A prototype of feature-based *Kmeans*

Changfeng Liu

July 8, 2016

Abstract

In this report, I propose a new method to extract features from time series for clustering. In this feature space, the six classes of Synthetic Control[2] data set are located separately. With *Kmeans*, it can mostly distinguish the six classes. I introduce the design of feature vector, the experiment results and some potential future work in this report.

Contents

1 Introduction

Generally, there are three kinds of time series clustering: raw-data-based, feature-based and model-based[1]. In this report, I use feature-based *K-means* to cluster time series.

The basic idea is to extract several features which are able to distinguish time series with different frequent patterns. In other words, in this feature space, time series with different patterns are far from each other while time series with similar patterns are close.

I implemented this algorithm and tried it on "Synthetic Control Chart Time Series"[2] data set.

2 Feature Vector Design

I designed a three-dimension feature vector in this very first version, namely for seasonality, global trend and shift detection.

2.1 D1: Seasonality

2.1.1 Basic Idea

Seasonal (cyclic) time series have obvious periodic components and thus in frequent spectrum they should have larger amplitudes.

The aim of seasonality dimension is to distinguish seasonal time series (no matter how long period they have) from non-seasonal time series.

2.1.2 Implementation

Algorithm 1 shows the details of how to obtain the values of seasonality.

Algorithm 1 Seasonality Dimension

Input: ts ▷ Raw time series data.
Output: *seasonality*

$ts\_norm \leftarrow \textsc{Normalize}(ts)$
$spectrum \leftarrow \text{FFT}(ts\_norm)$
$amplitude \leftarrow \textsc{Abs}(spectrum)$
$tmp \leftarrow \textsc{Max}(amplitude(3 : length - 2))$ ▷ Discard the first two points and the last two points.
$seasonality \leftarrow \textsc{Signal}(tmp) * (\textsc{Abs}(tmp))^{0.5}$

function $\textsc{Normalize}(ts)$
    $average \leftarrow \textsc{Mean}(ts)$
    $std \leftarrow \textsc{StandardDeviation}(ts)$
    $ts\_norm \leftarrow (ts - mean)/std$
    return ts_norm

Because ts_norm is a real number sequence, *spectrum* is symmetric. The first and last points of *amplitude* are direct current components, which are zero after normalization, thus I discard them.

I also discard the second and penultimate points because the values of them can be large in some non-seasonal time series. This is totally empirical and I haven't figured out the reason.

2.1.3 Results

Figure 1 is a histogram. The X axis shows the value of this feature, and the Y axis shows the numbers of time series whose values of this feature are in a specific X-value interval.

From figure 1, we can easily see that cyclic time series are separated from other five kinds of time series.

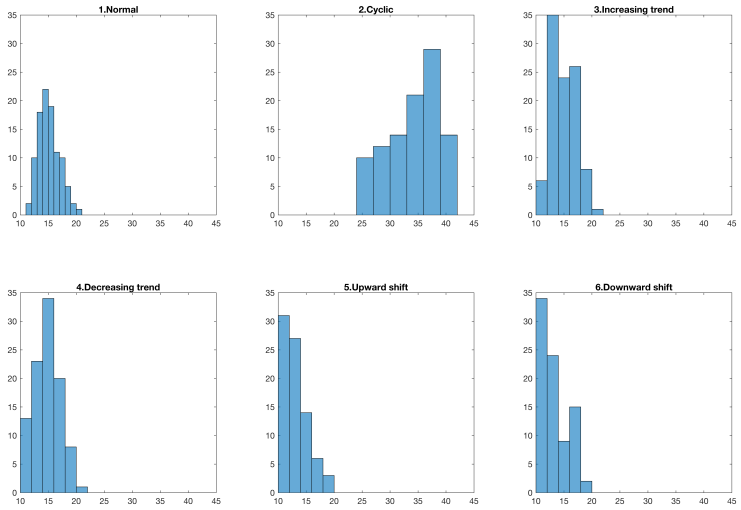

Figure 1: Seasonality histograms of six kinds of time series.

2.2 D2: Global Trend

Global trend is to show how much the value increases or decreases by from the start to the end.

2.2.1 Basic Idea

Apply normalization and smoothing firstly in order to avoid scaling and noise, and then calculate the difference between last point and first point.

2.2.2 Implementation

Algorithm 2 shows the details.

