example

September 19, 2019

0.1 Demonstration the High-Resolution Load Profiling algorithm

import the class for profiling from HRLP.py

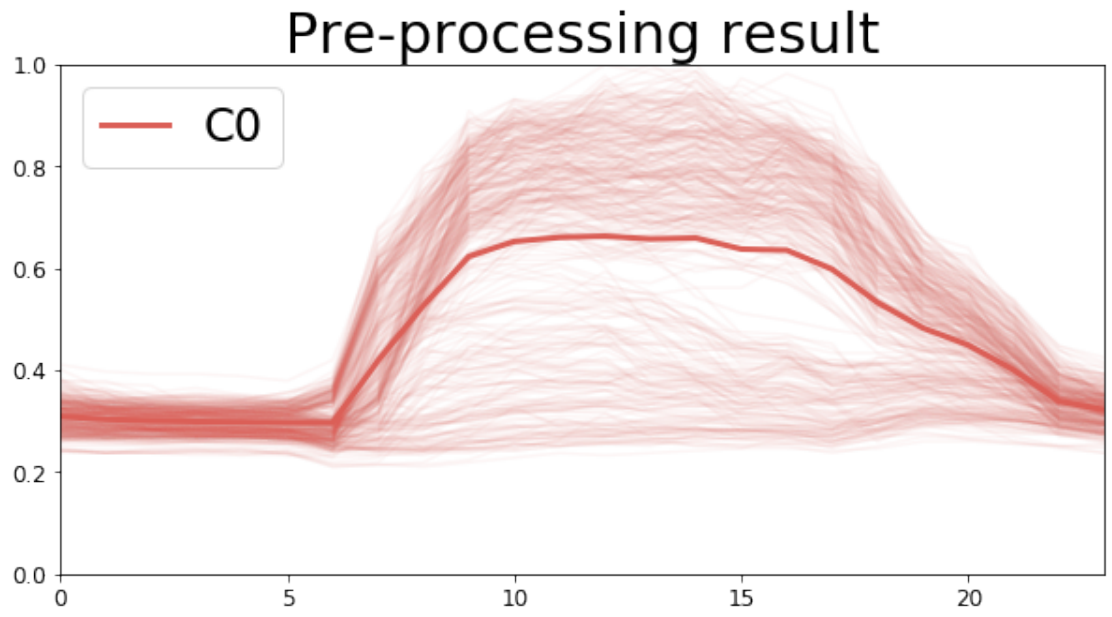
```
In [1]: # the third parties packages (pandas, numpy, sklearn, matplotlib, seaborn) are imported
        from HRLP import *
```

Initialization (preprocessing automatically done)

```
In [2]: # dir: the directory of the raw data; Pre: preprocess or not
        # the class expects a csv file with the first column as the timestamp
        # and the second column as the time series energy consumption data of a building
        test = Bldg2profile(dir = 'example.csv', Pre = True)

        # after initialization, the data can be visualized using the plot function
        # step: the result of which step to plot
        test.plot(step = 0)
```

