TErex1.0

Arya Kaul
Jigar Patel
Lowan Kim
Jenny Lee

*TE stands for transposable elements

I. BIOLOGICAL MOTIVATION

Objective

Given an input query, TErex1.0 identifies all the transposable elements within the sequence through an alignment method that uses the Dfam database.

Significance

Accurate transposon annotation facilitates biological research regarding mutations and evolutionary processes. Transposons (otherwise known as transposable elements, or TEs) are elements that move, or "jump around," from one position to another within the genome, and are found abundantly in nearly all prokaryotes and eukaryotes[1]. As a consequence of their ability to relocate, TEs can potentially lead to deleterious mutations; they have recently been found in cancer-related genes, which suggest a role in disease development[2]. TEs also drive genomic evolution and gene regulation by enabling translocation, exon shuffling, and increased diversity, among its many effects[3]. Their genetic influence makes transposons important elements to identify and study.

Applications

You can use TErex1.0 to:

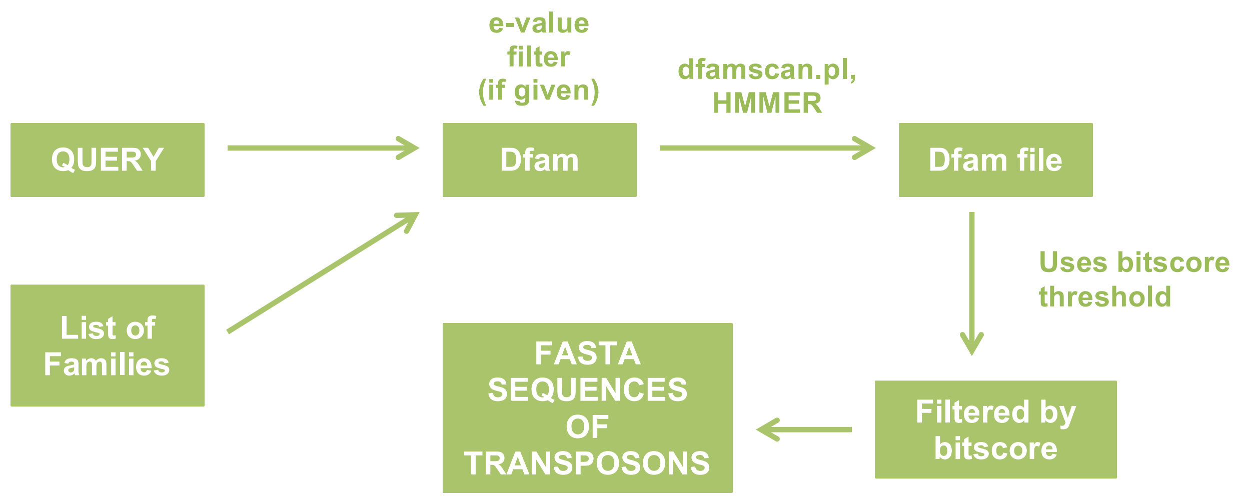
- **Study genomic evolution!** – Use TErex1.0 to identify TEs over a few generations of an organism. Check for patterns or conservations of TEs. How do transposons affect the evolutionary process of a genome?
- **Study diseases!** – Search for TEs in disease-related genes. Screen a healthy and unhealthy organism (control and experiment) to see if any TEs are related to disease development.
- **Study specific TEs!** – Looking to study specific TEs? The Alu sequence, for example, is highly conserved in primates. It contributes to genetic diversity, but also causes disease through insertional mutagenesis[4]. If you're only interested in studying this element, use TErex to filter for only the Alu family in your query!

---

[1] Pray, L. A. Transposons: The Jumping Genes. *Nature Education* **1** (2008).
[2] Miki, Y., *et al*. Disruption of the *APC* gene by a Retrotransposal Insertion of L1 Sequence in a Colon Cancer. *Cancer Research* (1992).
[3] Kazazian, H. H. Mobile Elements: Drivers of Genome Evolution. *Science* **303**, 1626-1632 (2004).
[4] Deininger, P. *Alu* elements: know the SINEs. *Genome Biology* **12** (2011).

II. TErex1.0 DETAILS

Pipeline Overview



Step-By-Step Walkthrough

Command Line – main.py

- Input: 1) path of FASTA file and 2) name of the output directory
  - Option to read in a list of families they want to compare their query to
  - Option to set bitscore threshold for filtered output sequences
  - Option to set the e-value threshold to further filter the output sequences

dfamscan[5]

- Input: command line arguments
  - Runs dfamscan.pl
    - Uses e-value threshold to filter data
    - dfamscan takes user query and compares it to the Dfam database
- Output: dfamRaw.out file

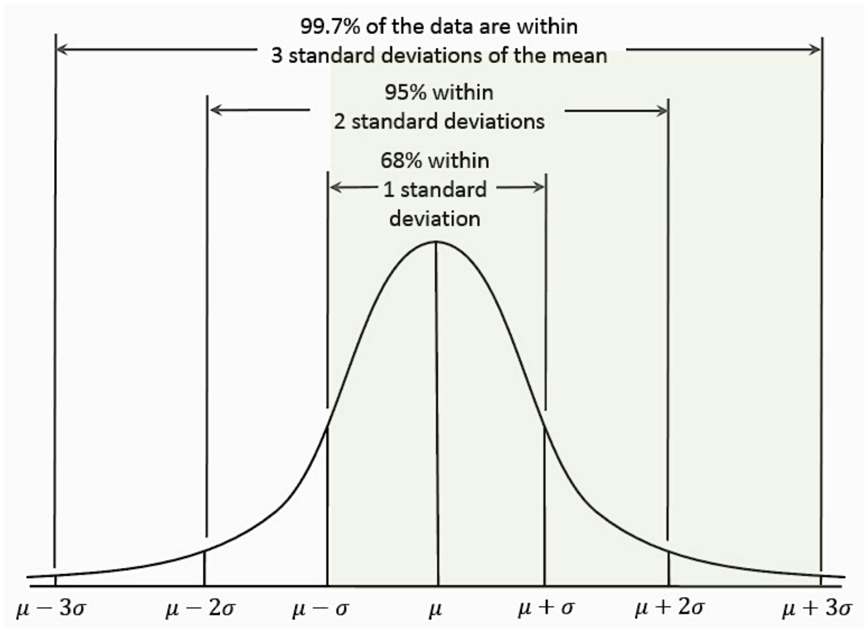
[5] Hubley, R., *et al*. The Dfam database of repetitive DNA families. *Nucleic Acids Research* **44**, D81-D89 (2015).

