Chapter 19

# Sequencing Trait-Associated Mutations (STAM) to Clone Rust Resistance Genes

Fei Ni, Yang Yu, Lynn Epstein, Daolin Fu, and Jiajie Wu

**Abstract**

Sequencing trait-associated mutations (STAM) is a simple and straightforward gene cloning method that was developed in wheat. It uses full-length isoform sequencing (Iso-Seq) of the wild type as the reference and employs transcriptome sequencing of multiple, independently derived mutants for gene cloning. The STAM method eliminates the need for fine-mapping or a high-quality whole genome assembly of a specific wheat cultivar, and it could also be used in other plant species with complex genomes. Detailed, bioinformatic analysis protocol and tips for STAM are provided in this chapter.

**Key words** Gene cloning, STAM, Iso-Seq, RNA-Seq, Mutagenesis, Ethyl methanesulfonate (EMS), Resistance gene, Stripe rust

## 1 Introduction

Wheat remains difficult for the identification of a gene of interest due to its polyploid nature, huge genome size, high percentage of repetitive DNA, as well as its relatively long life cycle of 4–6 months. For wheat disease resistance (R gene), there are only around 50 genes have been successfully cloned [1]. Among them, most were cloned through a map-based cloning strategy, which is tedious and time-consuming. Map-based cloning is also challenging when the target gene is located in recombination suppressed region, and particularly when a reference genome for the mapping parent is not available. Although the sequencing technologies are developing very fast in recent years, the cost of sequencing and assembling the genome of a specific wheat cultivar is still high or unaffordable for most laboratories and requires computation expertise. By taking advantage of combined utilization of next-generation sequencing and mutagenesis, the identification of genes-of-interest in wheat becomes less complicated than before. For example, wheat stripe rust resistance genes *Yr7*, *Yr5/YrSP*, and stem rust resistance genes

*Sr22*, *Sr26*, *Sr27*, *Sr45*, and *Sr61* have been successfully cloned by using MutRenSeq [2–4]. However, MutRenSeq is only useful for nucleotide binding and leucine-rich repeat (NLR) gene cloning and cannot be applied to other structural class of genes.

The PacBio isoform sequencing (Iso-Seq) is a workflow for sequencing and analyzing full-length (FL) transcript isoforms [5]. It allows direct sequencing of transcripts up to 10 kb without use of a reference genome. The full-length complementary DNA (cDNA) is sequenced in a single-sequencing read with no downstream bioinformatics assembly required. Thus, Iso-Seq outputs a large portion of the gene contents in a genome without sequencing the repetitive DNA region and is cost-effective compared to whole-genome sequencing. Disease resistance genes are prone to be captured in an Iso-Seq since they are usually expressed in tissues upon pathogen infection.

STAM uses a combination of mutant lines with loss of resistance and Iso-Seq to identify resistance genes [6]. STAM does not need gene fine-mapping and does not require a genomic assembly and therefore can complete gene cloning in a relatively faster and simpler way. This method has successfully been applied to clone the stripe rust resistance gene *YrNAM* from hexaploid wheat [6]. At the same time, a similar method which is called MutIsoSeq has successfully been applied to clone the wheat leaf rust resistance gene *Lr9/Lr58* [7]. Theoretically, these methods can also be applied to other plant species that are amenable to mutagenesis. In this chapter, a detailed step-by-step protocol of STAM is described. The protocol will be described in the context of *YrNAM* as an example [6].

STAM uses isoform sequencing (Iso-Seq) of the wild-type (WT) and transcriptome sequencing of relevant mutants. First, full-length, WT transcripts are used to generate a reference of nonredundant transcripts. Then, RNA-Seq reads of loss-of-function mutants are mapped onto the reference, variants are called, and counts of mutated transcripts are tallied. Accordingly, the STAM workflow contains three parts (Fig. 1). In Subheading 3.1, we describe the de novo transcriptome reconstruction; in Subheading 3.2, construction of final transcriptome reference; and in Subheading 3.3, variant calling and identification of a candidate gene.

