24**[**order**(**df24**$**V9**)**,**]**

df25**=**df25**[**order**(**df25**$**V9**)**,**]**

df26**=**df26**[**order**(**df26**$**V9**)**,**]**

df27**=**df27**[**order**(**df27**$**V9**)**,**]**

df28**=**df28**[**order**(**df28**$**V9**)**,**]**

df29**=**df29**[**order**(**df29**$**V9**)**,**]**

df30**=**df30**[**order**(**df30**$**V9**)**,**]**

df31**=**df31**[**order**(**df31**$**V9**)**,**]**

df32**=**df32**[**order**(**df32**$**V9**)**,**]**

df33**=**df33**[**order**(**df33**$**V9**)**,**]**

df34**=**df34**[**order**(**df34**$**V9**)**,**]**

df35**=**df35**[**order**(**df35**$**V9**)**,**]**

df36**=**df36**[**order**(**df36**$**V9**)**,**]**

df37**=**df37**[**order**(**df37**$**V9**)**,**]**

df38**=**df38**[**order**(**df38**$**V9**)**,**]**

df39**=**df39**[**order**(**df39**$**V9**)**,**]**

data1**=**cbind**(**df01**[**,c**(**1,4,5,9,6,7**)]**,df01**[**,10**]**,df02**[**,10**]**,df03**[**,10**]**,df04**[**,10**]**,df05**[**,10**]**, df06**[**,10**]**,df07**[**,10**]**,df08**[**,10**]**,df09**[**,10**]**,df10**[**,10**]**,df11**[**,10**]**,df12**[**,10**]**,df13**[**,10**]**, df14**[**,10**]**,df15**[**,10**]**,df16**[**,10**]**,df17**[**,10**]**,df18**[**,10**]**,df19**[**,10**]**,df20**[**,10**]**,df21**[**,10**]**, df22**[**,10**]**,df23**[**,10**]**,df24**[**,10**]**,df25**[**,10**]**,df26**[**,10**]**,df27**[**,10**]**,df28**[**,10**]**,df29**[**,10**]**, df30**[**,10**]**,df31**[**,10**]**,df32**[**,10**]**,df33**[**,10**]**,df34**[**,10**]**,df35**[**,10**]**,df36**[**,10**]**,df37**[**,10**]**, df38**[**,10**]**,df39**[**,10**])**

data2**=**cbind**(**df01**[**,c**(**1,4,5,9,6,7**)]**,df01**[**,13**]**,df02**[**,13**]**,df03**[**,13**]**,df04**[**,13**]**,df05**[**,13**]**, df06**[**,13**]**,df07**[**,13**]**,df08**[**,13**]**,df09**[**,13**]**,df10**[**,13**]**,df11**[**,13**]**,df12**[**,13**]**,df13**[**,13**]**, df14**[**,13**]**,df15**[**,13**]**,df16**[**,13**]**,df17**[**,13**]**,df18**[**,13**]**,df19**[**,13**]**,df20**[**,13**]**,df21**[**,13**]**, df22**[**,13**]**,df23**[**,13**]**,df24**[**,13**]**,df25**[**,13**]**,df26**[**,13**]**,df27**[**,13**]**,df28**[**,13**]**,df29**[**,13**]**, df30**[**,13**]**,df31**[**,13**]**,df32**[**,13**]**,df33**[**,13**]**,df34**[**,13**]**,df35**[**,13**]**,df36**[**,13**]**,df37**[**,13**]**, df38**[**,13**]**,df39**[**,13**])**

data3**=**cbind**(**df01**[**,c**(**1,4,5,9,6,7**)]**,df01**[**,15**]**,df02**[**,15**]**,df03**[**,15**]**,df04**[**,15**]**,df05**[**,15**]**, df06**[**,15**]**,df07**[**,15**]**,df08**[**,15**]**,df09**[**,15**]**,df10**[**,15**]**,df11**[**,15**]**,df12**[**,15**]**,df13**[**,15**]**, df14**[**,15**]**,df15**[**,15**]**,df16**[**,15**]**,df17**[**,15**]**,df18**[**,15**]**,df19**[**,15**]**,df20**[**,15**]**,df21**[**,15**]**, df22**[**,15**]**,df23**[**,15**]**,df24**[**,15**]**,df25**[**,15**]**,df26**[**,15**]**,df27**[**,15**]**,df28**[**,15**]**,df29**[**,15**]**, df30**[**,15**]**,df31**[**,15**]**,df32**[**,15**]**,df33**[**,15**]**,df34**[**,15**]**,df35**[**,15**]**,df36**[**,15**]**,df37**[**,15**]**, df38**[**,15**]**,df39**[**,15**])**

write.table**(**data1, "tempdata1.txt", col.names**=**F, row.names**=**F, sep**=**"\t", quote**=**F**)**

write.table**(**data2, "tempdata2.txt", col.names**=**F, row.names**=**F, sep**=**"\t", quote**=**F**)**

write.table**(**data3, "tempdata3.txt", col.names**=**F, row.names**=**F, sep**=**"\t", quote**=**F**)**