Family List Filter (if provided at command line) – familyScreener.py

- Input: dfamRaw.out
    - Removes families that aren't part of the user-specified list
- Output: clean version of dfamRaw.out file with undesired families removed (completed dfamscan, family filtered file)

Bitscore Threshold Filter – bitscore.py

- Input: family filtered file
    - Removes all transposons below the bitscore threshold
        - Default bitscore threshold is generated if user did not input a bitscore at command line



If a bitscore value is not provided at the command line, then a default bitscore is calculated. We generate the default threshold value by subtracting one standard deviation from the average of all bitscore values. This way, we discard most of the outliers and retain TEs with the top 84% bitscores.

- Output: bitscore filtered file

Sequence Generator – getFasta.py

- Input: bitscore filtered file
    - Generates the sequences of the transposons using the start and end positions and + or – values (forward or reverse)
- Output: FASTA file of transposon elements

TErex1.0 Implementation

TErex1.0 requires the following dependencies:

- Python2.7
- Python package Numpy
- nhmmer through hmmer3.1
- perl

If the user would like to download the Dfam database locally:

```
http://dfam.org/web_download/Current_Release/Dfam.hmm.gz
```

TErex1.0 runs a main python script that imports sub python scripts. It runs a perl script called dfamscan, which uses HMMER3, to generate a file of all identified TEs. We use dfamscan's e-value filter if the user provides us with an e-value threshold. Subsequently, the file of TEs is filtered by the bitscore threshold, of which the user also has the option of inputting at the command line. If no bitscore threshold is provided, then a default value is determined in a process that requires the Numpy python package. TErex1.0 then uses this filtered file to output a FASTA file with all the sequences.

Team Attributions

Python Script	Written By
main.py	Arya Kaul
familyScreener.py	Arya Kaul
bitscore.py	Jigar Patel, Jenny Lee
getFasta.py	Lowan Kim

III. TErex1.0 RESULTS

Using TErex1.0

On command line, run:

```
python main.py -h
```

The following help message will display:

```
usage: main.py [-h] [--fastafile FASTAFILE] [--dfamPath DFAMPATH]
               [--outputPath OUTPUTPATH] [--families FAMILIES]
               [--bitscoreThreshold BITSCORETHRESHOLD]
               [--evalueThreshold EVALUETHRESHOLD] [--tester]

Check the help flag

optional arguments:
  -h, --help            show this help message and exit
  --fastafile FASTAFILE
                        Path to the fasta file to run TErex on
  --dfamPath DFAMPATH   Path to the Dfam database that would like to be used
  --outputPath OUTPUTPATH
                        Path that you would like the outputted files to be
                        outputted to
  --families FAMILIES   Path to the text file with list of families that you
                        want filtered. If no option is chosen, every family
                        will be displayed. Format of the file should be a line
                        of family names separated by commas. Example file is
                        provided in TErex1.0/test/family_example
  --bitscoreThreshold BITSCORETHRESHOLD
                        Input the value of the SMALLEST bitscore you would
                        want to obtain. i.e. if a value of 30 is chosen, every
                        bitscore below 30 will be thrown out. If no value is
                        chosen, our default method which chooses every
                        bitscore >= mean - onestandard deviation
  --evalueThreshold EVALUETHRESHOLD
                        Input the value of the SMALLEST evalue you would
                        accept. i.e. if an evalue of 1e-30 is chosen, every
                        e-value below 1e-30 will be thrown out (this is WAY
                        too stringent). If no value is specified an evalue of
                        0.01 is chosen
  --tester              Test that everything is working fine. The test method
                        will be to run: python main.py ./Test/test.fa
                        ./Test/DfamSubset.hmm ./Test
```

TErex1.0 Performance

Questions about accuracy and precision are not applicable in the context of our tool, as our objective was to streamline the dfamscan process. Any flaws within our output are a result of the shortcomings of Dfam.

Advantages	Disadvantages
• Run TErex1.0 locally • No need to install and run programs individually • Filters and extracts the transposon sequences for the user • Friendly user interface • Cute dinosaur	• Realistic queries on Dfam (i.e. sequence of a chromosome) takes a long time to run on the average student laptop • Default thresholds may be too stringent

Conclusion

TErex1.0 takes in a query and an optional list of families, and aligns the query sequence to the Dfam database in order to identify all the TEs. We firmly believe that TErex1.0 provides a convenient and useful extension to the Dfam functionality.

[https://github.com/a1kaul/TErex1.0]

```
           boing         boing         boing
 e-e           . - .         . - .         . - .
(\_/)\       '       `.   ,'       `.   ,'       .
 `-'\ `--.___,         . .           . .          .
    '\( ,_.-'
       \\               "             "            a:f
       ^'
```