## 2 Materials

### 2.1 Plant Materials and Sequencing Data

1. **Wild type for Gene of Interest**

   Homozygous plants carrying the gene of interest are used as WT for ethyl methanesulfonate (EMS) treatment (*see* **Note 1**). Protocols for seed mutagenesis with EMS are derived from Wang et al. [8] and Mago et al. [9].

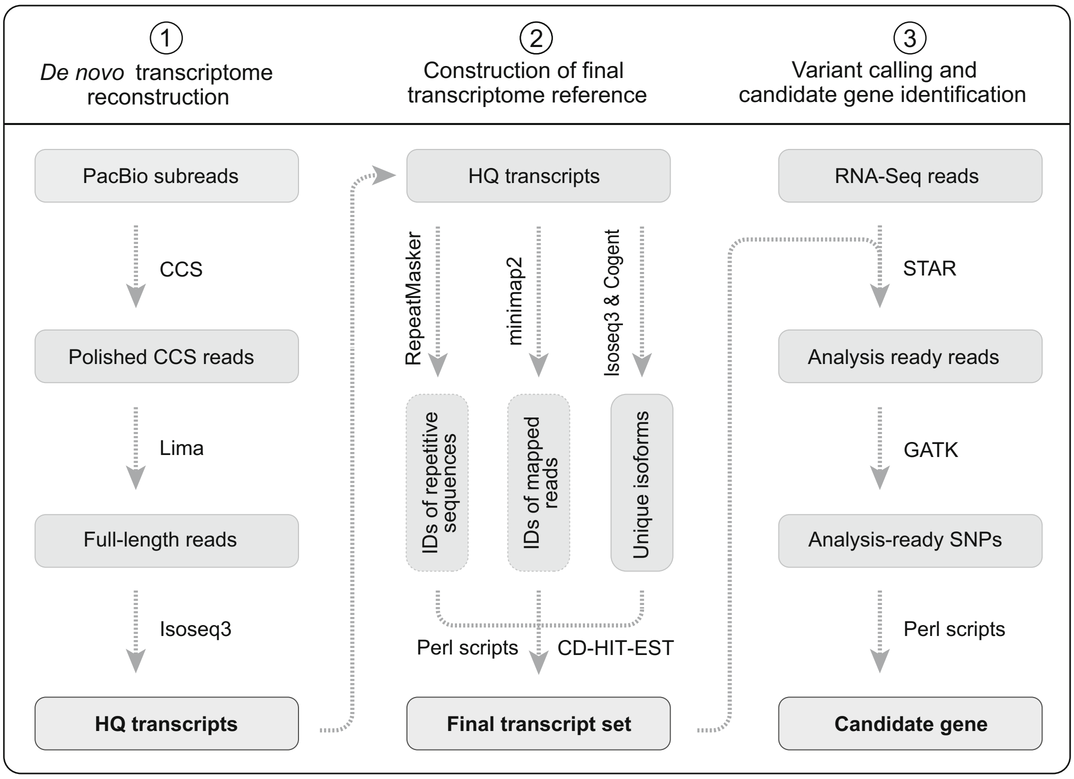

**Fig. 1** Workflow for sequencing trait-associated mutations, divided by task and annotated with appropriate software for use at each step in the pipeline

Iso-Seq is performed for WT plants, using the PacBio Sequel II system. Data size of 90 Gb (subreads) is recommended for bread wheat.

2. **EMS Mutants**

   We initially screen the EMS population of WT to obtain loss-of-function mutants, repeat the phenotype investigation in subsequent generations of the mutants, keep self-pollinating, and harvest plants which show a homozygous loss-of-function phenotype (*see* **Note 2**).

   RNA-Seq is performed for multiple independently derived mutants (*see* **Note 3**). The sequencing platform could be Illumina HiSeq 2500 with paired-end reads of 2 × 150 bp. Data size of 15–20 Gb is recommended for each wheat mutant.

## 2.2 Computer Setup

Bioinformatics analysis requires a server installed with a Linux operating system. Python needs to be installed to run software for processing third-generation sequencing data. The software circular consensus sequencing (CCS), Lima, and IsoSeq3 require Python version 2.7 (https://www.python.org/downloads/), while Cogent and cDNA_Cupcake require Python version 3.7. It is necessary to download the appropriate version. In order to use software for different Python versions, a separate conda environment can be created.