Algorithm 2 Global Trend Dimension

Input: ts ▷ Raw time series data.
Output: $global\_trend$

$ts\_norm \leftarrow \text{NORMALIZE}(ts)$
$ts\_smooth \leftarrow \text{SMOOTHING}(ts\_norm, 6)$
$global\_trend \leftarrow ts\_smooth(length) - ts\_smooth(1)$

function NORMALIZE(ts)
    $average \leftarrow \text{MEAN}(ts)$
    $std \leftarrow \text{STANDARDDEVIATION}(ts)$
    $ts\_norm \leftarrow (ts - mean)/std$
    return ts_norm

function SMOOTHING(ts, wl) ▷ wl is the window length
    $window \leftarrow [1/(2 * wl); \text{ONES}(wl - 1, 1) * (1/wl); 1/(2 * wl)]$
    $ts\_smooth \leftarrow \text{CONVOLUTION}(ts\_norm, window)$
    return ts_smooth

2.2.3 Results

The results are shown is Figure 2. As you can see, it can easily divide the six classes into three groups: (increasing trend, up shift), (decreasing trend, down shift) and (normal, cyclic). However it is hard for global trend to tell the differences within a group, for example, increasing trend and up shift.

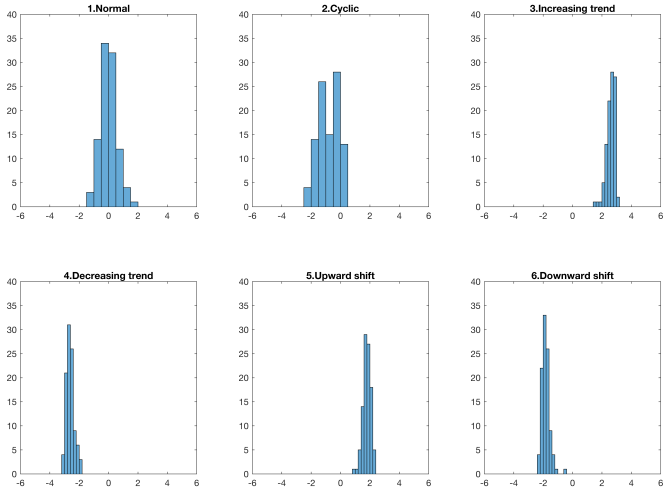

Figure 2: Global trend histograms of six kinds of time series.

2.3 D3: Up/Down Shifts vs Increasing/Decreasing Trend

Since we have been able to distinguish cyclic, up trend and down trend with the help of D1 and D2, what we need to do is to tell the differences between shifts and steady trends.

2.3.1 Basic Idea

The basic idea is to detect the shifts of time series and leverage that up/down shifts contribute most of the total increasement / decreasement.

2.3.2 Implementation

Algorithms 3 shows the details.

Algorithm 3 Shift Dimension

Input: ts ▷ Raw time series data.
Output: $shift$

$ts\_norm \leftarrow \textsc{Normalize}(ts)$
$ts\_smooth \leftarrow \textsc{Smoothing}(ts\_norm, 6)$
$slope\_step \leftarrow \textsc{GetStepSlope}(ts\_smooth)$
$global\_trend \leftarrow ts\_smooth(length) - ts\_smooth(1)$
$slope\_step\_abs \leftarrow \textsc{Abs}(slope\_step)$
$shift \leftarrow global\_trend / \textsc{Max}(slope\_step\_abs)$

function $\textsc{Normalize}(ts)$
    $average \leftarrow mean(ts)$
    $std \leftarrow standard\_deviation(ts)$
    $ts\_norm \leftarrow (ts - mean)/std$
    return ts_norm

function $\textsc{Smoothing}(ts, wl)$ ▷ wl is the window length
    $window \leftarrow [1/(2 * wl); \textsc{Ones}(wl - 1, 1) * (1/wl); 1/(2 * wl)]$
    $ts\_smooth \leftarrow \textsc{Convolution}(ts\_norm, window)$
    return ts_smooth

function $\textsc{GetStepSlope}(ts, step)$
    for $i = 1 \rightarrow length - step$ **do**
        $slope\_step(i) \leftarrow ts(i + step) - ts(i)$
    return $slope\_step$

2.3.3 Results

Figure 3 shows the histograms of shift dimension. We can separate shifts from steady trends using shift dimension.