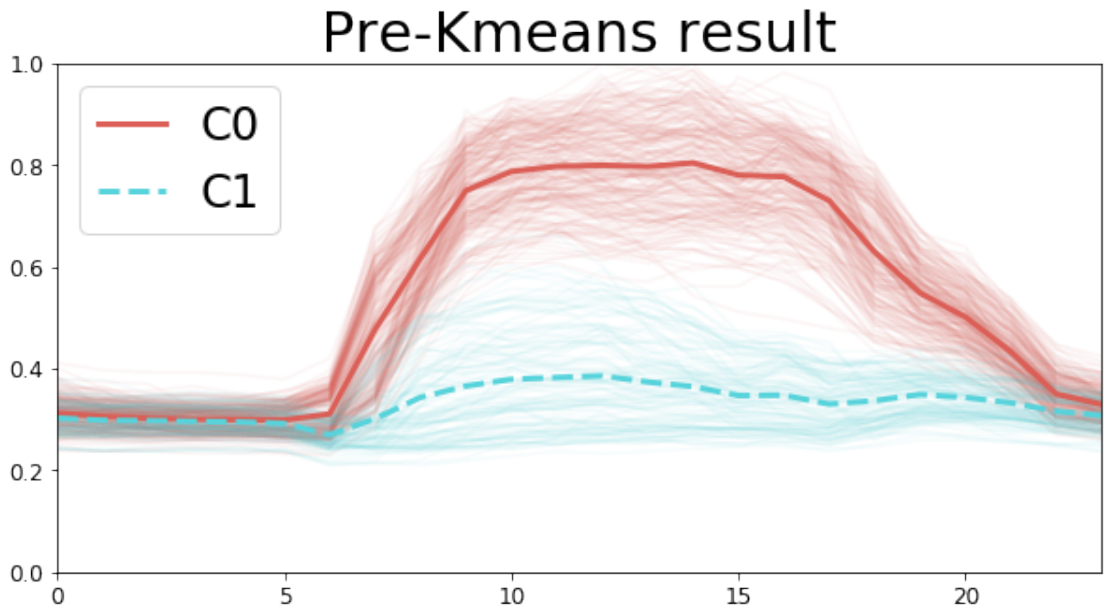
```
the building is metered from 2015-01-01 to 2015-12-31.
```



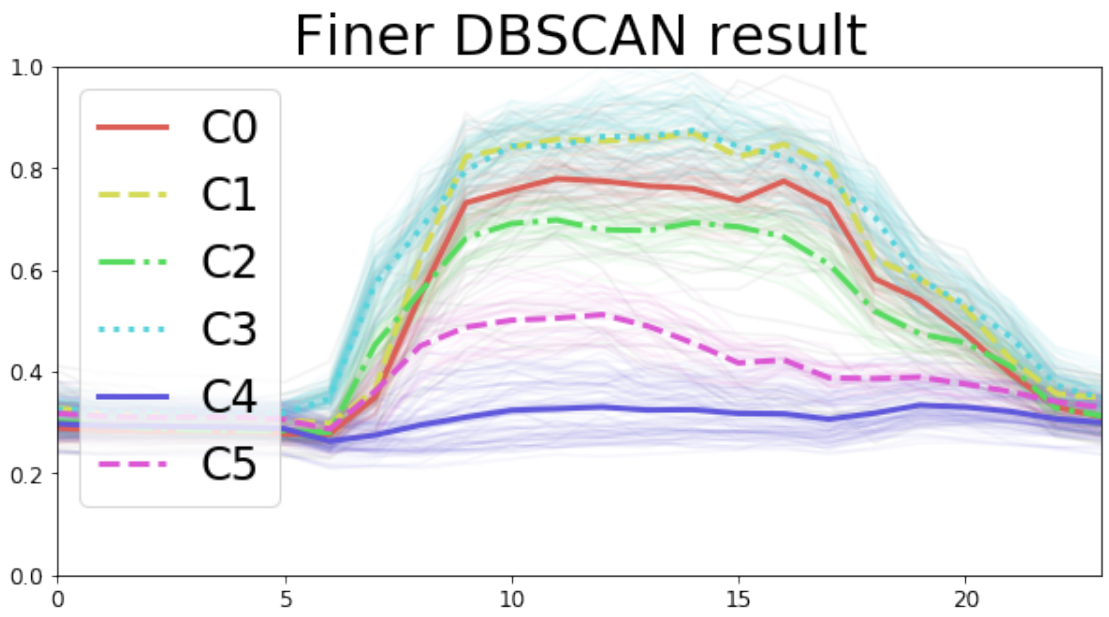
High-Resolution Load Profiling

In [3]:
```
# the main function to do high-resolution profiling
# after preprocessing, the profiling function consists of three main steps
# three argument indicating whether or not to show the intermediate results
test.profile(showKmeans = True, showDBSCAN = False, showFinal = False)
```