### 2.3 Databases Required

1. IWGSC RefSeq v2.1 [https://wheat-urgi.versailles.inra.fr/Seq-Repository].
2. Poaceae repeat element database [http://pgsb.helmholtz-muenchen.de/plant/index.jsp].
3. Genome of wheat stripe rust pathogen *Puccinia striiformis* f. sp. *tritici* [https://www.ncbi.nlm.nih.gov/datasets/genome/GCF_021901695.1/].

### 2.4 Public Bioinformatics Tools and Perl Scripts

1. CCS (https://github.com/PacificBiosciences/ccs).
2. Lima (https://github.com/pacificbiosciences/barcoding).
3. Isoseq3 (https://github.com/pacificbiosciences/isoseq).
4. RepeatMasker (http://www.repeatmasker.org).
5. Minimap2 (https://github.com/lh3/minimap2).
6. Pbmm2 (https://github.com/PacificBiosciences/pbmm2).
7. Cogent (https://github.com/Magdoll/Cogent).
8. cDNA_Cupcake (https://github.com/Magdoll/cDNA_Cupcake).
9. Samtools (https://www.htslib.org).
10. Fastp (https://github.com/OpenGene/fastp).
11. GATK (https://gatk.broadinstitute.org).
12. fish_fa.pl (https://github.com/Feiny/STAM/tree/main/Scripts).
13. fish_longest_fa.pl (https://github.com/Feiny/STAM/tree/main/Scripts).
14. remove_repeats.pl (https://github.com/Feiny/STAM/tree/main/Scripts).
15. select_homo.pl (https://github.com/Feiny/STAM/tree/main/Scripts).
16. select_mutation1.pl (https://github.com/Feiny/STAM/tree/main/Scripts).
17. select_mutation2.pl (https://github.com/Feiny/STAM/tree/main/Scripts).
18. filter_mutation.pl (https://github.com/Feiny/STAM/tree/main/Scripts).

## 3 Methods

This approach has three main pipelines: (Subheading 3.1) De novo transcriptome reconstruction, (Subheading 3.2) Construction of final transcriptome reference, and (Subheading 3.3) Variant calling and candidate gene identification. All scripts included herein are to be executed using the Linux command line. Command lines are shown below with a "$" prefix in italics, and descriptions are shown before commands with a "#" prefix. Where a command does not fit on a single page line but must be executed as a single command line, the escape character "\" is used to separate lines.

### 3.1 De Novo Transcriptome Reconstruction

This step is to generate the high-quality (HQ) full-length transcripts of the WT with PacBio raw sequencing reads (PacBio subreads).

1. **Generate Circular Consensus Sequencing (CCS) Reads Using *ccs***

```
$ ccs -j 64 --min-passes 3 --min-length 10 \
--min-rq 0.99 movie.subreads.bam movie.ccs.bam
```

   The "movie.subreads.bam" is the raw sequencing reads file produced by PacBio SMRT sequencing. The "movie.ccs.bam" file is the output of *ccs* and contains the circular consensus sequencing (CCS) reads. If the sequencing data includes two or more "subreads.bam" files, run the *ccs* command for each file, respectively.

2. **Generate Full-Length (FL) Reads Using *lima***
   Process the CCS reads to remove the 5′ and 3' cDNA primers and barcodes. The sequences of 5′ and 3' cDNA primers used in library construction are listed in a file named "primers.fasta" which is provided by the sequence service provider. One output file named "movie.fl.primer_5p--primer_3p.bam" will be generated.

```
$ lima --isoseq movie.ccs.bam \
primers.fasta movie.fl.bam
```

3. **Generate Full-Length, Non-concatemer (FLNC) Reads Using *isoseq3 refine***
   Remove PolyA tails and artificial concatemers and generate FLNC reads.

```
$ isoseq3 refine --require-polya \
movie.fl.primer_5p--primer_3p.bam \
primers.fasta movie.flnc.bam
```

4. **Generate High-Quality (HQ) Transcripts Using *isoseq3 cluster***

   Cluster the FLNC reads and generate polished transcripts. *See* **Note 4** if there are more than one "flnc.bam" files need to be processed.

```
$ isoseq3 cluster --verbose --use-qvs movie.flnc.bam \
movie.clustered.bam
```

## 3.2 Construction of Final Transcriptome Reference

This step is to further clean the HQ transcripts by removing repetitive DNA and pathogen contamination and then collapsing redundant transcripts into unique isoforms.