2.4 Other Processing

After obtaining all features of time series, I use them to form the feature vectors. Then apply normalization to each dimension for all time series data. The

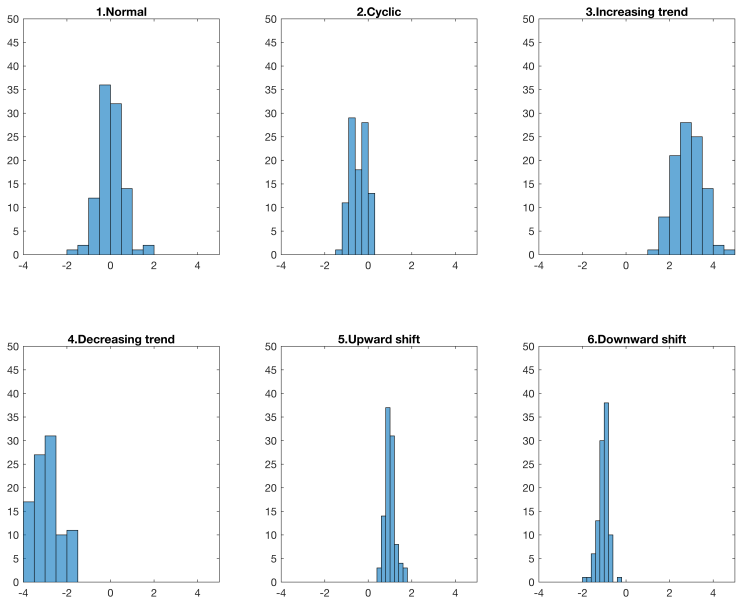

Figure 3: Shift histograms of six kinds of time series.

normalization is the same as *Normalize* function in algorithm 1.

3 Experiments

3.1 Groundtruth

The correct clusters in feature space is shown in Figure 4. Six classes of time series are six separated clusters in feature space, which is good for Kmeans method.

3.2 Good Results

Figure 5 shows a good clustering result with 94% accuracy. Table 1 shows the clustering details where red numbers mean right clusters.

Table 1: Cross table for a good result and groundtruth.

	Normal	Cyclic	Increasing	Decreasing	Up shift	Down shift
Cluster 1	87	0	0	0	0	2
Cluster 2	0	0	0	88	0	0
Cluster 3	0	0	91	0	0	0
Cluster 4	0	100	0	0	0	0
Cluster 5	8	0	9	0	100	0
Cluster 6	5	0	0	12	0	98

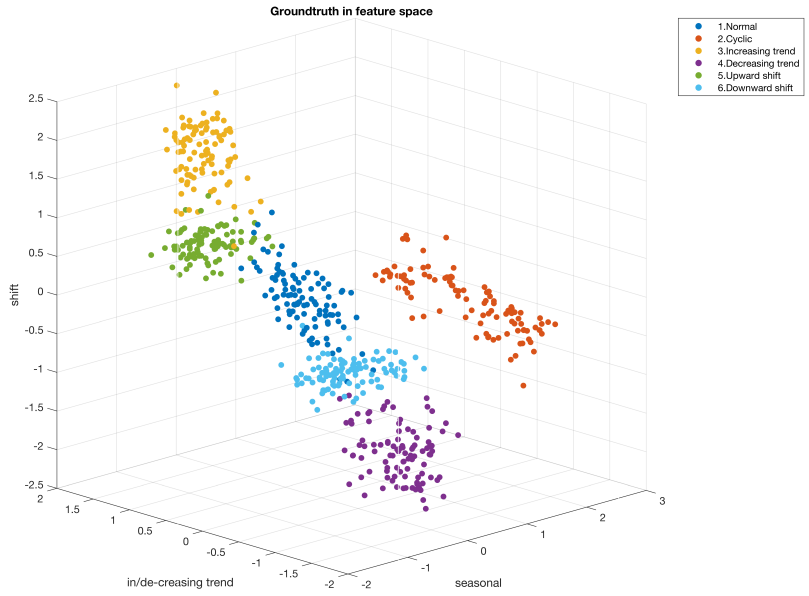

Figure 4: Correct clusters in feature space.