#then use text editor to replace ":" "\_ClassI;" and "\_ClassII;" to "\_" ; replace ";" to nothing; files to be modified are “tempdata1.txt”, “tempdata2.txt” and “tempdata3.txt”

data01**=**read.table**(**textConnection**(**gsub**(**" ", "\t", readLines**(**"tempdata1.txt"**))))**

data02**=**read.table**(**textConnection**(**gsub**(**" ", "\t", readLines**(**"tempdata2.txt"**))))**

data03**=**read.table**(**textConnection**(**gsub**(**" ", "\t", readLines**(**"tempdata3.txt"**))))**

data01**=**data01**[**,c**(**1**:**3, 7, 12**:**52**)]**

data02**=**data02**[**,c**(**1**:**3, 7, 12**:**52**)]**

data03**=**data03**[**,c**(**1**:**3, 7, 12**:**52**)]**

name**=**c**(**"chr","start","end","TEm","score","str","Ctrl\_a","Ctrl\_b","Ctrl\_c","Mock01\_a", "Mock01\_b","Mock01\_c","Mock03\_a","Mock03\_b","Mock03\_c","Mock06\_a","Mock06\_b", "Mock06\_c","Mock12\_a","Mock12\_b","Mock12\_c","Yeast01\_a","Yeast01\_b","Yeast01\_c", "Yeast03\_a","Yeast03\_b","Yeast03\_c","Yeast06\_a","Yeast06\_b","Yeast06\_c", "Yeast12\_a","Yeast12\_b","Yeast12\_c","Botrytis01\_a","Botrytis01\_b","Botrytis01\_c", "Botrytis03\_a","Botrytis03\_b","Botrytis03\_c","Botrytis06\_a","Botrytis06\_b", "Botrytis06\_c","Botrytis12\_a","Botrytis12\_b","Botrytis12\_c"**)**

colnames**(**data01**)=**name

colnames**(**data02**)=**name

colnames**(**data03**)=**name

write.table**(**data01, "BedCov\_OverlapBP\_sense\_count.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

write.table**(**data02, "BedCov\_OverlapBP\_sense\_breadthCov.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

write.table**(**data03, "BedCov\_OverlapBP\_sense\_depth.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

## Filtration of expressed TEs (sub-pipeline 2)

#R code

count**=**read.table**(**"BedCov\_OverlapBP\_sense\_count.txt", header**=**T, sep**=**"\t"**)**

depth**=**read.table**(**"BedCov\_OverlapBP\_sense\_depth.txt", header**=**T, sep**=**"\t"**)**

#ctrl

df**=**count

df**$**Cmean00**=**rowMeans**(**df**[**,7**:**9**])**

data1 **<-** subset**(**df, df**$**Cmean00 **>** 10**)**

df**=**depth

df**$**Cmean00**=**rowMeans**(**df**[**,7**:**9**])**

data2 **<-**subset**(**df, df**$**Cmean00 **>** 5**)**

ctrl**=**merge**(**data1, data2, by**=**"TEm"**)**

ctrl**=**ctrl**[**,**-**c**(**10**:**45,47**:**51,55**:**90**)]**

colnames**(**ctrl**)=**c**(**"TEm","chr","start","end","score","str","count\_Ctrl\_a","count\_Ctrl\_b", "count\_Ctrl\_c","count\_Cmean00","depth\_Ctrl\_a","depth\_Ctrl\_b","depth\_Ctrl\_c", "depth\_Cmean00"**)**

write.table**(**ctrl, "BedCov\_OverlapBP\_PassC.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

#mock

df**=**count

df**=**df**[**,c**(**1**:**6,10**:**21**)]**

df**$**Mmean01**=**rowMeans**(**df**[**,7**:**9**])**

df**$**Mmean03**=**rowMeans**(**df**[**,10**:**12**])**

df**$**Mmean06**=**rowMeans**(**df**[**,13**:**15**])**

df**$**Mmean12**=**rowMeans**(**df**[**,16**:**18**])**

min\_read **=** 10

data1 **<-** df**[**apply**(**df**[**,19**:**22**]**,1,**function(**x**){**max**(**x**)})** **>** min\_read,**]**

df**=**depth

df**=**df**[**,c**(**1**:**6,10**:**21**)]**

df**$**Mmean01**=**rowMeans**(**df**[**,7**:**9**])**

df**$**Mmean03**=**rowMeans**(**df**[**,10**:**12**])**

df**$**Mmean06**=**rowMeans**(**df**[**,13**:**15**])**

df**$**Mmean12**=**rowMeans**(**df**[**,16**:**18**])**

min\_depth **=** 5

data2 **<-** df**[**apply**(**df**[**,19**:**22**]**,1,**function(**x**){**max**(**x**)})** **>** min\_depth,**]**

data**=**merge**(**data1, data2, by**=**"TEm"**)**

data**=**data**[**,**-**c**(**23**:**27**)]**

colnames**(**data**)=**c**(**"TEm","chr","start","end","score","str","count\_Mock01\_a", "count\_Mock01\_b","count\_Mock01\_c","count\_Mock03\_a","count\_Mock03\_b", "count\_Mock03\_c","count\_Mock06\_a","count\_Mock06\_b","count\_Mock06\_c", "count\_Mock12\_a","count\_Mock12\_b","count\_Mock12\_c","count\_Mmean01", "count\_Mmean03","count\_Mmean06","count\_Mmean12","depth\_Mock01\_a", "depth\_Mock01\_b","depth\_Mock01\_c","depth\_Mock03\_a","depth\_Mock03\_b", "depth\_Mock03\_c","depth\_Mock06\_a","depth\_Mock06\_b","depth\_Mock06\_c", "depth\_Mock12\_a","depth\_Mock12\_b","depth\_Mock12\_c","depth\_Mmean01", "depth\_Mmean03","depth\_Mmean06","depth\_Mmean12"**)**

write.table**(**data, "BedCov\_OverlapBP\_PassM.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