1. **Mask Transposable Elements Using RepeatMasker**

   RepeatMasker is used to mask the transposable elements in the HQ transcript sequences. Then, the command *grep* is used to generate a file containing the IDs of repetitive sequences, which is used to obtain the final transcripts.

```
$ zcat movie.clustered.hq.fasta.gz > \
movie.clustered.hq.fasta
$ RepeatMasker -gff -nolow -no_is -xsmall -pa 12 \
movie.clustered.hq.fasta
$ grep -v '^#' movie.clustered.hq.fasta.out.gff | \
cut -f 1 | sort -t "/" -k 2 -n > \
movie.masked.hq.fasta.out.id
```

2. **Remove Transcripts from the Pathogen (*see* Note 5)**

   To make sure the target gene is expressed in the plant tissue used for Iso-Seq, thus could be sequenced, it is recommended to artificially infect the WT plants with pathogen before sampling for sequencing. This brings contamination of transcripts from the pathogen. The sequence reads from pathogen can be removed before generating the transcript reference for STAM analysis. For example, in the analysis of *YrNAM* gene cloning, the HQ transcripts are aligned to the genome assembly of wheat stripe rust fungus *Pst* 134E16 [10]. To do so, use *minimap2* to map the HQ transcripts to the *Pst* reference genome, and use command *awk* to generate a file that contains the IDs of mapped reads, which can also be used to obtain the final transcripts.

```
$ minimap2 -t 8 -Y -R \
"@RG\tID:Sample\tSM:hs\tLB:ga\tPL:PacBio" \
--MD -ax splice:hq -uf --secondary=no \
Pst134e.reference.fasta movie.clustered.hq.fasta \
> aligned.Pst134e.sam
```

```
$ awk '$3!="*"{print $1}' aligned.Pst134e.sam > \
aligned.mapped.id
```

3. **Collapse Redundant Transcripts into Unique Isoforms**

   STAM uses nonredundant transcripts as the reference for WT. It takes three steps to obtain nonredundant transcripts. First, we align HQ transcripts to the Chinese Spring reference genome IWGSC RefSeq v2.1 using *pbmm2*. Then, for the mapped reads, we remove redundancy using *isoseq collapse* tool, and for unmapped reads, run python tools *Cogent* and *cDNA_Cupcake*. Finally, we merge nonredundant transcripts derived from mapped and unmapped reads using CD-HIT-EST tool with the sequence identity threshold at 100% (-c 1). The resulting nonredundant transcripts will be used as the reference for subsequent variant calling.

4. **Map the HQ Transcripts to a Reference Genome**

```
$ pbmm2 align --preset ISOSEQ --sort --unmapped \
movie.clustered.hq.bam iwgsc_refseqv2.1_assembly.fa \
movie.mapped.bam
```

   To run *pbmm2*, the parameter of "--unmapped" is used to include unmapped transcripts in the output bam file. The whole genome assembly of Chinese Spring (iwgsc_refseq-v2.1_assembly.fa) is used as a reference genome for alignment. Alternatively, genome assemblies from other cultivars, such as Fielder [11] and Kenong 9204 [12], can also be used.

5. **Collapse Mapped Reads into Unique Isoforms**

```
$ isoseq collapse movie.mapped.bam movie.collapsed.gff
```

   Detailed descriptions and explanations of output can be found in the Iso-Seq Collapse Web page. Among the output files, "movie.collapsed.fasta" contains the collapsed transcripts.

6. **Collapse Unmapped Reads into Unique Isoforms**

   A series of command lines needs to be conducted to collapse unmapped reads into unique isoforms. We extract unmapped reads from the output file "movie.mapped.bam" in **Step (4)**; then, partition the unmapped reads into gene families, and reconstruct transcribed regions for each gene using *Cogent* and *cDNA_Cupcake*. A file named "hq_transcripts.fasta.collapsed.filtered.rep.fa" will be generated.

