Week1-2: Transposable Element Annotation and Classification

TEs are broadly categorized into **Class I** (retrotransposons) and **Class II** (DNA transposons), which differ in their mechanisms of transposition.

- **Class I TEs (Retrotransposons)** use an RNA intermediate in a “copy and paste” mechanism, increasing their copy number in the genome.
- **Class II TEs (DNA transposons)** transpose directly via DNA in a “cut and paste” manner.

These classes are further subdivided into **orders**, **superfamilies**, **families**, and **clades**.

For a more in-depth overview of TE classification, the following articles provide valuable information:

- A unified classification system for eukaryotic transposable elements (Wicker et al., 2007)
- Twenty years of transposable element analysis in *Arabidopsis thaliana* genome (Quesneville, 2020)

This guide will walk you through the entire process of annotating Transposable Elements (TEs) in a genome using **EDTA**, comparing the outputs, and refining the TE classification with **TEsorter**. You will also gain insights into post-processing the outputs.

Exercise Overview

1. **TE Annotation** using **EDTA**.
2. **Visualizing and comparing TE annotations** from **EDTA**.
3. **Refining TE classifications** using **TEsorter**.
4. **Generate and Visualizing TE dynamics**.

Cluster Refreshers: Running tools on the IBU SLURM cluster

Use this copy-paste SBATCH template to run any containerised tool (EDTA, TEsorter, custom scripts) on the IBU cluster.

Save as 01-run_tool.sh in a scripts directory and run with `sbatch scripts/01-run_tool.sh`

> **Note:**
>
> Always create a `logs` directory in your working directory to save the output and error logs from the SLURM job. Always number your scripts to keep track of the order of execution.

```
#!/bin/bash
#SBATCH --job-name=JOB_NAME             # e.g. TE_annotation
#SBATCH --partition=pibu_el8            # IBU cluster partition
#SBATCH --cpus-per-task=NCPUS           # e.g. 20
#SBATCH --mem=MEM                       # e.g. 64G or 200G
#SBATCH --time=TIME                     # format D-HH:MM, e.g. 2-00:00
#SBATCH --output=logs/%x_%j.out
#SBATCH --error=logs/%x_%j.err


# User-editable variables
WORKDIR=/path/to/your/workdir
CONTAINER=/path/to/TOOL_container.sif    # e.g. /data/.../EDTA2.2.sif
INPUT_FASTA=$WORKDIR/data/assemblies/assembly.fasta
OUTDIR=$WORKDIR/results/TOOL_run
TOOL_CMD="TOOL_COMMAND --input $INPUT_FASTA --other-options"  # replace
with actual command inside container

mkdir -p "$OUTDIR"
cd "$OUTDIR"


# Full run: runs TOOL_CMD inside the container using allocated CPUs
apptainer exec --bind "$WORKDIR" \
"$CONTAINER" \
"$TOOL_CMD --threads \$SLURM_CPUS_PER_TASK"
```

---

1. TE Annotation using EDTA

EDTA (Extensive de novo TE Annotator) is a powerful pipeline designed for the annotation of both structurally intact and fragmented transposable elements (TEs). It generates a non-redundant TE library, classifies the elements into superfamilies, and outputs detailed annotations for whole-genome TE studies. You can find more details in the official EDTA repository and its published paper:

- GitHub: EDTA
- Paper: EDTA Publication

Steps to Run EDTA

1. **Create a New Working Directory**
   Start by creating a new directory where all the EDTA output files will be saved.

```
WORKDIR=/path/to/your/new/working/directory
mkdir -p $WORKDIR/results/EDTA_annotation
cd $WORKDIR/results/EDTA_annotation
```

2. **Run EDTA**

   We provide a singularity/Apptainer container for EDTA version 2.2, which is available at: `/data/courses/assembly-annotation-course/CDS_annotation/containers/EDTA2.2.sif`

```
apptainer exec \
--bind $WORKDIR
/data/courses/assembly-annotation-
course/CDS_annotation/containers/EDTA2.2.sif \
EDTA.pl --help
```