#yeast

df**=**count

df**=**df**[**,c**(**1**:**6,22**:**33**)]**

df**$**Mmean01**=**rowMeans**(**df**[**,7**:**9**])**

df**$**Mmean03**=**rowMeans**(**df**[**,10**:**12**])**

df**$**Mmean06**=**rowMeans**(**df**[**,13**:**15**])**

df**$**Mmean12**=**rowMeans**(**df**[**,16**:**18**])**

min\_read **=** 10

data1 **<-** df**[**apply**(**df**[**,19**:**22**]**,1,**function(**x**){**max**(**x**)})** **>** min\_read,**]**

df**=**depth

df**=**df**[**,c**(**1**:**6,22**:**33**)]**

df**$**Mmean01**=**rowMeans**(**df**[**,7**:**9**])**

df**$**Mmean03**=**rowMeans**(**df**[**,10**:**12**])**

df**$**Mmean06**=**rowMeans**(**df**[**,13**:**15**])**

df**$**Mmean12**=**rowMeans**(**df**[**,16**:**18**])**

min\_depth **=** 5

data2 **<-** df**[**apply**(**df**[**,19**:**22**]**,1,**function(**x**){**max**(**x**)})** **>** min\_depth,**]**

data**=**merge**(**data1, data2, by**=**"TEm"**)**

data**=**data**[**,**-**c**(**23**:**27**)]**

colnames**(**data**)=**c**(**"TEm","chr","start","end","score","str","count\_Yeast01\_a", "count\_Yeast01\_b","count\_Yeast01\_c","count\_Yeast03\_a","count\_Yeast03\_b", "count\_Yeast03\_c","count\_Yeast06\_a","count\_Yeast06\_b","count\_Yeast06\_c", "count\_Yeast12\_a","count\_Yeast12\_b","count\_Yeast12\_c","count\_Ymean01", "count\_Ymean03","count\_Ymean06","count\_Ymean12","depth\_Yeast01\_a", "depth\_Yeast01\_b","depth\_Yeast01\_c","depth\_Yeast03\_a","depth\_Yeast03\_b", "depth\_Yeast03\_c","depth\_Yeast06\_a","depth\_Yeast06\_b","depth\_Yeast06\_c", "depth\_Yeast12\_a","depth\_Yeast12\_b","depth\_Yeast12\_c","depth\_Ymean01", "depth\_Ymean03","depth\_Ymean06","depth\_Ymean12"**)**

write.table**(**data, "BedCov\_OverlapBP\_PassY.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

#botrytis

df**=**count

df**=**df**[**,c**(**1**:**6,34**:**45**)]**

df**$**Mmean01**=**rowMeans**(**df**[**,7**:**9**])**

df**$**Mmean03**=**rowMeans**(**df**[**,10**:**12**])**

df**$**Mmean06**=**rowMeans**(**df**[**,13**:**15**])**

df**$**Mmean12**=**rowMeans**(**df**[**,16**:**18**])**

min\_read **=** 10

data1 **<-** df**[**apply**(**df**[**,19**:**22**]**,1,**function(**x**){**max**(**x**)})** **>** min\_read,**]**

df**=**depth

df**=**df**[**,c**(**1**:**6,34**:**45**)]**

df**$**Mmean01**=**rowMeans**(**df**[**,7**:**9**])**

df**$**Mmean03**=**rowMeans**(**df**[**,10**:**12**])**

df**$**Mmean06**=**rowMeans**(**df**[**,13**:**15**])**

df**$**Mmean12**=**rowMeans**(**df**[**,16**:**18**])**

min\_depth **=** 5

data2 **<-** df**[**apply**(**df**[**,19**:**22**]**,1,**function(**x**){**max**(**x**)})** **>** min\_depth,**]**

data**=**merge**(**data1, data2, by**=**"TEm"**)**

data**=**data**[**,**-**c**(**23**:**27**)]**

colnames**(**data**)=**c**(**"TEm","chr","start","end","score","str","count\_Botrytis01\_a", "count\_Botrytis01\_b","count\_Botrytis01\_c","count\_Botrytis03\_a", "count\_Botrytis03\_b","count\_Botrytis03\_c","count\_Botrytis06\_a", "count\_Botrytis06\_b","count\_Botrytis06\_c","count\_Botrytis12\_a", "count\_Botrytis12\_b","count\_Botrytis12\_c","count\_Bmean01","count\_Bmean03", "count\_Bmean06","count\_Bmean12","depth\_Botrytis01\_a","depth\_Botrytis01\_b", "depth\_Botrytis01\_c","depth\_Botrytis03\_a","depth\_Botrytis03\_b", "depth\_Botrytis03\_c","depth\_Botrytis06\_a","depth\_Botrytis06\_b", "depth\_Botrytis06\_c","depth\_Botrytis12\_a","depth\_Botrytis12\_b", "depth\_Botrytis12\_c","depth\_Bmean01","depth\_Bmean03","depth\_Bmean06", "depth\_Bmean12"**)**

write.table**(**data, "BedCov\_OverlapBP\_PassB.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

## Alignment using bwa (sub-pipeline 3)

#!/bin/bash

#Build index

bwa index -a bwtsw Vitis\_genome.fasta

bwa index -a is AllTEsExpanded\_curated.fa

#Align read pairs against sequences of annotated TEs

**for** i **in** **{**01**..**39**};** **do**

bwa mem -t 8 **\**

AllTEsExpanded\_curated.fa **\**

${i}\_pairs\_R1.fastq **\**

${i}\_pairs\_R2.fastq **\**

**|** samtools view **-**Su **-** **\**

**|** samtools sort **-** -o ${i}\_TEmapped.bam**;**

**done**

#index the resulting bam file

**for** i **in** **{**01**..**39**};** **do**

samtools index **/**${i}\_TEmapped.bam**;**

**done**

## Capturing TE-related danglers and quantifying their read count using TEFingerprint (sub-pipeline 3)

#!/bin/bash

#tef preprocess (no clipped reads used)