   # To extract unmapped reads.

```
$ samtools fasta -f 4 movie.mapped.bam > \
movie.unmapped.fasta
```

# To create the k-mer profile of unmapped reads and calculate pairwise distances.

```
$ run_mash.py -k 30 --cpus=6 movie.unmapped.fasta
```

# To process the distance file and create the partitions.

```
$ process_kmer_to_graph.py movie.unmapped.fasta \
movie.unmapped.fasta.s1000k30.dist \
partitions_movie/ movie
```

# To generate batch commands to run family finding on each bin.

```
$ generate_batch_cmd_for_Cogent_reconstruction.py parti-
tions_movie \
> batch_cmd_for_Cogent_reconstruction.sh && \
sh batch_cmd_for_Cogent_reconstruction.sh
```

# To get the unassigned sequences and concatenate with Cogent contigs.

```
$ tail -n 1 movie.partition.txt | tr ',' '\n' \
> movie.unassigned.list
$ perl fish_fa.pl movie.unassigned.list \
movie.clustered.hq.fasta.gz > unassigned.fasta
$ mkdir collected && cd collected
$ cat ../partitions_movie/*/cogent2.renamed.fasta \
../unassigned.fasta > cogent.fake_genome.fasta
```

# To collapse the redundant isoforms. The “movie.clustered.hq.fasta.gz” is generated in Subheading 3.1 using *isoseq3 cluster*.

```
$ minimap2 -ax splice -t 30 -uf --secondary=no \
cogent.fake_genome.fasta ../movie.clustered.hq.fasta.gz \
> hq_transcripts.fasta.sam
```

# “movie.clustered.cluster_report.csv” is generated in Subheading 3.1 using *isoseq3 cluster*.

```
$ sort -k 3,3 -k 4,4n hq_transcripts.fasta.sam \
> hq_transcripts.fasta.sorted.sam
$ collapse_isoforms_by_sam.py -- \
input ../movie.clustered.hq.fasta \
-s hq_transcripts.fasta.sorted.sam -o \
hq_transcripts.fasta
$ get_abundance_post_collapse.py \
hq_transcripts.fasta.collapsed \
../movie.clustered.cluster_report.csv
```

# "hq_transcripts.fasta.collapsed.filtered.rep.fa" contains collapsed isoforms.

```
$ filter_away_subset.py hq_transcripts.fasta.collapsed
```

7. **Merge Nonredundant Transcripts to Generate the Final Transcript Reference**

   We extract the longest transcript for each gene, remove repetitive sequences or contaminants, and run *cd-hit-est*. The output file named "movie.final.transcripts.fasta" is the final set of transcripts that will be used as a reference for variant calling.

```
$ perl fish_longest_fa.pl movie.collapsed.fasta \
> movie.longest.fasta
$ cat movie.longest.fasta \
 ./collected/hq_transcripts.fasta.collapsed.filtered.rep.fa \
> movie.combined.fasta
$ perl remove_repeats.pl movie.masked.hq.fasta.out.id \
aligned.mapped.id movie.combined.fasta \
> movie.combined.filtered.fasta
$ cd-hit-est -i movie.combined.filtered.fasta \
-o movie.final.transcripts.fasta -c 1 -T 12
```

### 3.3 Variant Calling and Identification of a Candidate Gene

Five to ten independent mutants (*see* **Note 3**) can be sequenced via RNA-Seq to perform STAM analysis. We first process the raw sequencing data for quality control using *fastp*, then map the reads to the full-length transcripts reference, and perform variant calling by using Genome Analysis Toolkit (GATK) or SAMtools. Here, the GATK method is demonstrated as an example.

1. **Adapter Trimming and Quality Filtration of the RNA-Seq Data Using *fastp***

```
$ for name in $(cat sample.id)
do
 fastp --thread 2 \
 --in1 $name.R1.fastq.gz \
 --in2 $name.R2.fastq.gz \
 --out1 $name.R1.clean.fq.gz \
 --out2 $name.R2.clean.fq.gz \
 -q 3 \
 -u 50 \
 --length_required 150 \
 -h $name.html \
 -j $name.json
done
```

All sample names are recorded in the file "sample.id." This file is a plaintext file created by the user containing each sample name separated by a new line.

