Anlysis and Application of Pearson's Correlation and Case Matching Similarity on Time Series

Marc-Alain Chételat
marc-alain.chetelat@uzh.ch

January 24th, 2016

1 INTRODUCTION

A time serie s is a sequence of data points that consists of measurements over a time interval. The data points are frequently stored in a time series $DB = \{s_1, s_2, ..., s_n\}$. Such data monitoring and analysis is done in a lot of domains such as financial markets, meteorology or astronomy. E.g. weather stations register in a fixed time interval outside temperatures in cities, valleys or even on top of mountains.

We first analyze the Pearson's Correlation Coefficient and the Case Matching algorithm and calculate the values for sample data (Table 1). Then the algorithms are implemented using Java. Finally we analyze the runtime as well as the space complexity.

2 EXAMPLE DATA SET

We are using weather data of 21 weather stations located in South Tyrol. Each station measures and persists the outside temperature every five minutes starting on the January 1st, 2014 and ending on the December 30th, 2014. Hence the structure of my data set is [Station ID] - [Timestamp] - [Temperature]. Table 1 shows sample data of two time series s_1 and s_2 in an interval of 45 minutes of two different measurement stations in the South Tyrol. This data will be used in the later examples.

Time t	$s_1(t)$ °C	$s_2(t)$ °C
13:10	3.425	4.429
13:15	3.263	4.5
13:20	3.085	4.529
13:25	2.861	4.629
13:30	2.837	4.671
13:35	3.124	4.6
13:40	3.5	4.6
13:45	3.5	4.694
13:50	3.5	4.841
13:55	3.5	4.788

Table 1: Sample weather data.

3 PEARSON'S CORRELATION COEFFICIENT

3.1 Definition[1]

PCC measures *linear* correlation between two time series. A *linear* correlation exists if a parameter changes proportionally as another parameters changes. An example is shown in Figure 1. The function is defined as $f(x) = \frac{2}{3}x$. Due to linearity, if value x increases 3 from 3 to 6, y increases proportionally $2 = 3 \times \frac{2}{3}$ from 2 to 4. To apply PCC, the two time series must be scaled to intervals. The PCC indicates the degree of correlation. It ranges from -1 to +1. Indicating +1 (-1 respectively) a *perfect positive* (*negative* respectively) *linear* correlation. If PCC indicates 0, there is *no correlation* at all between the time series. Let $s_1(t)$

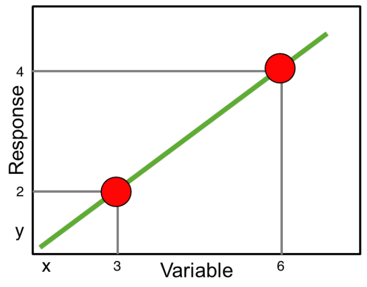

Figure 1: Linear correlation.

and $s_2(t)$ be two time series such that $|s_1(t)| = |s_2(t)|$, then PCC is defined as

$$PCC(s_1, s_2) = \frac{\sum_{t=1}^{|s_1|}(s_1(t) - \bar{s_1})(s_2(t) - \bar{s_2})}{\sqrt{\sum_{t=1}^{|s_1|}(s_1(t) - \bar{s_1})^2}\sqrt{\sum_{t=1}^{|s_2|}(s_2(t) - \bar{s_2})^2}} \tag{1}$$

where $\bar{s_i} = \frac{1}{|s_i|}\sum_{t=1}^{|s_i|} s_i(t)$.