mydir**=/**home**/**user

source activate tefingerprint

**export** PATH**=$HOME/**tools**/**TEFingerprint-master**/**applications**/:$PATH**

**cd** **/**home**/**user**/**tools**/**TEFingerprint-master**/**

**for** i **in** **{**01**..**39**};** **do**

tef preprocess **\**

**$mydir/**${i}\_TEmapped.bam **\**

**--**reference **$mydir/**Vitis\_genome.fasta **\**

**--**output **$mydir/**preprocess**/**${i}\_danglers.bam **\**

**--**exclude-tails **\**

**--**tempdir **$mydir/**temp **\**

**--**threads 8**;**

**done**

#generate unique TE family lists

**awk** **-**vOFS**=**'\t' '{print $1}' **$mydir/**TEfamilies\_named.lst \

**|** **sort** **|** **uniq** **>** **$mydir/**TEfamilies.txt

#tef compare

#threshold for TE-related dangler > 10 reads, so the minimum value is set to 11 (-m 11)

tef compare **$mydir/**preprocess**/**01\_danglers.bam **$mydir/**preprocess**/**02\_danglers.bam \

**$mydir/**preprocess**/**03\_danglers.bam **$mydir/**preprocess**/**04\_danglers.bam \

**$mydir/**preprocess**/**05\_danglers.bam **$mydir/**preprocess**/**06\_danglers.bam \

**$mydir/**preprocess**/**07\_danglers.bam **$mydir/**preprocess**/**08\_danglers.bam \

**$mydir/**preprocess**/**09\_danglers.bam **$mydir/**preprocess**/**10\_danglers.bam \

**$mydir/**preprocess**/**11\_danglers.bam **$mydir/**preprocess**/**12\_danglers.bam \

**$mydir/**preprocess**/**13\_danglers.bam **$mydir/**preprocess**/**14\_danglers.bam \

**$mydir/**preprocess**/**15\_danglers.bam **$mydir/**preprocess**/**16\_danglers.bam \

**$mydir/**preprocess**/**17\_danglers.bam **$mydir/**preprocess**/**18\_danglers.bam \

**$mydir/**preprocess**/**19\_danglers.bam **$mydir/**preprocess**/**20\_danglers.bam \

**$mydir/**preprocess**/**21\_danglers.bam **$mydir/**preprocess**/**22\_danglers.bam \

**$mydir/**preprocess**/**23\_danglers.bam **$mydir/**preprocess**/**24\_danglers.bam \

**$mydir/**preprocess**/**25\_danglers.bam **$mydir/**preprocess**/**26\_danglers.bam \

**$mydir/**preprocess**/**27\_danglers.bam **$mydir/**preprocess**/**28\_danglers.bam \

**$mydir/**preprocess**/**29\_danglers.bam **$mydir/**preprocess**/**30\_danglers.bam \

**$mydir/**preprocess**/**31\_danglers.bam **$mydir/**preprocess**/**32\_danglers.bam \

**$mydir/**preprocess**/**33\_danglers.bam **$mydir/**preprocess**/**34\_danglers.bam \

**$mydir/**preprocess**/**35\_danglers.bam **$mydir/**preprocess**/**36\_danglers.bam \

**$mydir/**preprocess**/**37\_danglers.bam **$mydir/**preprocess**/**38\_danglers.bam \

**$mydir/**preprocess**/**39\_danglers.bam **\**

-f **$(< $mydir/TEfamilies.txt)** **\**

**-**m 11 **\**

-e 600 **\**

**-**q 30 **\**

-t 8 **\**

**>** **$mydir/**compare**/**AllRep\_comparison.gff

#use "Excel" to open the file and define tab, comma, semicolon and "=" as separators, #and then save the file as tab-separated text file.

#capture TE-related danglers overlapping with annotated TEs using “bedtools intersect”

#!/bin/bash

**cd ~**

bedtools intersect -a **./**compare**/**AllRep\_comparison.gff \

-b AllTEsExpanded\_curated\_V3.gtf \

**-**wa \

**-**wb \

**>** AllRep\_comp\_intersect.gtf

#use "Excel" to open the output file and define tab, comma, semicolon, space and "=" as #separators, and then save it as tab-separated text file: AllRep\_comp\_intersect\_V4.gtf.

## Filtration of expressed TEs (sub-pipeline 3)

#R code

#select expressed TEs from TEFingerprint data

dangler**=**read.table**(**"AllRep\_comp\_intersect\_V4.txt", header**=**F, sep**=**"\t"**)**

sense**=**read.table**(**"BedCov\_OverlapBP\_sense\_count.txt", header**=**T, sep**=**"\t"**)**

#ctrl

dangler2**=**dangler**[**,c**(**146,54**:**56**)]**

dangler2**$**mean**=**rowMeans**(**dangler2**[**,2**:**4**])**

dangler2**=**subset**(**dangler2, dangler2**$**mean**>**10**)**

colnames**(**dangler2**)=**c**(**"id", "ctrl\_a", "ctrl\_b", "ctrl\_c", "mean"**)**

df**=**merge**(**dangler2, sense, by.x**=**"id", by.y**=**"TEm"**)**

df**$**countMean**=**rowMeans**(**df**[**,11**:**13**])**

df2**=**subset**(**df, df**$**countMean**>**10**)**

df2**=**df2**[**,1**:**5**]**

dftest**=**df2**[!**duplicated**(**df2**$**id**)**,**]**

write.table**(**df2, "AllRep\_comp\_intersect\_exprCtrl\_V4.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

#mock

dangler2**=**dangler**[**,c**(**146,57**:**68**)]**

dangler2**$**meanM01**=**rowMeans**(**dangler2**[**,2**:**4**])**

dangler2**$**meanM03**=**rowMeans**(**dangler2**[**,5**:**7**])**

dangler2**$**meanM06**=**rowMeans**(**dangler2**[**,8**:**10**])**

dangler2**$**meanM12**=**rowMeans**(**dangler2**[**,11**:**13**])**

min\_read **=** 10

dangler2 **<-** dangler2**[**apply**(**dangler2**[**,14**:**17**]**,1,**function(**x**){**max**(**x**)})** **>** min\_read,**]**

colnames**(**dangler2**)=**c**(**"id", "M01\_a", "M01\_b", "M01\_c","M03\_a", "M03\_b", "M03\_c","M06\_a",