   Here's an example command to run EDTA:

```
apptainer exec \
--bind $WORKDIR \
/data/courses/assembly-annotation-
course/CDS_annotation/containers/EDTA2.2.sif \
EDTA.pl \
--genome $WORKDIR/data/assemblies/assembly.fasta \
--species others \
--step all \
--sensitive 1 \
--cds "/data/courses/assembly-annotation-
course/CDS_annotation/data/TAIR10_cds_20110103_representative_gene_mod
el_updated" \
```

```
--anno 1 \
--threads 20
```

> **Note:**
>
> TE annotation tools like EDTA can be computationally demanding and time consuming. To maximize efficiency, it's a good idea to launch these jobs in the early half of the day. This allows you to start the annotation in the morning and have the results ready for analysis by the second half of the day. In your `sbatch` script, request atleast 20 cpus and sufficient time (e.g., 24-48 hours) to ensure the job completes successfully.

Pro Tips for Running EDTA

1. **Always use the `--anno 1` option**:
   By default, EDTA does not perform a genome-wide TE annotation. Enabling this option ensures that the annotated TE sequences are output to the `.gff3` file, which contains both intact and fragmented elements.

2. **Always use the `--cds` option**:
   To ensure that gene sequences are not misclassified as TEs, you should use the `--cds` option with the coding sequences (CDS) file. We recommend using the *Arabidopsis thaliana* CDS file located at: `/data/courses/assembly-annotation-course/CDS_annotation/data/TAIR10_cds_20110103_representative_gene_model_updated`

> **Important Output Files from EDTA**
>
> Upon completion, EDTA will generate several key files that are essential for understanding the transposable element landscape of your genome:
>
> 1. **Non-redundant TE Library:** `$genome.mod.EDTA.TElib.fa`
>   This FASTA file contains the transposable elements classified up to the superfamily level. The sequences are labeled using the three-letter naming system from Wicker et al. (2007), and each sequence represents a TE family.
>
> 2. **Whole-genome TE Annotation:** `$genome.mod.EDTA.TEanno.gff3`
>   This GFF3 file contains both intact and fragmented TE annotations across the entire genome.
>
> 3. **Intact TE Annotation:** `$genome.mod.EDTA.intact.gff3`
>   This file lists only the structurally intact TE annotations. This can be useful for focusing on recently active or well-conserved TEs.
>
> 4. **Summary of Whole-genome TE Annotation:** `$genome.mod.EDTA.TEanno.sum`
>   This file provides a summary of the annotated TEs, including counts and number of base pairs in the genome per superfamily and family.
>
> 5. **RepeatMasker Output:** `$genome.mod.EDTA.anno/$genome.mod.out`
>   This output from RepeatMasker contains detailed information on TE copies, including their percentage of diversity compared to the reference sequences in the TE library. The

> percentage of diversity can be used to estimate the age of TE insertions, which is crucial for understanding TE evolutionary dynamics.

Intact full length LTR-RTs

While EDTA is still running, you can start visualizing the TE annotations from Step 1, focusing on the structure-based annotation of full-length LTR retrotransposons (LTR-RTs).

Percent identity is a measure of similarity between two DNA sequences. In the case of Long Terminal Repeats (LTRs) of retrotransposons, it reflects the evolutionary divergence between the two flanking sequences. Initially, LTR retrotransposons contain two identical LTRs surrounding an internal coding region. Over time, these LTRs accumulate mutations, and as a result, the percent identity between them decreases. Thus, by analyzing the percent identity of LTR sequences within a retrotransposon, we can infer the relative age of the element and the time since their insertion.