3.2 Example

The two time series shown in Table 1 are used to compute the PCC. The first step is to calculate the means for each data set defined as $\bar{s_i} = \frac{1}{|s_i|}\sum_{t=1}^{|s_i|} s_i(t)$, giving us the results listed in Table 2. The residuum $s_i(t) - \bar{s_i}$ quantifies how

$\bar{s_1}(t)$	$\bar{s_2}(t)$
4.628	3.260

Table 2: Mean values for $s_1(t)$ and $s_2(t)$.

much the actual value differs from the time series' mean. Whether the value differs positively or negatively does not matter. Therefore the residuum gets squared. The mean value of all residua is calculated resulting in the time series' standard deviation defined as

$$\sigma = \sqrt{\frac{1}{|s_i|}\sum_{t=1}^{|s_i|}(s_i(t) - \bar{s_i})^2}. \tag{2}$$

The standard deviation indicates how much the values scatter around the time series' mean. In Table 3 the standard deviations for the corresponding time series are listed. The last element missing is the covariance. It quantifies the

	$s_1(t)$	$s_2(t)$
σ	0.380	0.803

Table 3: Standard deviation for $s_1(t)$ and $s_2(t)$.

linear correlation of two variables. A positive algebraic sign means that the two compared variables are positively correlated (e.g. one variable increases its size the other one does too). A negative algebraic sign means the opposite (e.g. one variable decreases on increasing the other one). The sample's covariance is defined as

$$Cov(s_1, s_2) = \sum_{t=1}^{|s_1|}(s_1(t) - \bar{s_1})(s_2(t) - \bar{s_2}) \tag{3}$$

resulting in $Cov(s_1, s_2) = 0.068$. The covariance's dimension depends on the values' sizes. Therefore it reflects the *direction* (positive/negative) of the correlation only. Information about the correlation's *strength* is not available. The PCC reflects the standardized covariance making a statement about the correlation's strength and a comparison possible.

$$PCC(s_1, s_2) = \frac{0.068}{0.380 \times 0.803} = 0.223 \tag{4}$$

The PCC of 0.223 concludes a weak positive correlation between the two time series $s_1(t)$ and $s_2(t)$.

4 CASE MATCHING SIMILARITY[2]

4.1 Definition

As shown the PCC can be used to measure the correlation of *linear* correlated values. The Case Matching Similarity (CMS) works as well with time series that are *non-linearly* correlated. A parameter's value in a *non-linear* correlated data set does not change proportionally as changing the other parameter's value. Figure 2 shows a *non-linear* correlation. The function is defined as $f(x) = x^3$. If x increases from 0.5 to 1.5 the corresponding y-value is not increasing proportionally. However *non-linearity* does not mean values can not correlate! E.g. we consider two measuring stations in a valley. Station 1 is located on one side where morning sun reaches, station 2 is located on the other side of the valley where evening sun shines in. In the morning, station 1 registers higher temperature values than station 2 for the same time (in the evening vice versa) because sun heats up the air around station 1 and does the same in the evening around station 2 on the other side of the valley. Hence different factors influence on temperature, the time series of station 1 and 2 do not have to correlate *linearly* but they still *correlate* due to stable meteorological conditions. CMS is considering this fact and assumes related to data from Table 1 that the same

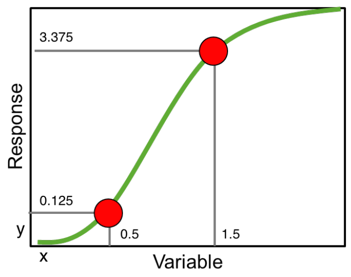

Figure 2: Non-linear correlation.

values of $s_1(t)$ and $s_2(t)$ *co-occur* frequently. This means if for points in time t_1, t_2 the values in s_1 are similar the values in s_2 are as well. Even if the values from s_1 distinct clearly from those in s_2, i.e.

$$\forall t_1, t_2 : s_1(t_1) \approx s_1(t_2) \rightarrow s_2(t_1) \approx s_2(t_2) \tag{5}$$

CMS proposes an algorithm to actually measure the *strength* of co-occurrence between two time series s_1 and s_2. CMS splits the range of time series s_1 into equal-sized sub-ranges, called *buckets*. Each bucket contains the values of s_2 such that s_1 is within the bucket limits.