"M06\_b", "M06\_c","M12\_a", "M12\_b", "M12\_c", "meanM01", "meanM03", "meanM06",

"meanM12"**)**

df**=**merge**(**dangler2, sense, by.x**=**"id", by.y**=**"TEm"**)**

df**$**countMean01**=**rowMeans**(**df**[**,26**:**28**])**

df**$**countMean03**=**rowMeans**(**df**[**,29**:**31**])**

df**$**countMean06**=**rowMeans**(**df**[**,32**:**34**])**

df**$**countMean12**=**rowMeans**(**df**[**,35**:**37**])**

min\_read **=** 10

df2 **<-** df**[**apply**(**df**[**,62**:**65**]**,1,**function(**x**){**max**(**x**)})** **>** min\_read,**]**

df2**=**df2**[**,1**:**17**]**

dftest**=**df2**[!**duplicated**(**df2**$**id**)**,**]**

write.table**(**df2, "AllRep\_comp\_intersect\_exprMock\_V4.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

#yeast

dangler2**=**dangler**[**,c**(**146,69**:**80**)]**

dangler2**$**meanY01**=**rowMeans**(**dangler2**[**,2**:**4**])**

dangler2**$**meanY03**=**rowMeans**(**dangler2**[**,5**:**7**])**

dangler2**$**meanY06**=**rowMeans**(**dangler2**[**,8**:**10**])**

dangler2**$**meanY12**=**rowMeans**(**dangler2**[**,11**:**13**])**

min\_read **=** 10

dangler2 **<-** dangler2**[**apply**(**dangler2**[**,14**:**17**]**,1,**function(**x**){**max**(**x**)})** **>** min\_read,**]**

colnames**(**dangler2**)=**c**(**"id", "Y01\_a", "Y01\_b", "Y01\_c","Y03\_a", "Y03\_b", "Y03\_c","Y06\_a",

"Y06\_b", "Y06\_c","Y12\_a", "Y12\_b", "Y12\_c", "meanY01", "meanY03", "meanY06",

"meanY12"**)**

df**=**merge**(**dangler2, sense, by.x**=**"id", by.y**=**"TEm"**)**

df**$**countMean01**=**rowMeans**(**df**[**,38**:**40**])**

df**$**countMean03**=**rowMeans**(**df**[**,41**:**43**])**

df**$**countMean06**=**rowMeans**(**df**[**,44**:**46**])**

df**$**countMean12**=**rowMeans**(**df**[**,47**:**49**])**

min\_read **=** 10

df2 **<-** df**[**apply**(**df**[**,62**:**65**]**,1,**function(**x**){**max**(**x**)})** **>** min\_read,**]**

df2**=**df2**[**,1**:**17**]**

dftest**=**df2**[!**duplicated**(**df2**$**id**)**,**]**

write.table**(**df2, "AllRep\_comp\_intersect\_exprYeast\_V4.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

#botrytis

dangler2**=**dangler**[**,c**(**146,81**:**92**)]**

dangler2**$**meanB01**=**rowMeans**(**dangler2**[**,2**:**4**])**

dangler2**$**meanB03**=**rowMeans**(**dangler2**[**,5**:**7**])**

dangler2**$**meanB06**=**rowMeans**(**dangler2**[**,8**:**10**])**

dangler2**$**meanB12**=**rowMeans**(**dangler2**[**,11**:**13**])**

min\_read **=** 10

dangler2 **<-** dangler2**[**apply**(**dangler2**[**,14**:**17**]**,1,**function(**x**){**max**(**x**)})** **>** min\_read,**]**

colnames**(**dangler2**)=**c**(**"id", "B01\_a", "B01\_b", "B01\_c","B03\_a", "B03\_b", "B03\_c","B06\_a",

"B06\_b", "B06\_c","B12\_a", "B12\_b", "B12\_c", "meanB01", "meanB03", "meanB06",

"meanB12"**)**

df**=**merge**(**dangler2, sense, by.x**=**"id", by.y**=**"TEm"**)**

df**$**countMean01**=**rowMeans**(**df**[**,50**:**52**])**

df**$**countMean03**=**rowMeans**(**df**[**,53**:**55**])**

df**$**countMean06**=**rowMeans**(**df**[**,56**:**58**])**

df**$**countMean12**=**rowMeans**(**df**[**,59**:**61**])**

min\_read **=** 10

df2 **<-** df**[**apply**(**df**[**,62**:**65**]**,1,**function(**x**){**max**(**x**)})** **>** min\_read,**]**

df2**=**df2**[**,1**:**17**]**

dftest**=**df2**[!**duplicated**(**df2**$**id**)**,**]**

write.table**(**df2, "AllRep\_comp\_intersect\_exprBotrytis\_V4.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