Steps:

1. **Extract Percent identity of two LTRs from full length LTR-RTs**: Use the file `assembly.fasta.mod.EDTA.raw/assembly.fasta.mod.LTR.intact.raw.gff3`
2. **Refine TE classification for full length LTR-RTs and split them into Clades**: Generate the clade classification file using TEsorter. Use the Apptainer container for TEsorter located at `/data/courses/assembly-annotation-course/CDS_annotation/containers/TEsorter_1.3.0.sif`

```
apptainer exec --bind $WORKDIR /data/courses/assembly-annotation-
course/CDS_annotation/containers/TEsorter_1.3.0.sif TEsorter
assembly.fasta.mod.EDTA.raw/assembly.fasta.mod.LTR.raw.fa -db rexdb-plant
```

3. **Plot the number of LTR-RTs in each clade with their corresponding percent identity**: Plot the histogram of the number of LTR-RTs in each clade with their corresponding percent identity.

Follow the instructions in the script /data/courses/assembly-annotation-course/CDS_annotation/scripts/02-full_length_LTRs_identity.R

> **Guiding questions:**
>
> - Are there differences in the number of full length LTR-RTs between the clades?
> - Are there any clades with High percent identity (e.g., 99-100%) or young insertions and Low percent identity (e.g., 80-90%) or old insertions?

---

2. Visualizing and comparing TE annotations from EDTA

Once the TE annotation tool has finished running, you can visualize and compare the TE annotations from EDTA to gain insights into the transposable element landscape of the accessions.

2.1 Compare the TE Annotations summary file from EDTA

The summary file "$genome.mod.EDTA.TEanno.sum" generated by EDTA provides a quick overview of the TE content in the genome.

Report the number of base pairs and the percent of the genome occupied by each superfamily. Compare the TE superfamilies and their abundance across different accessions with other group members to identify similarities and differences in the TE annotations.

> **Guiding questions:**
>
> - Which TE superfamily is the most abundant in the genome?
> - Are there any differences in TE content between the accessions?

2.2 Visualizing TE Annotations

You can use either circos (https://circos.ca) or R package like circlize (https://jokergoo.github.io/circlize/) to generate plots that show the distribution of TEs across the genome. If your genome is not chromosome-level, you can use the top longest scaffolds as pseudo-chromosomes for visualization.

To create plots for different TE superfamilies using the `circlize` package in R, follow these steps:

1. Load the `circlize` library.
2. Read the TE annotation data from the GFF3 file: `assembly.fasta.mod.EDTA.TEanno.gff3`
3. Filter the TE annotation data to select the most abundant TE superfamilies.
4. Create a custom ideogram data from a the fai index file of the assembly fasta. The data should include the scaffold names, start, and end positions, where the start position is 0 and the end position is the scaffold length.

Output of `samtools faidx assembly.fasta` has the length of scaffolds in the second column.

6. Initialize the circos plot with the custom ideogram using the `circos.genomicInitialize` function.
7. Plot the TE density for each superfamily using the `circos.genomicDensity` function giving colour `col`, `track.height` and `window.size` options.

> **Guiding questions:**
>
> - Are there any regions with high TE density?
> - Do the distribution of Gypsy and Copia and other TIR DNA transposons overlap, are there any differences?

---

3. Refining TE Classification with TEsorter

TEsorter Overview

TEsorter (Zhang et al., 2022) is a tool designed to classify transposable elements (TEs) based on their protein domains. This method relies on homology-based detection to assign TEs into categories, providing detailed insight into the composition of TEs in a genome.

TEsorter focuses on **Class I TEs**, specifically performing clade-level classification of **LTR retrotransposons**, such as the **Copia** and **Gypsy** superfamilies.

The Python code for TEsorter is available here: TEsorter GitHub.

Step 1: Extract Copia and Gypsy Sequences

Start by extracting the Copia and Gypsy sequences from the EDTA-generated TE library (`$genome.mod.EDTA.TElib.fa`). These sequences will be used as input files for TEsorter.

Use `seqkit grep` from the **SeqKit toolkit** to subset specific sequences from the fasta file.