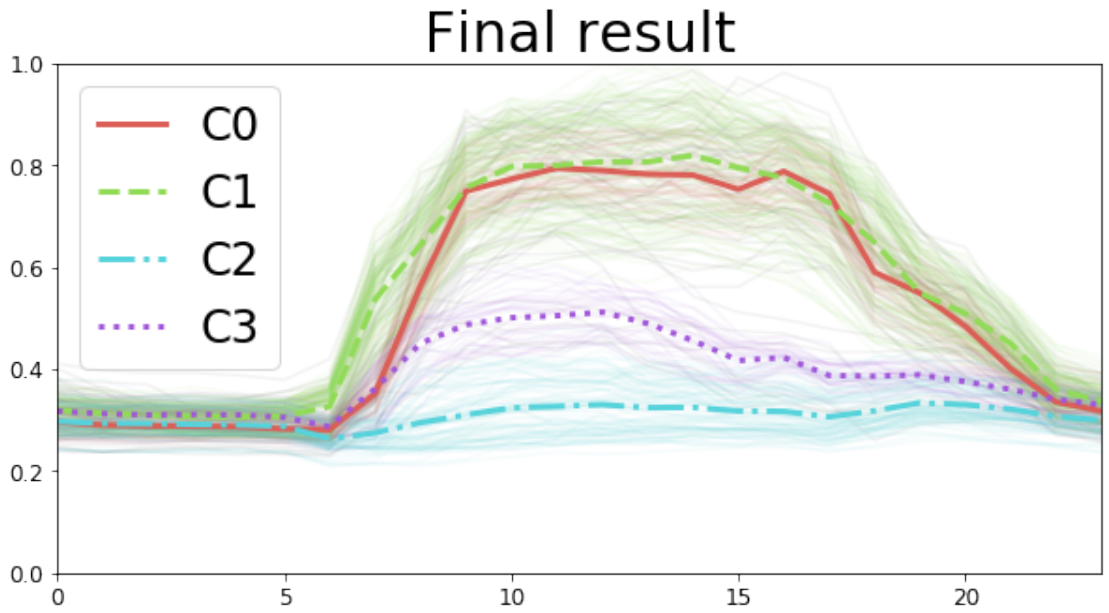
```
The preliminary K-means clustering resulted in 2 clusters.
```



In [4]:
```
# after profiling, the result can always be plotted using the plot function
test.plot(step = 2)
```