2. **Generate Reference Index**

```
$ ref='<path_to_reference>/\
movie.final.transcripts.fasta'
$ STAR --runThreadN 8 \
 --runMode genomeGenerate \
 --genomeDir <path_to_reference>/ \
 --genomeFastaFiles $ref
$ java -jar picard.jar CreateSequenceDictionary \
 --REFERENCE $ref \
 --OUTPUT \
<path_to_reference>/movie.final.transcripts.dict
$ samtools faidx \
<path_to_reference>/movie.final.transcripts.fasta
```

3. **Map RNA-Seq Reads from EMS Mutants to the Transcript Reference (*see* Note 6)**

```
    $ refdir='<path_to_reference_index>'
    $ for name in $(cat sample.id)
    do
     STAR --runThreadN 24 \
     --genomeDir $refdir \
     --readFilesIn $name.R1.clean.fq.gz $name.R2.clean.fq.gz \
     --readFilesCommand zcat \
     --outSAMtype BAM SortedByCoordinate \
     --limitBAMsortRAM 50000000000 \
     --outFileNamePrefix $name. \
     --outSAMmapqUnique 60
java -XX:ParallelGCThreads=12 -jar picard.jar \
     AddOrReplaceReadGroups \
     --INPUT $name.Aligned.sortedByCoord.out.bam \
     --OUTPUT $name.RG.bam \
     --SORT_ORDER coordinate \
     --MAX_RECORDS_IN_RAM 1000000 \
     --RGLB $name \
     --RGPL ILLUMINA \
     --RGPU BARCODE \
     --RGSM $name
java -XX:ParallelGCThreads=12 -jar picard.jar \
     MarkDuplicates \
     --INPUT $name.RG.bam \
      --OUTPUT $name.dedupped.bam \
     --CREATE_INDEX true \
     --VALIDATION_STRINGENCY SILENT \
     --METRICS_FILE $name.metrics
rm -rf $name.Aligned.sortedByCoord.out.bam \
$name.Log.out $name.Log.progress.out \
$name.SJ.out.tab $name.RG.bam $name.metrics
done
```

4. **Variant Calling Using GATK**

```
$ ref='<path_to_reference>/\
movie.final.transcripts.fasta'
$ for name in $(cat sample.id)
do
 gatk --java-options -Xmx10G HaplotypeCaller \
-R $ref -I $name.dedupped.bam -OVI false -ERC GVCF \
 -O $name.g.vcf.gz 1>$name.hc.log 2>&1
 gatk --java-options -Xmx10G IndexFeatureFile \
 -I $name.g.vcf.gz
done
```

# For example, to call variants based on seven mutants. Adjust the scripts of "*--variant Mutant*.g.vcf.gz*" accordingly based on the number of mutants used.

```
$ gatk --java-options -Xmx10G CombineGVCFs \
 -R $ref \
 --variant Mutant1.g.vcf.gz \
 --variant Mutant2.g.vcf.gz \
 --variant Mutant3.g.vcf.gz \
 --variant Mutant4.g.vcf.gz \
 --variant Mutant5.g.vcf.gz \
 --variant Mutant6.g.vcf.gz \
 --variant Mutant7.g.vcf.gz \
 --variant WT.g.vcf.gz \
 -O cohort.g.vcf.gz

$ gatk --java-options -Xmx10G GenotypeGVCFs \
 -R $ref \
 -V cohort.g.vcf.gz \
 -O output.vcf.gz
```

# Variant filtering.

```
$ gatk --java-options -Xmx10G VariantFiltration \
 -R $ref \
 -V output.vcf.gz \
 -O output.filtered.vcf.gz \
 --filter-name "GATK_Filter" \
 --filter-expression "DP < 5.0 || \
FS > 60.0 || MQ < 40.0 || QD < 2.0"
```

5. **Mutation Filtering**

   We filter the mutation sites identified in last step using perl scripts “select_homo.pl”, “select_mutation1.pl,” and “select_-mutation2.pl”. Only those single nucleotide ploymorphisms (SNPs) that can meet certain criteria are kept for further analysis (*see* **Note 7**).