$$b_z = \{s_2(t) | \forall t : zw \leq s_1(t) < (z+1)w\} \tag{6}$$

with $z \in Z$ as the bucket's ID and $w \in \mathbb{R}_{>0}$ as the bucket's width. CMS then calculates the bucket's standard deviation. The smaller the deviation is, the closer are the values $s_2(t) \in b_z$ to the bucket mean $\bar{b}_z$, the more similar are they to each other and the *stronger* is the co-occurrence. Formally, the standard deviation for each bucket is defined as

$$\sigma_z = \sqrt{\frac{1}{|b_z|} \sum_{s_2(t) \in b_z} (s_2(t) - \bar{b_z})^2} \tag{7}$$

with $\bar{b_z} = \frac{1}{|b_z|} \sum_{s_2(t) \in b_z} s_2(t)$ as the bucket's mean. Let $B = \{b_z | \forall z \in Z : b_z \neq \emptyset\}$ be the set of all non-empty buckets. Then CMS is defined as the average bucket standard deviation, where each term is weighted by the number of elements in the corresponding bucket

$$CMS(s_1, s_2) = \frac{1}{|B|} \sum_{b_z \in B} \frac{|b_z|}{|s_2|} \sigma_z \tag{8}$$

4.2 Example

Using the example data set from Table 1, each value $s(t)$ gets mapped into the corresponding buckets b_z with bucket values $[zw, (z+1)w)$. In the example we will calculate the $CMS(s_1, s_2)$ for bucket widths $w = \{0.2, 0.5, 1.0\}$. In tables 4, 5 and 6 the buckets b_z, timestamps t and the corresponding values $s_2(t)$ are shown. If width w is large, a bucket covers a wider range values $s_1(t)$

b_z	$s_2(t)$ °C	t
b_{14}	4.629	13:25
	4.671	13:30
b_{15}	4.529	13:20
	4.6	13:35
b_{16}	4.5	13:15
b_{17}	4.429	13:10
	4.6	13:40
	4.694	13:45
	4.841	13:50
	4.788	13:55

Table 4: Buckets b_z with corresponding $s_2(t)$ for bucket width $w = 0.2$.

and more values $s_2(t)$ are put into the same bucket b_z (e.g. with $w = 0.2$, the number of buckets $|B| = 4$, where with $w = 1.0$, $|B|$ decreases to 2). Equation 7 calculates each bucket's standard deviation. Again, the standard deviation indicates how much differ the bucket's values from its average (e.g. a value $s_2(t)$ in bucket b_{14} differs 0.021 °C in average from the bucket mean $\bar{b_{14}}$). Table 7 shows all standard deviations for all buckets b_z. Finally CMS calculates the sum of the average bucket standard deviation weighted by the number of elements in the corresponding bucket (e.g. for $w = 0.2$ CMS is defined as $CMS(s_1, s_2) = \frac{2}{10}0.021 + \frac{2}{10}0.035 + \frac{1}{10}0 + \frac{5}{10}0.146 = 0.084$). We conclude that

b_z	$s_2(t)$ $°C$	t
b_5	4.629	13:25
	4.671	13:30
b_6	4.429	13:10
	4.5	13:15
	4.529	13:20
	4.6	13:35
b_7	4.6	13:40
	4.694	13:45
	4.841	13:50
	4.788	13:55

Table 5: Buckets b_z with corresponding $s_2(t)$ for bucket width $w = 0.5$.

b_z	$s_2(t)$ $°C$	t
b_2	4.629	13:25
	4.671	13:30
b_3	4.429	13:10
	4.5	13:15
	4.529	13:20
	4.6	13:35
	4.6	13:40
	4.694	13:45
	4.841	13:50
	4.788	13:55

Table 6: Buckets b_z with corresponding $s_2(t)$ for bucket width $w = 1.0$.

