Digital image processing using α-molecules to detect adaptive evolution

Mahmudul Hasan, Md Ruhul Amin, Michael DeGiorgio

1 Introduction

α-DAWG is a framework to distinguish signs of selective sweep from genetic data. We use α-molecule transformation, such as wavelets, curvelets or a combination of both to extract information from the data to facilitate classification using machine learning. This software package can be used for applying α-DAWG to classify any genetic regions into sweeps and neutrals, i.e, regions showing signs of selective sweep and region without them. We will show how to process data in `.ms` or `.vcf` format and make to suitable for the model. 100 samples of each class for both training and testing have been provided to get started with an example, although the results demonstrated in the paper have been based on the model trained by 10000 samples for each class.

2 Downloads and requirements

To download the software, please go to `github_link_to_be_added`.

We will need `python` and `R` and `matlab` for different parts of the project.

For `python`, please install the following packages in a `conda` environment, `pandas`, `numpy`,`scipy`, `skimage`, `argparse`, `keras` and all relevant dependencies.

For `R` the required package, `wavelsim`, which we will use for the wavelet transform of the data.

For `matlab`, we will use the Curvelab package for curvelet transform.

3 Data Preprocessing: Simulated and Empirical

100 samples for each class are given in the `./Data/MS_files_train` and `./Data/MS_files_test` folders. Note that these are simulated data. To demonstate how to make α-DAWG works on empirical data, we have included data for one chromosome (chr22) in `.vcf` format from the CEU population in the `./Data/VCF` folder.

in `./Data/MS_files_train` and `./Data/MS_files_test` folder, the files are names as `neut_1.ms`, `neut_2.ms` and so on for neutrals and similarly `sweep_1.ms` and so on for sweeps.

To preprocess the `.ms` files, use the following format form of command

```
python3 MS_CSV.py <number of samples> <sweep or neutral> <train or test>
```

Use 1 for sweep sample and 0 for neutrals. And use 1 for test samples and 0 for train samples. For example,

```
python3 MS_CSV.py 100 1 1
```

would process 100 sweep `.ms` samples for testing. The resulting data are saved in their same respective train or test folders.

The above step converts the `.ms` files to `.csv` files. The `.csv` files further need to be sorted and resized so all of them are of the same shape. For this use the following command,

```
python3 Parsing_CSVs.py <number of samples> <sweep or neutral> <train or test>
```

Like before use 1 for sweep sample and 0 for neutrals, and use 1 for test samples and 0 for train samples. As an example

```
python3 Parsing_CSVs.py 100 1 1
```

will process 100 sweep samples for testing. The end products are matrices of 64×64. They are stored in the `./Data/CSV_files` folder.

Now we need to wavelet and curvelet transform them. For wavelet transform, run the following `R` command

```
Rscript Transform_Wavelet.R <wavalet level> <number of samples> <sweep or neutral> <train or test>
```

Wavelet level can not be higher that five, sicce our matrices are 64×64. For example,

```
Rscript Transform_Wavelet.R 1 100 1 1
```

will transform the 100 sweep test samples (output from `Parsing_CSVs.py`). The output will be a 100×4096 matrix named `Wavelets_sweep_test_.csv` in `./Data`. Other names will follow the same obvious convention.

For curvelet transform we use the Curvelab package in `Matlab`. Please Download Curvelab and copy *only* the `fdct_wrapping_matlab` folder in the parent directory, this is the only folder we will use. You will also need to copy the included `Transform_curvelet.m` in the new imported `fdct_wrapping_matlab` folder.

Now, open Matlab from terminal at the `.\fdct_wrapping_matlab` directory. For example, in macOS typing,

```
/Applications/MATLAB_R2022b.app/bin/matlab  -nodesktop
```

should open Matlab. Now the following command should perform the curvelet transform,

```
Transform_Curvelet(<number of samples>), <sweep or neutral>, <test or train>)
```

The output will be collected in a matrix, stored in `./Data`. For example running

```
Transform_Curvelet(100, 1, 1)
```

will transform 100 sweep test samples and will output a matrix fo size 100×10521. Each 64×64 matrix yields a total of 10521 curvelet coefficients.

Emiprical data usually come in `.vcf` format, so we also discuss how to process `.vcf` files. A sample file `CEU22.vcf` has been included in `.\Data\VCF`, we discuss how to process this.

First, to convert the `.vcf` to `.ms` file, please use the following

```
python3 VCF_MS.py <File name (without the .vcf suffix)>
```

This will create the necessary amount of `.ms` files, they will be stored in `./Data/VCF/MS_files`. To convert these `.ms` files to `.csv` files, please do the following

```
python3 VCF_MS_CSV.py <number of samples>
```

The `.csv` files will be stored in `./Data/VCF/CSV/` folder. Now we need to parse them. This is done by the following command

```
python3 Parse_VCF.py <number of samples>
```

These are saved in `./Data/CSV_files`, with names `output_1.csv`, `output_2.csv` and so on.

To wavelet transform them, use

```
Rscript EMP_Transform_Wavelet.R <wavelet level> <number of samples>
```

the output will be a single matrix with ¡number of samples¿ of rows, named `EMP_Wavelets_.csv` in `./Data`.

For curvelet transforming the empirical data, another Matlab file `EMP_Transform_curvelet.m` is provided, please copy that in the `fdct_wrapping_matlab` folder. Now, open Matlab as described before and run

```
EMP_Transform_Curvelet(<number of samples>)
```

This will make a matrix with ¡number of samples¿ number of rows in `./Data`.

4 Model Testing

To run the madel, on a `python` environment, use the following command

```
python3 alpha_DAWG.py <wavelet level> <chromosome number>
```

For wavelet level please use the wavelet level you have already used for wavelet transform. The resulting probabilities from the model will be stored in `Probabilities of samples in simulated data.csv` for simulated data and `Probabilities of samples in <chromosome>.csv` for emipirical data.