Example command to extract Copia and Gypsy sequences:

```
# Extract Copia sequences
seqkit grep -r -p "Copia" $genome.mod.EDTA.TElib.fa > Copia_sequences.fa

# Extract Gypsy sequences
seqkit grep -r -p "Gypsy" $genome.mod.EDTA.TElib.fa > Gypsy_sequences.fa
```

Step 2: Run TEsorter

Run **TEsorter** on each input file (Copia and Gypsy) using Apptainer. For this example, we are using the **TEsorter_1.3.0.sif** container located in `/data/courses/assembly-annotation-course/CDS_annotation/containers`.

```
apptainer exec --bind $WORKDIR /data/courses/assembly-annotation-
course/CDS_annotation/containers/TEsorter_1.3.0.sif TEsorter
Copia_sequences.fa -db rexdb-plant

apptainer exec --bind $WORKDIR /data/courses/assembly-annotation-
course/CDS_annotation/containers/TEsorter_1.3.0.sif TEsorter
Gypsy_sequences.fa -db rexdb-plant
```

Important Output Files

1. **`$.liban.rexdb-plant.dom.faa`**:
   Contains annotated protein sequences. These sequences can be used for downstream analyses like TE phylogenetic analysis.

2. `$.liban.rexdb-plant.cls.tsv`:
   Contains the classification of transposable elements into their respective classes, orders, and families. This file is key for assessing TE diversity within the genome.

Now, you can integrate the refined classifications from TEsorter on all TEs back into your TE annotation files for a more detailed understanding of the TE landscape in your genome.

> **Guiding questions:**
>
> Using the final TEanno.gff3 file from EDTA and the clade classification of all TEs from TEsorter can you provide an estimate of the number of Copia and Gypsy elements in each clade? What are the most abundant clades in your genome?

---

4. Dynamics of Transposable Elements (TEs)

Dating of TEs / Estimation of Insertion Age

The age of a transposable element (TE), or the time of its insertion, can be estimated by calculating the divergence of its sequence from a consensus (or most representative) sequence. This estimation relies on the assumption that copies of the same TE family originated from a single active "mother" element. At the time of insertion, all copies are identical, and over time, they accumulate mutations. The more divergent a copy becomes from the consensus sequence, the older it is presumed to be.

TE Age Estimation Workflow

To estimate the insertion age of TEs, follow these steps:

1. **Parse RepeatMasker Outputs:**
    - Use the Perl script `parseRM.pl` (available at https://github.com/4ureliek/Parsing-RepeatMasker-Outputs) to process the raw alignment outputs from **RepeatMasker**.
    - This script calculates the corrected percentage of divergence of each TE copy from its consensus sequence. It accounts for the extremely high mutation rate at CpG sites, providing a more accurate estimate of divergence. Sequences with low divergence indicate recent insertions, while sequences with high divergence suggest older insertions.
    - For further reading, refer to this study.
2. **Binning TE Divergence Data:**
    - Perform the following steps for the Repeatmasker file `$genome.mod.out`:

```
module add BioPerl/1.7.8-GCCcore-10.3.0
perl parseRM.pl -i $genome.mod.out -l 50,1 -v
```

    - Pro Tip: You can find the input file `$genome.mod.out` in the **EDTA** output directory, typically located at `$genome.mod.EDTA.anno`.
3. **Generate a TE Landscape Plot:**
    - Use the R script `plot_div.R` (available at `/data/courses/assembly-annotation-course/CDS_annotation/scripts`) to:
        - Visualize the divergence of TEs across the genome.
        - Generate a landscape graph, showing TE divergence and abundance for each superfamily.
    - This landscape can provide insights into the dynamics and evolutionary history of transposable elements within the genome.
4. **Dating TEs Using Substitution Rates:**
    - Date the divergence of the annotated TEs from their consensus sequence using the **synonymous substitution rate** for **Brassicaceae species**. This rate is estimated at **8.22 ×**

10^−9 substitutions per synonymous site per year (Kagale et al., 2014).

- The insertion time (T) can be calculated using theformula T = K/2r formula, where K is the sequence divergence, and r is the substitution rate.

> **Guiding questions:**
>
> - Can you identify recent and ancient TE activity peaks?
> - Are there differences in TE dynamics between the accessions studied in the group?
> - How do the TE dynamics differ between Copia and Gypsy elements?