## Merge of expressed TE candidates from the 3 sub-pipelines

#R code

library**(**VennDiagram**)**

cal.overlap**=function** **(**x**)** **{**

**if** **(**1 **==** length**(**x**))** **{**

overlap **<-** x

**}**

**else** **if** **(**2 **==** length**(**x**))** **{**

A **<-** x**[[**1**]]**

B **<-** x**[[**2**]]**

nab **<-** intersect**(**A, B**)**

a1 **=** A**[!** A %in% nab**]**

a2 **=** B**[!** B %in% nab**]**

overlap **<-** list**(**a1 **=** a1, a2 **=** a2, a3 **=** nab**)**

**}**

**else** **if** **(**3 **==** length**(**x**))** **{**

A **<-** x**[[**1**]]**

B **<-** x**[[**2**]]**

C **<-** x**[[**3**]]**

nab **<-** intersect**(**A, B**)**

nbc **<-** intersect**(**B, C**)**

nac **<-** intersect**(**A, C**)**

nabc **<-** intersect**(**nab, C**)**

a5 **=** nabc

a2 **=** nab**[!** nab %in% a5**]**

a4 **=** nac**[-**which**(**nac %in% a5**)]**

a6 **=** nbc**[-**which**(**nbc %in% a5**)]**

a1 **=** A**[-**which**(**A %in% c**(**a2, a4, a5**))]**

a3 **=** B**[-**which**(**B %in% c**(**a2, a5, a6**))]**

a7 **=** C**[-**which**(**C %in% c**(**a4, a5, a6**))]**

overlap **<-** list**(**a5 **=** a5, a2 **=** a2, a4 **=** a4, a6 **=** a6, a1 **=** a1,

a3 **=** a3, a7 **=** a7**)**

**}**

**else** **if** **(**4 **==** length**(**x**))** **{**

A **<-** x**[[**1**]]**

B **<-** x**[[**2**]]**

C **<-** x**[[**3**]]**

D **<-** x**[[**4**]]**

nab **<-** intersect**(**A, B**)**

nbc **<-** intersect**(**B, C**)**

nac **<-** intersect**(**A, C**)**

nbd **<-** intersect**(**B, D**)**

nad **<-** intersect**(**A, D**)**

ncd **<-** intersect**(**C, D**)**

nabc **<-** intersect**(**nab, C**)**

nabd **<-** intersect**(**nab, D**)**

nacd **<-** intersect**(**nac, D**)**

nbcd **<-** intersect**(**nbc, D**)**

nabcd **<-** intersect **(**nabc, D**)**

a6 **=** nabcd

a12 **=** nabc**[-**which**(**nabc %in% a6**)]**

a11 **=** nabd**[-**which**(**nabd %in% a6**)]**

a5 **=** nacd**[-**which**(**nacd %in% a6**)]**

a7 **=** nbcd**[-**which**(**nbcd %in% a6**)]**

a15 **=** nab**[-**which**(**nab %in% c**(**a6,a11,a12**))]**

a4 **=** nac**[-**which**(**nac %in% c**(**a6,a5,a12**))]**

a10 **=** nad**[-**which**(**nad %in% c**(**a6,a5,a11**))]**

a13 **=** nbc**[-**which**(**nbc %in% c**(**a6,a7,a12**))]**

a8 **=** nbd**[-**which**(**nbd %in% c**(**a6,a7,a11**))]**

a2 **=** ncd**[-**which**(**ncd %in% c**(**a6,a5,a7**))]**

a9 **=** A**[-**which**(**A %in% c**(**a4,a5,a6,a10,a11,a12,a15**))]**

a14 **=** B**[-**which**(**B %in% c**(**a6,a7,a8,a11,a12,a13,a15**))]**

a1 **=** C**[-**which**(**C %in% c**(**a2,a4,a5,a6,a7,a12,a13**))]**

a3 **=** D**[-**which**(**D %in% c**(**a2,a5,a6,a7,a8,a10,a11**))]**

overlap **<-** list**(**a6**=**a6, a12**=**a12, a11**=**a11, a5**=**a5, a7**=**a7, a15**=**a15,

a4**=**a4, a10**=**a10, a13**=**a13, a8**=**a8, a2**=**a2, a9**=**a9, a14**=**a14, a1**=**a1, a3**=**a3**)**

**}**

**else** **{**

flog.error**(**"Invalid size of input object", name **=** "VennDiagramLogger"**)**

stop**(**"Invalid size of input object"**)**

**}**

return**(**overlap**)**

**}**

#ctrl

RE**=**read.table**(**"k100Mm\_AllSenseTEm\_curatedGTF\_exprCtrl05\_V4.txt", header**=**T, sep**=**"\t"**)** #from sub-pipeline 1

df**=**read.table**(**"BedCov\_OverlapBP\_PassC.txt", header**=**T, sep**=**"\t"**)** #from sub-pipeline 2

dangler**=**read.table**(**"AllRep\_comp\_intersect\_exprCtrl\_V4.txt", header**=**T, sep**=**"\t"**)** #from sub-pipeline 3

RE**=**RE**[!**grepl**(**"\_\_", RE**$**id**)**,**]**

RE**=**RE**[**order**(**RE**$**id**)**,**]**

dangler**=**dangler**[**order**(**dangler**$**id**)**,**]**

df**=**df**[**order**(**df**$**TEm**)**,**]**

REuniq**=**RE**[!**duplicated**(**RE**$**id**)**,**]**

dfuniq**=**df**[!**duplicated**(**df**$**TEm**)**,**]**

dangleruniq**=**dangler**[!**duplicated**(**dangler**$**id**)**,**]**

overlap**<-**cal.overlap**(**x**=**list**(**"count&depth"**=**dfuniq**[**,1**]**, "htseq-count"**=**REuniq**[**,1**]**, "TEFingerprint"**=**dangleruniq**[**,1**]))**

capture.output**(**summary**(**overlap**)**, file **=** "OverlapSummary3groups\_exprCand\_ctrl.txt"**)**

capture.output**(**print**(**overlap**)**, file **=** "OverlapPrint3groups\_exprCand\_ctrl.txt"**)**

a2**=**data.frame**(**id**=**overlap**$**a2**)**

a4**=**data.frame**(**id**=**overlap**$**a4**)**

a5**=**data.frame**(**id**=**overlap**$**a5**)**

a1**=**data.frame**(**id**=**overlap**$**a1**)**

a3**=**data.frame**(**id**=**overlap**$**a3**)**

a6**=**data.frame**(**id**=**overlap**$**a6**)**

a7**=**data.frame**(**id**=**overlap**$**a7**)**

a2**$**C\_tracking**=**"countdepth\_htseq"