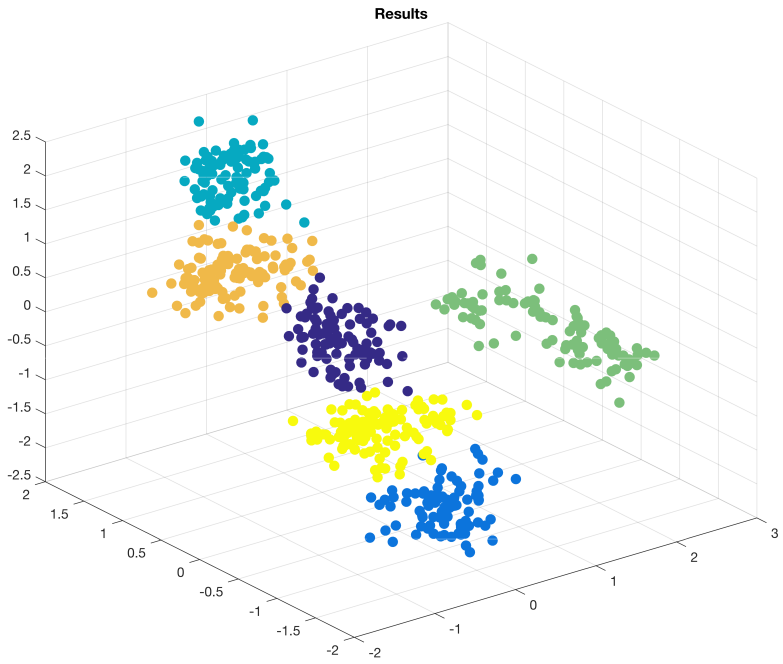

Figure 5: Good clustering result using Kmeans.

3.3 Not So Good Results

Figure 6 shows a not so good clustering result with 72% accuracy. Table 2 shows the clustering details where red numbers indicate right clusters.

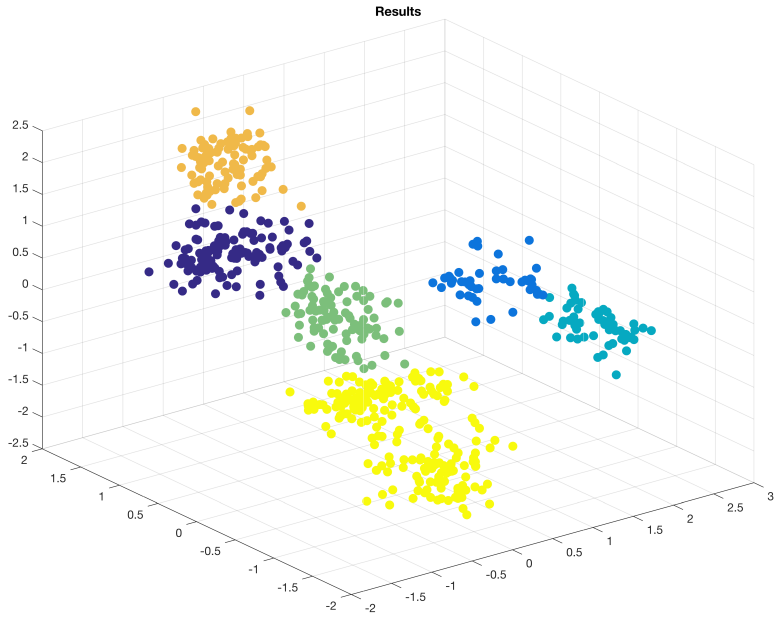

Figure 6: Not so good clustering result using Kmeans.

Table 2: Cross table for a not so good result and groundtruth.

	Normal	Cyclic	Increasing	Decreasing	Up shift	Down shift
Cluster 1	10	0	9	0	100	0
Cluster 2	0	45	0	0	0	0
Cluster 3	0	55	0	0	0	0
Cluster 4	87	0	0	0	0	2
Cluster 5	0	0	91	0	0	0
Cluster 6	3	0	0	100	0	98

4 Potential Work

4.1 Decomposition

We could do *time series decomposition* first and divide a time series sequence into three components: seasonality, trend and noise. Actually, the smoothing operation of current algorithm is a part of decomposition.

4.2 Representative Trends

Try to find a proper way to find the representative trend of each cluster.

4.3 Other Data Sets

I will try this algorithm on other data sets and add more features based on visual perception to make this algorithm more robust and universal.

4.4 Initialization of Kmeans

There might be some problems with the initialization of Kmeans, which likely leads to the not good result of 3.3. I will try to figure it out.

References

[1] Liao, T. Warren. "Clustering of time series data—a survey." *Pattern recognition* 38.11 (2005): 1857-1874.

[2] UCI Machine Learning Repository. "Synthetic Control Chart Time Series."http://archive.ics.uci.edu/ml/databases/synthetic_control/synthetic_control.data.html.