   # Filter SNPs that are homozygous and with read depth ≥ 5.

```
$ zcat output.filtered.vcf.gz | perl select_homo.pl \
> homo.txt
```

   # Filter SNPs that are EMS type mutations (G-to-A and C-to-T transitions).

```
$ perl select_mutation1.pl homo.txt > mutation1.txt
```

   # Filter transcripts that only contain one SNP for each mutant line.

```
$ perl select_mutation2.pl mutation1.txt \
> mutation2.txt
```

   # Filtering for SNPs with missing rate ≤ 60% and must be at least present in WT.

```
$ perl filter_mutation.pl mutation2.txt \
> mutation3.txt
```

6. **Identification of Candidate Transcripts for the Causal Gene**

   With the nonredundant transcripts carrying mutations, count the number of mutants that have a mutation site in each transcript. Those transcripts that have mutations in maximum number of mutant lines are considered to be candidates for the causal gene. For example, in the STAM analysis of seven *YrNAM* loss of function mutants, 7434 transcripts were found to contain SNPs (meet the criteria in **step 4**). Among them, 5680 transcripts have mutations in only one of those seven mutant lines, 760 transcripts have mutations in two lines, 72 transcripts have mutations in three lines, 3 transcripts have mutations in four lines, 1 transcript has mutations in six lines, and no transcript has mutations in five or seven lines. Thus, the one transcript that has mutations in six mutant lines represents a good candidate.

```
$ grep -v '^#' mutation3.txt | cut -f 1 | sort \
| uniq -c | sort -k 1 -n -r > mutation_count.txt
```

## 4 Notes

1. Theoretically, STAM could also be adapted for heterozygous plants. However, homozygous plants are preferred to be used as WT. Wheat cultivars, advanced lines, landraces, and so on are suitable materials. In addition, the WT should carry a single gene that determines the phenotype of interest. For gene cloning in plant materials with a polygenic trait, STAM may not be a good option. However, it is feasible to screen monogenic descendants from these materials first and then perform STAM with monogenic plants as the WT.
2. Obtaining homozygous loss-of-function mutants is critical for a successful STAM analysis. For mutant lines, it is important to confirm the phenotype for at least two generations and check if the mutant line is non-segregating. For studies on diseases such as stripe rust, the phenotyping in different generations should be investigated using the same pathogen races. Besides, when sampling the mutants for RNA-Seq, the best way is to sample the same plant that was screened for phenotype.
3. Based on our experiences in the cloning of *YrNAM* [6], and previous publications such as the cloning of *Sr22* and *Sr45* [13], as well as the probability analysis [14], five to ten independently derived EMS mutants should be sufficient for high-throughput sequencing methods for gene identification in wheat. It can be expected to obtain approximately ten independent loss-of-function mutants in an EMS population of 1500 individuals.
4. For example, in the cloning of *YrNAM*, two RNA samples (0 and 24 h post stripe rust pathogen infection) were sequenced. Therefore, two raw data files, "movie-1.subreads.bam" and "movie-2.subreads.bam," were produced in Iso-Seq. Accordingly, two FLNC bam files, "movie-1.flnc.bam" and "movie-2.flnc.bam," are generated in **step 3.1**. To generate HQ transcripts with two or more FLNC bam files, we can use the *ls* command to include all bam files into a list (flnc.fofn), and then generate a single HQ transcripts file using the isoseq3 cluster tool.

```
$ ls movie*.flnc.bam > flnc.fofn
$ isoseq3 cluster --verbose --use-qvs \
flnc.fofn movie.clustered.bam
```

5. This step is optional.
6. With large enough files or multiple threads, the option "--outSAMtype BAM SortedByCoordinate" of STAR is known to sometimes produce a fatal error. Users can first generate unsorted bam files with the option "--outSAMtype BAM Unsorted," and then use samtools to sort them.
7. SNPs considered should meet all the following conditions: (1) homozygous sites; (2) typical EMS mutations (either G to A or C to T mutation); (3) count of SNPs per transcript in each mutant = 1; (4) read depth 5 or more; and (5) missing rate 60% or less. Among the above criteria, the first three should not be changed, while the last two (read depth and missing rate) can be adjusted.

