Easy353 —— Short Manual

zzhen

zzhen0302@163.com

May 22,2022

1. About Easy353

Easy353 is a tool for recovering Angiosperms353 gene set (AGS), which can filter and de novo assemble reads from high throughput sequencing data (including genome skimming, RNA-seq, and target enrichment) based on reference sequences from Kew Tree of Life Explorer[1], helping users capture AGS accurately and effectively.

> The Angiosperms353 gene set (AGS) is a universal low-copy protein-coding nuclear gene set across all angiosperm plants for phylogeny reconstruction[2], which can provide phylogenetic information at multiple scales and has promising applications.

2. Easy353 and Easy353-GUI

Easy353 is an easy-to-use Python program and is designed to be compatible with versions higher than 3.6. And, **Easy353 is distributed with two user interfaces: a graphical user interface (Easy353-GUI) and a command-line interface (Easy353-cmd)**. It's recommended that users with laptops or desktop computers could use the Easy353-GUI, which can be run on Windows and macOS without installation.

3. Downloading Easy353

Easy353 is open source under **MIT**. It is distributed via plant720 github repository: https://github.com/plant720/Easy353 where you can always download the most up to date version. Make sure to watch the github repository to remain up to date regarding code changes.

> Version numbers follow the notation **x.y.z** where **x** changes with major code reorganizations, **y** changes when new features are added and **z** changes with bug fixes.

4. Installing Easy353

Easy353 comes in two different flavors: Easy353-cmd and Easy353-GUI. Thus, before installation you need to know on what kind of system you intend to execute Easy353. The Easy353-cmd can be run at all popular systems(Linux, macOS and Windows); while the Easy353-GUI is designed for Windows and macOS.

When working with small amounts of data on a Windows or Mac computer, we advice using the Easy353-GUI; larger amounts of data should be processed using the Easy353-cmd.

4.1 Get Easy353-GUI

You can download Easy353-GUI from https://github.com/plant720/Easy353/release, which can be run directly by double-clicking without installation.

4.2 Get Easy353-cmd

There are generally two ways to install Easy353-cmd:

- Option 1 **Using the setup.py**
- Option 2 **In situ configuration**

Option 1. Using the setup.py

You should use git to download the entire Easy353 repository and install the Easy353 using the setup.py.

```
# get a local copy of easy353 source code
git clone https://github.com/plant720/Easy353.git
# install the code
cd Easy353
python setup.py install --user
```

Using the setup.py, **you should have Python library setuptools installed** (`sudo apt install -y python-setuptools` or `sudo yum install -y python-setuptools` or `pip install setuptools`). Generally, if you have Python installed, you should skip this step.

For some Linux nad macOS systems, after above commands you still cannot execute `build_database.py` and `easy353.py` in a new terminal directly, meaning `~/.local/bin` was not added to the $PATH, you have to manually add `~/.local/bin` :

```
# add ~/.local/bin to PATH
echo "export PATH=~/.local/bin:\$PATH" >> ~/.bashrc
source ~/.bashrc
```

Option 2. In situ configuration

Use git to download the entire Easy353 repository.

```
# Supposing you are going to install it at ~/Applications
mkdir ~/Applications # create directories if not existed
cd ~/Applications
git clone https://github.com/plant720/Easy353.git
```

Use following commands to make Easy353 scripts executable

```
chmod 755 Easy353/build_database.py
chmod 755 Easy353/easy353.py
```

Add Easy353 to the $PATH.

```
echo "export PATH=~/Applications/Easy353:\$PATH" >> ~/.bashrc
source ~/.bashrc
```

At last, install python libraries **biopython,psutil, requests, and beautifulsoup4** using pip or conda.

```
# install required libs
pip install biopython psutil requests beautifulsoup4
```

5. Input File Format

5.1 Reads Files

Reads files are the data generated from High-throughput Sequencing (HTS). For Easy353, it could assemble any HTS reads, given in the **FASTQ** format. Paired or unpaired reads are OK. And Compressed files (`.gz`) are also available as input files.

```
@ST-E00600:58:HKYNGALXX:1:1101:1434:1000 1:N:0:GAGTTCGA
ATTGGGCACGACACGAAACGAAATTTTTGTTAACAGGAATTTAGAAGTAGATATTCAAGTTATGGTTGGTTATTAAGGTTATCTTATCAAAAGATAGAGCAAGATGGTTATGCTTTTTAAGGCTAAGGAAAATGTCAGCATCTACTTCAT
+
AAFFFJJJJJJJJJJJJJJJJJJJJJJJJJJJ7JJJJJJJJJJJJJJJJJJJJJJJJJ7JJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJ77JJJJJ7J7JJJ7JJJJJJJJ7JJJJJJ7JJJJJJJJJJJJ-JJJJJJ-JJJJJ-J7-JJ
```

5.2 Reference files

Reference files are **FASTA**-formatted text files that contain one or more DNA sequences (Angiosperms353 gene sequences). Those sequences are homologous genes from the closely related species of target species, which could be downloaded from Kew Tree of Life Explorer.