a4**$**C\_tracking**=**"countdepth\_TEF"

a5**$**C\_tracking**=**"countdepth\_htseq\_TEF"

a1**$**C\_tracking**=**"countdepthonly"

a3**$**C\_tracking**=**"htseqonly"

a6**$**C\_tracking**=**"htseqTEFonly"

a7**$**C\_tracking**=**"TEFonly"

allexpr**=**rbind**(**a1,a2,a3,a4,a5,a6,a7**)**

write.table**(**allexpr,"AllExpeCandidate\_tracking\_ctrl.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

setwd**(**"./analysis/ECstress\_TEalignment/ExprCandidate"**)**

df**=**read.table**(**"AllExpeCandidate\_tracking\_ctrl.txt", header**=**T, sep**=**"\t"**)**

ref**=**read.table**(**"../../Reference/Curated/AllTEsExpanded\_curated\_tags\_V4.txt", header**=**T, sep**=**"\t"**)**

newdf**=**merge**(**ref, df, by**=**"id"**)**

write.table**(**newdf,"AllExpeCandidate\_ctrl\_tag\_new.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

#mock

RE**=**read.table**(**"k100Mm\_AllSenseTEm\_curatedGTF\_exprMock05\_V4. txt", header**=**T, sep**=**"\t"**)** #from sub-pipeline 1

dangler**=**read.table**(**"/AllRep\_comp\_intersect\_exprMock\_V4.txt", header**=**T, sep**=**"\t"**)** #from sub-pipeline 2

df**=**read.table**(**"BedCov\_OverlapBP\_PassM.txt", header**=**T, sep**=**"\t"**)** #from sub-pipeline 3

RE**=**RE**[!**grepl**(**"\_\_", RE**$**id**)**,**]**

RE**=**RE**[**order**(**RE**$**id**)**,**]**

dangler**=**dangler**[**order**(**dangler**$**id**)**,**]**

df**=**df**[**order**(**df**$**TEm**)**,**]**

dfuniq**=**df**[!**duplicated**(**df**$**TEm**)**,**]**

REuniq**=**RE**[!**duplicated**(**RE**$**id**)**,**]**

dangleruniq**=**dangler**[!**duplicated**(**dangler**$**id**)**,**]**

overlap**<-**cal.overlap**(**x**=**list**(**"count&depth"**=**dfuniq**[**,1**]**, "htseq-count"**=**REuniq**[**,1**]**, "TEFingerprint"**=**dangleruniq**[**,1**]))**

capture.output**(**summary**(**overlap**)**, file **=** "OverlapSummary3groups\_exprCand\_mock.txt"**)**

capture.output**(**print**(**overlap**)**, file **=** "OverlapPrint3groups\_exprCand\_mock.txt"**)**

a2**=**data.frame**(**id**=**overlap**$**a2**)**

a4**=**data.frame**(**id**=**overlap**$**a4**)**

a5**=**data.frame**(**id**=**overlap**$**a5**)**

a1**=**data.frame**(**id**=**overlap**$**a1**)**

a3**=**data.frame**(**id**=**overlap**$**a3**)**

a6**=**data.frame**(**id**=**overlap**$**a6**)**

a7**=**data.frame**(**id**=**overlap**$**a7**)**

a2**$**C\_tracking**=**"countdepth\_htseq"

a4**$**C\_tracking**=**"countdepth\_TEF"

a5**$**C\_tracking**=**"countdepth\_htseq\_TEF"

expr1**=**rbind**(**a2,a4,a5**)**

a1**$**C\_tracking**=**"countdepthonly"

a3**$**C\_tracking**=**"htseqonly"

a6**$**C\_tracking**=**"htseqTEFonly"

a7**$**C\_tracking**=**"TEFonly"

allexpr**=**rbind**(**a1,a2,a3,a4,a5,a6,a7**)**

write.table**(**allexpr,"AllExpeCandidate\_tracking\_mock.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

setwd**(**"./analysis/ECstress\_TEalignment/ExprCandidate"**)**

df**=**read.table**(**"AllExpeCandidate\_tracking\_mock.txt", header**=**T, sep**=**"\t"**)**

ref**=**read.table**(**"../../Reference/Curated/AllTEsExpanded\_curated\_tags\_V4.txt", header**=**T, sep**=**"\t"**)**

newdf**=**merge**(**ref, df, by**=**"id"**)**

write.table**(**newdf,"AllExpeCandidate\_mock\_tag\_new.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

#yeast

RE**=**read.table**(**"k100Mm\_AllSenseTEm\_curatedGTF\_exprYeast05\_V4.txt", header**=**T, sep**=**"\t"**)** #from sub-pipeline 1

dangler**=**read.table**(**"AllRep\_comp\_intersect\_exprYeast\_V4.txt", header**=**T, sep**=**"\t"**)** #from sub-pipeline 2

df**=**read.table**(**"BedCov\_OverlapBP\_PassY.txt", header**=**T, sep**=**"\t"**)** #from sub-pipeline 3