## Acknowledgments

This work was funded by the National Key R&D Program of China (2022YFD1201700), the National Natural Science Foundation of China (31971935), and the Natural Science Foundation of Shandong Province (ZR2022MC009).

## References

1. Hafeez AN, Arora S, Ghosh S et al (2021) Creation and judicious application of a wheat resistance gene atlas. Mol Plant 14(7): 1053–1070. https://doi.org/10.1016/j.molp.2021.05.014
2. Marchal C, Zhang J, Zhang P et al (2018) BED-domain-containing immune receptors confer diverse resistance spectra to yellow rust. Nat Plants 4(9):662–668. https://doi.org/10.1038/s41477-018-0236-4
3. Zhang J, Hewitt TC, Boshoff WHP et al (2021) A recombined *Sr26* and *Sr61* disease resistance gene stack in wheat encodes unrelated NLR genes. Nat Commun 12(1):3378. https://doi.org/10.1038/s41467-021-23738-0
4. Upadhyaya NM, Mago R, Panwar V et al (2021) Genomics accelerated isolation of a new stem rust avirulence gene-wheat resistance gene pair. Nat Plants 7(9):1220–1228. https://doi.org/10.1038/s41477-021-00971-5
5. Rhoads A, Au KF (2015) PacBio sequencing and its applications. Genomics Proteomics Bioinformatics 13(5):278–289. https://doi.org/10.1016/j.gpb.2015.08.002
6. Ni F, Zheng Y, Liu X et al (2023) Sequencing trait-associated mutations to clone wheat rust-resistance gene *YrNAM*. Nat Commun 14(1): 4353. https://doi.org/10.1038/s41467-023-39993-2
7. Wang Y, Abrouk M, Gourdoupis S et al (2023) An unusual tandem kinase fusion protein confers leaf rust resistance in wheat. Nat Genet 55(6):914–920. https://doi.org/10.1038/s41588-023-01401-2
8. Wang W, Guan X, Gan Y et al (2024) Creating large EMS populations for functional genomics and breeding in wheat. J Integr Agr 23(2): 484–493. https://doi.org/10.1016/j.jia.2023.05.039
9. Mago R, Till B, Periyannan S et al (2017) Generation of loss-of-function mutants for wheat rust disease resistance gene cloning. In:

Periyannan S (ed) Wheat rust diseases: methods and protocols. Springer, New York, pp 199–205. https://doi.org/10.1007/978-1-4939-7249-4_17

10. Schwessinger B, Jones A, Albekaa M et al (2022) A chromosome scale assembly of an Australian *Puccinia striiformis* f. sp. *tritici* isolate of the *PstS1* lineage. Mol Plant-Microbe Interact 35(3):293–296. https://doi.org/10.1094/MPMI-09-21-0236-A
11. Sato K, Abe F, Mascher M et al (2021) Chromosome-scale genome assembly of the transformation-amenable common wheat cultivar 'Fielder'. DNA Res 28:3. https://doi.org/10.1093/dnares/dsab008
12. Shi X, Cui F, Han X et al (2022) Comparative genomic and transcriptomic analyses uncover the molecular basis of high nitrogen-use efficiency in the wheat cultivar Kenong 9204. Mol Plant 15(9):1440–1456. https://doi.org/10.1016/j.molp.2022.07.008
13. Steuernagel B, Periyannan SK, Hernandez-Pinzon I et al (2016) Rapid cloning of disease-resistance genes in plants using mutagenesis and sequence capture. Nat Biotechnol 34(6):652–655. https://doi.org/10.1038/nbt.3543
14. Yu G, Matny O, Champouret N et al (2022) *Aegilops sharonensis* genome-assisted identification of stem rust resistance gene *Sr62*. Nat Commun 13(1):1607. https://doi.org/10.1038/s41467-022-29132-8