w	b_z	σ $°C$
0.2	b$_{14}$	0.021
	b_{15}	0.035
	b_{16}	0
	b_{17}	0.146
0.5	b_5	0.021
	b_6	0.061
	b_7	0.092
1.0	b_2	0.021
	b_3	0.133

Table 7: Standard deviations for all buckets where $w = \{0.2, 0.5, 1.0\}$.

for $w = 0.2$ the values $s_2(t)$ scatter 0.084 $°C$ around the average of all bucket's standard deviations. Table 8 shows every CMS-value for all bucket width w.

w	$CMS(s_1, s_2)$
0.2	0.084
0.5	0.066
1.0	0.111

Table 8: CMS for time series s_1, s_2 where $w = \{0.2, 0.5, 1.0\}$

4.3 Implementing CMS

The implementation is done in Java programming language. First, the data is mapped into a tree based table that allows to control having for each time stamp t exactly one value pair $s_1(t), s_2(t)$. The input of the algorithm are the two time series s_1, s_2 as a double array and the bucket width w as a float variable. Because a value pair only exists $\forall$ s$_1(t) \in$ s$_1$, $\exists$ s$_2(t) \in$ s$_2$ and $n = |s_1| = |s_2|$, a run through the data set is done in n steps. As shown in Algorithm 1, the data set is run through entirely three times. For more detailed complex analysis read section 4.4. Function "getBucketMap" maps each value $s_2(t)$ to its corresponding bucket b_z, persisted in a tree map with b_z as key and $s_2(t)$ as value. Each bucket's mean and standard deviation then gets persisted into the "bucketMeanMap", "bucketStandardDeviationMap" respectively with b_z as key and $\bar{b_z}$ / σ_z as value. Last but not least "getCMS" calculates the CMS of the two time series s_1, s_2.

4.4 Complexity Analysis

As shown in pseudo code of CMS, all values s_1 are run through once in function "getBucketMap" having a runtime of $O(n)$ with $n = |s_1| = |s_2|$. In the method "getBucketMeanMap" all values s_2 are run through as well thus having the same runtime $O(n)$, so does the method "getBucketStandardDeviationMap". The overall runtime complexity results in $O(3n) = O(n)$.

Regarding space complexity, assumed both time series s_1 and s_2 are loaded into main memory with space consumption $O(n)$ each. At runtime each value $s_2(t)$ is put into lists consuming $O(n)$ steps again. Overall space complexity equals $O(2n + n) = O(n)$.

4.5 Pseudo Code of CMS

Input: Two time series s_1, s_2 and bucket width w
Output: CMS(s_1, s_2)

Listing 1: Pseudo Code of CMS

```
function getBucketMap(s1, s2, w)
     bll = (int) s1(t)  // bll as bucket lower limit

```

```
    while (s2 is not persisted into map)
        for each s1(t) in s1
            if (s1(t) >= bll && s1(t) < bll + w) then
                bucketMap.put(bll, s2(t))
                else
                if (s1(t) > bll) then
                        bll += w
                    else
                        bll -= w

return bucketMap

function getBucketMeanMap(bucketMap)

    for each bll
        sum(s2)

    bucketMean = sum/bll.size

    bucketMeanMap(bll, bucketMean)

return bucketMeanMap

function getBucketStandardDeviationMap(bucketMap, bucketMeanMap)

    for each bll
        for each s2(t)
            sum(s2(t) - bucketMean)^2

    variance = sum/s2.size
    standardDeviation = sqrt(variance)

    bucketStandardDeviationMap(bll, standardDeviation)

return bucketStandardDeviationMap

function getCMS(bucketMap, bucketStandardDeviationMap, s2)

    for each bll
        cms = bll.size / s2.size * standardDeviation

return cms
```

References

[1] Jie Liu Abdullah Muen Suman Nath. “Fast Approximate Correlation for Massive Time-series Data”. In: (2010).

[2] Kevin Wellenzohn Michael Boehlen. “Continuous Imputation of Missing Values in Highly Correlated Time Series”. In: (2016).