RE**=**RE**[!**grepl**(**"\_\_", RE**$**id**)**,**]**

RE**=**RE**[**order**(**RE**$**id**)**,**]**

dangler**=**dangler**[**order**(**dangler**$**id**)**,**]**

df**=**df**[**order**(**df**$**TEm**)**,**]**

dfuniq**=**df**[!**duplicated**(**df**$**TEm**)**,**]**

REuniq**=**RE**[!**duplicated**(**RE**$**id**)**,**]**

dangleruniq**=**dangler**[!**duplicated**(**dangler**$**id**)**,**]**

overlap**<-**cal.overlap**(**x**=**list**(**"count&depth"**=**dfuniq**[**,1**]**, "htseq-count"**=**REuniq**[**,1**]**, "TEFingerprint"**=**dangleruniq**[**,1**]))**

capture.output**(**summary**(**overlap**)**, file **=** "OverlapSummary3groups\_exprCand\_yeast.txt"**)**

capture.output**(**print**(**overlap**)**, file **=** "OverlapPrint3groups\_exprCand\_yeast.txt"**)**

a2**=**data.frame**(**id**=**overlap**$**a2**)**

a4**=**data.frame**(**id**=**overlap**$**a4**)**

a5**=**data.frame**(**id**=**overlap**$**a5**)**

a1**=**data.frame**(**id**=**overlap**$**a1**)**

a3**=**data.frame**(**id**=**overlap**$**a3**)**

a6**=**data.frame**(**id**=**overlap**$**a6**)**

a7**=**data.frame**(**id**=**overlap**$**a7**)**

a2**$**C\_tracking**=**"countdepth\_htseq"

a4**$**C\_tracking**=**"countdepth\_TEF"

a5**$**C\_tracking**=**"countdepth\_htseq\_TEF"

expr1**=**rbind**(**a2,a4,a5**)**

a1**$**C\_tracking**=**"countdepthonly"

a3**$**C\_tracking**=**"htseqonly"

a6**$**C\_tracking**=**"htseqTEFonly"

a7**$**C\_tracking**=**"TEFonly"

allexpr**=**rbind**(**a1,a2,a3,a4,a5,a6,a7**)**

write.table**(**allexpr,"AllExpeCandidate\_tracking\_yeast.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

setwd**(**"./analysis/ECstress\_TEalignment/ExprCandidate"**)**

df**=**read.table**(**"AllExpeCandidate\_tracking\_yeast.txt", header**=**T, sep**=**"\t"**)**

ref**=**read.table**(**" ../../Reference/Curated/AllTEsExpanded\_curated\_tags\_V4.txt", header**=**T, sep**=**"\t"**)**

newdf**=**merge**(**ref, df, by**=**"id"**)**

write.table**(**newdf,"AllExpeCandidate\_yeast\_tag\_new.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

#botrytis

RE**=**read.table**(**"k100Mm\_AllSenseTEm\_curatedGTF\_exprBotrytis05\_V4.txt", header**=**T, sep**=**"\t"**)** #from sub-pipeline 1

dangler**=**read.table**(**"AllRep\_comp\_intersect\_exprBotrytis\_V4.txt", header**=**T, sep**=**"\t"**)** #from sub-pipeline 2

df**=**read.table**(**"BedCov\_OverlapBP\_PassB.txt", header**=**T, sep**=**"\t"**)** #from sub-pipeline 3

RE**=**RE**[!**grepl**(**"\_\_", RE**$**id**)**,**]**

RE**=**RE**[**order**(**RE**$**id**)**,**]**

dangler**=**dangler**[**order**(**dangler**$**id**)**,**]**

df**=**df**[**order**(**df**$**TEm**)**,**]**

dfuniq**=**df**[!**duplicated**(**df**$**TEm**)**,**]**

REuniq**=**RE**[!**duplicated**(**RE**$**id**)**,**]**

dangleruniq**=**dangler**[!**duplicated**(**dangler**$**id**)**,**]**

overlap**<-**cal.overlap**(**x**=**list**(**"count&depth"**=**dfuniq**[**,1**]**, "htseq-count"**=**REuniq**[**,1**]**, "TEFingerprint"**=**dangleruniq**[**,1**]))**

capture.output**(**summary**(**overlap**)**, file **=** "OverlapSummary3groups\_exprCand\_botrytis.txt"**)**

capture.output**(**print**(**overlap**)**, file **=** "OverlapPrint3groups\_exprCand\_botrytis.txt"**)**

a2**=**data.frame**(**id**=**overlap**$**a2**)**

a4**=**data.frame**(**id**=**overlap**$**a4**)**

a5**=**data.frame**(**id**=**overlap**$**a5**)**

a1**=**data.frame**(**id**=**overlap**$**a1**)**

a3**=**data.frame**(**id**=**overlap**$**a3**)**

a6**=**data.frame**(**id**=**overlap**$**a6**)**

a7**=**data.frame**(**id**=**overlap**$**a7**)**

a2**$**C\_tracking**=**"countdepth\_htseq"

a4**$**C\_tracking**=**"countdepth\_TEF"

a5**$**C\_tracking**=**"countdepth\_htseq\_TEF"

expr1**=**rbind**(**a2,a4,a5**)**

a1**$**C\_tracking**=**"countdepthonly"

a3**$**C\_tracking**=**"htseqonly"

a6**$**C\_tracking**=**"htseqTEFonly"

a7**$**C\_tracking**=**"TEFonly"

allexpr**=**rbind**(**a1,a2,a3,a4,a5,a6,a7**)**

write.table**(**allexpr,"AllExpeCandidate\_tracking\_botrytis.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**

setwd**(**"./analysis/ECstress\_TEalignment/ExprCandidate"**)**

df**=**read.table**(**"AllExpeCandidate\_tracking\_botrytis.txt", header**=**T, sep**=**"\t"**)**

ref**=**read.table**(**"../../Reference/Curated/AllTEsExpanded\_curated\_tags\_V4.txt", header**=**T, sep**=**"\t"**)**

newdf**=**merge**(**ref, df, by**=**"id"**)**

write.table**(**newdf,"AllExpeCandidate\_botrytis\_tag\_new.txt", col.names**=**T, row.names**=**F, sep**=**"\t", quote**=**F**)**