Here we provide an example of reference files: https://github.com/plant720/Easy353Test/tree/master/data:

> Each sequence's ID should contain the gene name and species name, separated by the `_`.

```
>Litchi_chinensis_6946
CAAACCAAAATACACAATATAGGGGCAACACTTGTTGGGGTTGATAAATTTGGTAACAAGTATTATGAGAAACTTGGAAGACATAGGTGGGTTGAATATGCAGAGAAAG
>Tristiropsis_sp._6946
GGAAGACATAGGTGGGTTGAATATGCAGAGAAAGGTCGCTACAATGCCTCTCAGGTGCCTCCGGAATGGCATGGTTGGCTTCACTTTATAACTGATCACACAGGAGATG
```

6. Parameters

One of most important command is the Easy353 help option that displays all options. You can get on-line help type: `easy353.py -h`, and you will get the following, very long listing, that will be discussed at length below:

```
usage: easy353.py [options]

Easy353 zzhen@sculab

optional arguments:
  -h, --help            show this help message and exit
  -1 FQ_FILE_1 [FQ_FILE_1 ...]
                        Input file(s) with forward paired-end reads
                        (*.fq/.gz/.tar.gz).
  -2 FQ_FILE_2 [FQ_FILE_2 ...]
                        Input file(s) with reverse paired-end reads
                        (*.fq/.gz/.tar.gz).
  -u UNPAIRED_FQ_FILE [UNPAIRED_FQ_FILE ...]
                        Input file(s) with unpaired (single-end) reads.
  -r REFERENCE          Input a file(directory) with references.
  -o OUTPUT_DIR         Output directory.
  -k1 FILTER_KMER       Kmer setting for filtering reads. Default:29
  -k2 ASSEMBLE_KMER     Kmer setting for assembling reads. Default:41
  -s STEP_LENGTH        Step length of the sliding window on the reads.
                        Default:1
  -t1 FILTER_THREAD     Threads setting for filtering reads. Default:1
  -t2 ASSEMBLE_THREAD   Threads setting for assembling reads. Default:4
  -kmer_limit KMER_LIMIT
                        Limit of kmer count. Default:8
  -f FUNCTION_MODE      0:all,1:filter,2:assemble. Default:0
  -min MINIMUM_LENGTH_RATIO
                        The minimum ratio of contig length to reference
                        average length. Default:1.0
  -max MAXIMUM_LENGTH_RATIO
                        The maximum ratio of contig length to reference
                        average length. Default:2.0
  -change_seed CHANGE_SEED
                        Times of changing seed. Default:32
  -reference_number REFERENCE_NUMBER
                        The number of the reference sequences used to build
                        hash table. Default:all
  -fast                 Whether to use fast mode.
```

6.1 Parameters to be specified

The following options need to be specified:

- `fq_file_1` and `fq_file_2`: The input files with paired-end reads, given in FASTQ format.
- `unpaired_fq_file`: The input file with unpaired-end reads.

> The `fq_file_1`,`fq_file_2` or `unpaired_fq_file` only need to be specified, depending on your data type.

- `reference`: A directory containing Angiosperms353 gene sequences from closely related species, given in FASTA format.
- `output_dir`: The output directory.

6.2 Parameters with default value

The following options have default values and do not need to be specified in general.