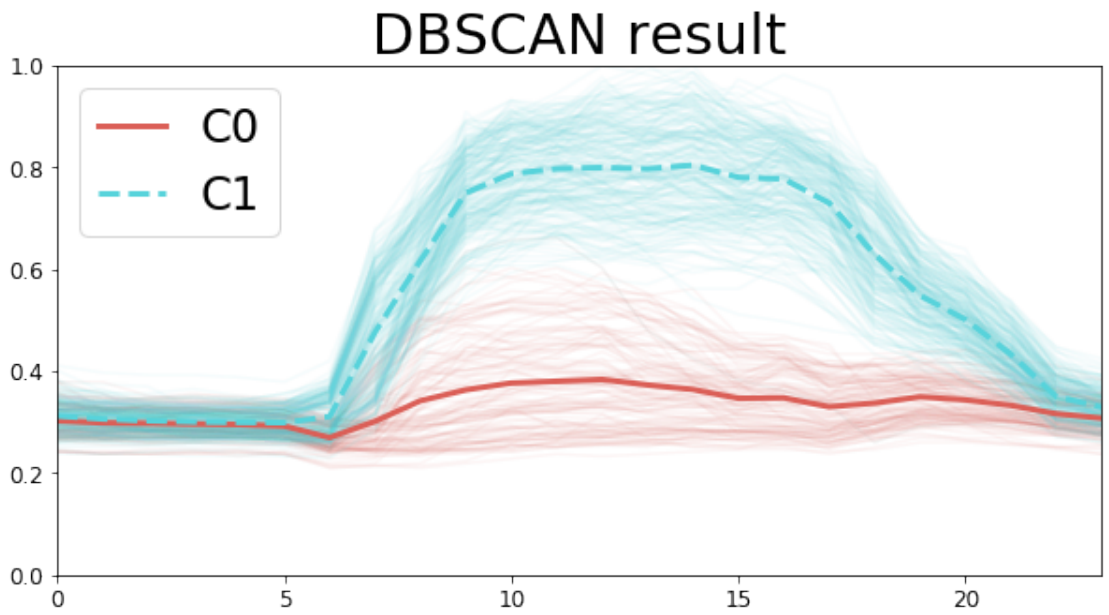


In [5]:
```
# after profiling, the result can always be plotted using the plot function
test.plot(step = 3)
```

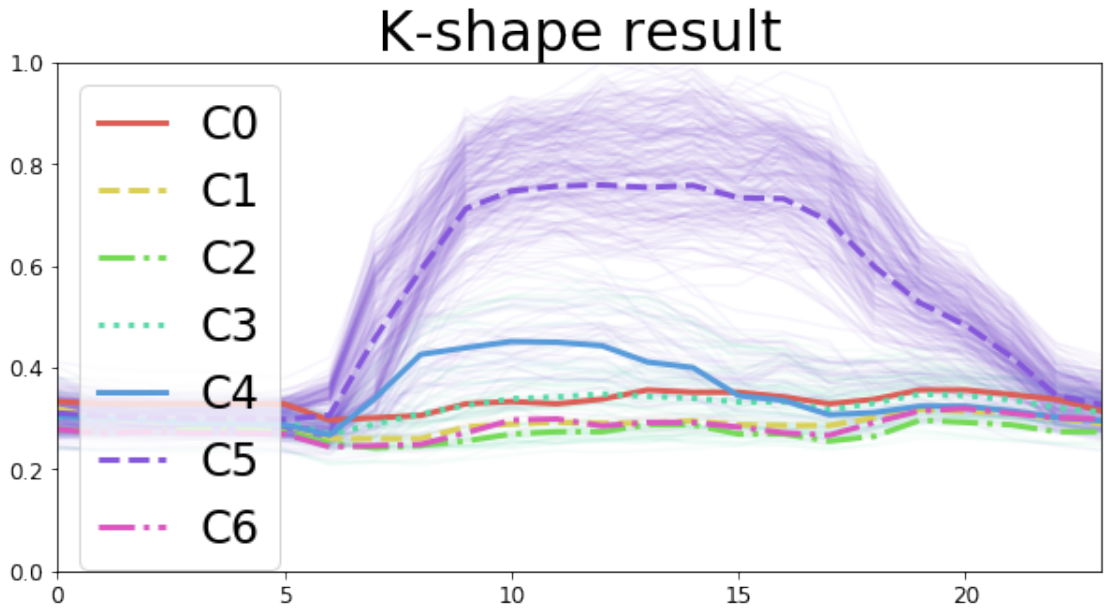


The profiling result using just K-means is plotted above as the intermediate result. Using the class methods, plot the result of other baseline methods

In [6]:
```
test.labelExt = np.asarray(test.DBSCAN(test.data))
test.plot(4,'DBSCAN')
```



In [7]:
```
from tslearn.clustering import KShape
# the iterative process of finding the optimal cluster number is skipped here
k1 = KShape(n_clusters=7,verbose=False, random_state=0).fit(np.asarray(test.data))
test.labelExt = np.asarray(k1.labels_)
test.plot(4,'K-shape')
```



In [8]:
```
from tslearn.clustering import TimeSeriesKMeans
k1 = TimeSeriesKMeans(n_clusters=2,verbose=False,metric='dtw').fit(np.asarray(test.data)
test.labelExt = np.asarray(k1.labels_)
test.plot(4,'dtw')
```