APPLE picking, a user guide

Ayelet Heimowitz, Joakim Andén and Amit Singer

Feburary, 2018

1 Downloading the APPLE picker

The current version of APPLE picker is available for users with MATLAB. It requires the Image Processing Toolbox to be installed. For best performance, we recommend running the code on a GPU using the Distributed Computing Toolbox, although the code will run fine without it. In the future we plan to release a Python version of the APPLE picker.

Download the package from

```
https://github.com/PrincetonUniversity/APPLEpicker
```

to some destination folder, *e.g.* applePicker. This is the folder from which the APPLE picker will run. This folder has the following structure

- data/ This folder contains the data used in the example script. To begin with, it only contains the hashes of the files in the dataset, but once the scripts are run, they will download the entire datasets into this folder.
- doc/ The documentation for the package.
- matlab/ The MATLAB source code. This contains all the functions needed to perform APPLE picking.
- results/ This directory will eventually contain the results of the APPLE picker. It is not present at install time but is created when the APPLE picker is run.
- LICENSE The package license.
- README A short README file for the package.
- setup.m A MATLAB setup script that initializes the environment and sets the necessary paths.

2 Getting Started

After downloading the package, start MATLAB in the package directory and run

```
setup;
```

This will initialize the environment to make sure the paths to the MATLAB files are correct.

To download some example datasets [1, 2, 3] and run the APPLE picker on them, run

```
getParticles;
```

Since the datasets are quite large, this may take some time. STAR files containing locations of the picked particles will be found in the results/ folder.

By default, the `getParticles` script will only perform particle picking on a single micrograph and will not show any images. This behavior can be modified by changing the `runAll` and `showImages` parameters in the script.

3 Running the APPLE picker

The function `ApplePicker` performs the particle picking. This function has 2 necessary inputs and 7 optional ones. The function is called as

```
ApplePicker(path, particleSize, 'property1', value1,
            'property2', value2, ... )
```

The inputs are described below.

- `path`

  This parameter contains the location of the micrographs to be processed. Assume all the micrographs are in the folder data/empiar10017. To pick particles from all the micrographs, we set the first input to

  ```
  'data/empiar10017/'.
  ```

  On the other hand, to pick particles from a single micrograph called Falcon_2012_06_12-14_33_35_0.mrc, we set it to

  ```
  'data/empiar10017/Falcon_2012_06_12-14_33_35_0.mrc'
  ```

- `particleSize`

  The second input into `ApplePicker` is an integer `particleSize` such that `particleSize`×`particleSize` is the size of a window containing the particle.

- Optional properties

  These are specified in the key-value form, `'property', value`, where `property` is one of the following:

  - `nOverlap`

    This optional input allows the user to specify the amount of overlap to be tolerated between two adjacent particles. The output of the APPLE picker is a .star file containing the center of a `particleSize`×`particleSize` box around each picked particle. The `nOverlap` property ensures that any two boxes of size `nOverlap`× `nOverlap` around those centers do not overlap. If an overlap exists, both particles are discarded.

    In order to set this property to, *e.g.* 62, call `ApplePicker` as in

    ```
    ApplePicker(path, particleSize, 'nOverlap', 62, ... )
    ```

    The default value for this property is

    $$\texttt{nOverlap} = \text{round}\left(\frac{\texttt{particleSize}}{10}\right).$$

  - `qSize`

    The APPLE picker assigns a score to query images according to the likelihood they contain a particle. This property determines the size of each query window. This size should be slightly smaller than the particle size. To set the size of the query images to, *e.g.*, 52×52, call

    ```
    ApplePicker(path, particleSize, 'qSize', 52, ... )
    ```

    The value specified for `qSize` should be divisible by 4. If it is not, the APPLE picker will choose the closest possible integer which is smaller that the specified qSize and divisible by 4.

    The default value for this property is

    $$\texttt{qSize} = 4 \times \lfloor\frac{t}{4}\rfloor, \qquad t = \text{round}\left(\frac{2 \times \texttt{particleSize}}{3}\right).$$

  - `classifierProps`

The APPLE picker uses a support vector machine classifier to determine the classification of each pixel in the micrograph. This feature allows to specify the size of the training set for the classifier. The input is a vector containing the number of particle examples and the number of noise examples to be used for classifier training. For example, in order to use 600 examples and 8000 noise examples, call

```
ApplePicker(path, particleSize, 'classifierProps', [600;
8000], ... )
```

The number of examples must be smaller than the number of query images. For a micrograph of size $N \times N$, there are approximately

$$2\left(\frac{N}{\text{qSize}}\right)^2$$

query images. The higher the concentration of particles in the micrograph, the more particle examples can be used.

The default values for the number of particle and noise examples is 5% and 30% of the query images, respectively.

– `containerSize`

This property depends on the size of the micrograph. For a micrograph of size $N \times N$, we suggest to aim for $300 - 400$ containers when setting this property.

To call `ApplePicker` with a chosen value L for `conainerSize`, call

```
ApplePicker(path, particleSize, 'containerSize', L, ... )
```

For a micrograph of size $M \times N$, the default value is

$$\texttt{containerSize} = \lfloor \frac{\min{(M, N)}}{18} \rfloor.$$

– `minParticle`

This property specifies the minimum diameter of a particle. The APPLE picker contains a classifier that makes a decision as to the classification of each pixel. Any cluster of particle pixels that would disappear if an erosion operation were done with a disk of diameter specified by minParticle will be discarded. In order to specify this minimal diameter to be, *e.g.*, 20, call

```
ApplePicker(path, particleSize, 'minParticle', 20, ... )
```

The default value for this property is `qSize`.

– `maxParticle`

  This property specifies the maximum diameter of a particle. Any cluster of particle pixels that would not disappear if an erosion operation were done with a disk of diameter specified by maxParticle will be discarded. In order to specify this maximal diameter to be, *e.g.*, 100, call

  ```
  ApplePicker(path, particleSize, 'maxParticle', 100, ... )
  ```

  The default value for this property is

  $$\texttt{maxParticle} = \frac{5}{3}\texttt{particleSize}.$$

– `showImages`

  In order to view images of the micrograph, the output of the classifier and the picked particles, set

  ```
  ApplePicker(path, particleSize, 'showImages', true, ... )
  ```

  By default, the APPLE picker does not present any images.

All of these optional parameters can be specified simultaneously by combining them in the call to `ApplePicker`.

3.1 Output and Examples

The output from the APPLE picker is a file containing the coordinates of the centers of each picked particle. The name of the output is the same as the name of the input file, but with .mrc replaced with .star. The output files are saved in the subdirectory results/.

Examples of running the APPLE picker on the β-galactosidase and KLH datasets are provided in the script `getParticles`.

3.2 Citation

If you use the APPLE picker, please cite our paper “APPLE Picker: Automatic Particle Picking, a Low-Effort Cryo-EM Framework”, available at https://arxiv.org/abs/1802.00469.

References

[1] Iudin, A., Korir, P., Salavert-Torres, J., Kleywegt, G., and Patwardhan, A. (2016). *EMPIAR: A public archive for raw electron microscopy image data*. Nature Methods, 13.

[2] Scheres, S. H. (2015). *Semi-automated selection of cryo-EM particles in RELION-1.3*. Journal of Structural Biology, 189, 114-122.

[3] Zhu, Y., Carragher, B., Mouche, F., and Potter, C. S. (2003). *Automatic particle detection through efficient hough transforms*. IEEE Transactions on Medical Imaging, 22, 1053-1062.