- `filter_kmer`: K-mer length setting for filtering, default value is 29. The `filter_kmer` is the length of the shared DNA segment between the reads and the reference sequences. Easy353 looks for a shared K-length DNA segment (K-mer) between a read and a reference to assess whether they are related. Consequently, when the value of `filter kmer` is set to 29, it means to search for reads that have 29bp in common with the reference sequence.
- `assemble_kmer`: K-mer length setting for assembly, default value is 41. The `assemble_kmer` is the length of the nodes in the de Bruijn graph (DBG). In reads assembly, the filtered reads are divided into K-mers; the k-mers are employed as the nodes of the DBG. It strongly depends on the input dataset. For Illumina reads(150bp) with sufficient coverage (> 40x), we have good results with k = 41.
- `kmer_limit`: Limit of K-mer count, default value is 8. The`kmer_limit` is used to remove erroneous, low-abundance K-mers. This parameter also strongly depends on the dataset. It corresponds to the smallest amount of times a correct k-mer appears in the reads. A typical value is 8, which means only K-mers with at least 9 occurrences will be used for assembly. If the dataset has high coverage, try larger values.
- `filter_thread`: Threads setting for reads filtering, defalut value is 1. This value is advised to ideally not exceed 4. And for Windows and macOS, the `filter thread` should be set to 1.
- `assemble_thread`: Threads setting for reads assembly, defalut value is 4.
- `reference_number`: The number of the reference sequences used to build hash table, and the full reference sequence will be used by default. When one target gene has more than 100 homologous reference sequences, we advise setting this `reference_number` to 100 since the amount of sequences in the fasta file will effect the memory size needed by Easy353.
- `fast`: The switch to control fast mode. If typing `-fast` in command line, Easy353 will run faster, but will use nearly twice as much memory.
- `step_length`: The length of the interval when splitting the reads into K-mer, default value is 1. With a sequence is AGTTACGTCA, when `step_length` is 1 and `kmer_size` is 5, we can get AGTTA, GTTAC, TTACG, TACGT, ACGTC and CGTCA; when `step_length`is 2, we will get AGTTA, TTACG, ACGTC. This parameter can be used to reduce the program runtime when the dataset is large with sufficient coverage.
- `change_seed`: The setting for the number of seed changes, default value is 32. Actually, `change seed` is the amount of times the assembly's beginning point can be changed. The seeds are high-abundance K-mers selected from filtered reads that serve as the beginning point for de novo assembly. When the assembled gene's length is less than the set value, Easy353 will alter the assembly beginning point.
- `minimum_length_ratio`: The minimum ratio of recovered gene's length to reference gene's average length, default value is 1.0. Easy353 will decide that the gene recovery has failed when the actual length ratio is less than this value.
- `maximum_length_ratio`: The maximum ratio of recovered gene's length to reference gene's average length, default value is 2.0. When the actual assembled gene's length is longer, Easy353

will use stronger limitations, such as higher `assemble_kmer` and `kmer_limit`.

7. Output

The output directory contains the `filtered_reads` and the `target_genes` .

- The directory, `filtered_reads`, contains filtered reads associated with target 353 genes.
- And the directory, `target_genes`, is the most important result directory of Easy353, which stores 353 gene sequences recovered by Easy353.
    - The files under `target_genes` are recovered successfully, and the `unrecovered_genes` directory contains 353 genes that were not recovered successfully, because the recovered genes are shorter in length.

8. Example

- Download the simulation data of *Glycine max*:

```
wget https://github.com/plant720/Easy353Test/raw/master/data/Gmax_sim_1.fastq.gz
wget https://github.com/plant720/Easy353Test/raw/master/data/Gmax_sim_2.fastq.gz
```

- After installation of Easy353 and downloading the simulation data of *Glycine max*, please download the AGS of related species as the reference

```
# Download the AGS data according to taxonomy: Glycine max is a species from Glycine genus in Fabaceae, so we download species from Fabaceae as the reference.
# The reference sequences are downloaded from https://treeoflife.kew.org/,so keep your devices connected to the network. And if you the build_database.py, please cite: https://doi.org/10.1093/sysbio/syab035.
build_database.py -o 353_ref_Fabaceae -c Fabaceae -t 10 -exclude Glycine_max -generate
# The final reference sequences can be found at 353_ref_Fabaceae/353gene after downloading

## Explanation of parameters
-o: the output directroy
-c: the taxonomy of species that used as reference
-t: the thread used to download files
-exclude: exclud species that are not used as reference
-generate: generate a csv that records the info of downloaded species
```

- Then do the recovery of Angiosperms353 gene set (AGS)

```
# use easy353.py to filter and assemble reads to get target genes
easy353.py -1 Gmax_sim_1.fastq.gz -2 Gmax_sim_2.fastq.gz -r 353_ref_Fabaceae/353gene -o test_package -k1 31 -k2 41 -t1 1 -t2 4 -reference_number 100
```

- Now, you can view the result of Easy353 in output directory

9. Recipes

To download the reference files of user_defined taxon. Generally, the user_defined taxon could be the genus/family of the target species.

```
build_database.py -o 353_ref -c user_defined_taxon -t 10 -generate
```

To recover the AGS of target species, typically I use:

```
easy353.py -1 forward.fq.gz -2 reverse.fq.gz -r 353_ref/353gene -o target_species_output
```

or with detail options:

```
easy353.py -1 forward.fq.gz -2 reverse.fq.gz -r 353_ref/353gene -o target_species_output -k1 29 -k2 41 -t1 1 -t2 4
```

or in a fast but not memory-economic way:

```
easy353.py -1 forward.fq.gz -2 reverse.fq.gz -r 353_ref/353gene -o target_species_output -k1 29 -k2 41 -t1 1 -t2 4 -fast
```

or with less reference files:

```
easy353.py -1 forward.fq.gz -2 reverse.fq.gz -r 353_ref/353gene -o target_species_output -k1 29 -k2 41 -t1 1 -t2 4 -reference_number 100
```

or with more recovered genes but short in length:

```
easy353.py -1 forward.fq.gz -2 reverse.fq.gz -r 353_ref/353gene -o target_species_output -k1 29 -k2 41 -t1 1 -t2 4 -min 0.5
# min is 0.5 means the length of coding sequences recovered was at least 50% of the target length
```

or with low rates of mutation and indel errors but short in length.

```
easy353.py -1 forward.fq.gz -2 reverse.fq.gz -r 353_ref/353gene -o
target_species_output -k1 29 -k2 41 -t1 1 -t2 4 -kmer_limit 16
```

10. Q&A

1. **What is the difference between the Easy353-cmd and Easy353-GUI?**

Easy353-gui is a user interface for Windows and macOS users that provides the same functionality as Easy353-cmd. The only difference is the default settings of the parameters, which may also be specified by users based on their needs.

2. **How much memory and time will be taken?**

Usually, Easy353 can recover the AGS of one species in less than 30 minutes, and memory utilization is less than 10 GB. The memory used by Easy353 is mainly affected by the `reference_number` , `filter_thread`and whether fast mode is used.

The `reference_number` is the number of the used reference sequences; all reference sequences will be broken into *k*-mers and then kept in a hash table; the hash table is the program's primary memory-using data.

The `filter_thread` is the number of thread used to filter reads. In Easy353, we use multiprocess to implement faster read filtering. In Windows and macOS, when we use multiprocess, the hash table will be directly copied, then the memory used will increase with the copy of hash table. For Linux, the memory pages are copy on write, so employing several processes does not significantly increase memory. Therefore, the defalut value of `filter_thread` is 1 and the value is advised to ideally not exceed 4. For Windows and macOS, the `filter thread` should be set to 1.

When you use `fast` mode, Easy353 will reverse and complement the reference sequences, and the original sequences and reverse-complemented sequences will be used to build hash table. The hash table will double in size and the memory used will be nearly twice.

So users with small RAM computers can further reduce memory consumption by limiting the value of `reference_number` ,`filter_thread`and not using `fast` mode.

3. **How should I set the `filter_kmer` and `assemble_kmer` :**

The `filter_kmer` and `assemble_kmer` strongly depends on the data. Generally, `filter_kmer` is set to 29 and `assemble_kmer` to 41, which was obtained from test on simulated data. The `filter_kmer` is mainly influenced by the similarity of the reference sequences, and we recommend that this value does not fall below 21. When setting `assemblr_kmer`, we need to consider the sequencing data size, and the value should not be less than 31.

4. **How should I set the `kmer_limit`:**

The `kmer_limit` is used to remove erroneous, low-abundance K-mers. This parameter also strongly depends on the dataset. It corresponds to the smallest amount of times a correct k-mer appears in the reads. A typical value is 8, which means only K-mers with at least 9 occurrences will be used for assembly. If the dataset has high coverage of the genome of target species, try larger values, and vice versa.

5. **What kind of data should I provide?**

Easy353 could recover AGS using genome skimming, transcriptome, and target enrichment sequencing data. While sufficient coverage of sequencing data is necessary, it is recommended that ~10x coverage of the genome of target species for sequencing data is optimal.

Reference

[1] Baker WJ, Bailey P, Barber V, Barker A, Bellot S, Bishop D, Botigué LR, Brewer G, Carruthers T, Clarkson JJ, et al. 2021. A Comprehensive Phylogenomic Platform for Exploring the Angiosperm Tree of Life. *Syst Biol.* 71:301-319.

[2] Johnson MG, Pokorny L, Dodsworth S, Botigue LR, Cowan RS, Devault A, Eiserhardt WL, Epitawalage N, Forest F, Kim JT. 2019. A universal probe set for targeted sequencing of 353 nuclear genes from any flowering plant designed using k-medoids clustering. *Syst Biol.* 68:594-606.