# Leveraging Metadata for Extracting Robust Multi-Variate Temporal Features

by

Xiaolan Wang

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved July 2013 by the
Graduate Supervisory Committee:

Kasim Selçuk Candan, Chair
Maria Luisa Sapino
Georgios Fainekos
Hasan Davulcu

ARIZONA STATE UNIVERSITY

August 2013

# ABSTRACT

In recent years, there are increasing numbers of applications that use multi-variate time series data where multiple uni-variate time series coexist. However, there is a lack of systematic of multi-variate time series. This thesis focuses on (a) defining a simplified inter-related multi-variate time series (IMTS) model and (b) developing robust multi-variate temporal (RMT) feature extraction algorithm that can be used for locating, filtering, and describing salient features in multi-variate time series data sets. The proposed RMT feature can also be used for supporting multiple analysis tasks, such as visualization, segmentation, and searching / retrieving based on multi-variate time series similarities. Experiments confirm that the proposed feature extraction algorithm is highly efficient and effective in identifying robust multi-scale temporal features of multi-variate time series.

# ACKNOWLEDGEMENTS

I would like to take this opportunity to thank for all the help I received in completing this thesis.

First of all, I greatly appreciate all the help I received from my advisor Dr. K. Selçuk Candan during my two years graduate study. This thesis could not be possible without his guidance and help. I would also like to thank my committee members: Dr. Maria Luisa Sapino, Dr. Georgios Fainekos and Dr. Hasan Davulcu for join my committee and give valuable advises for my thesis.

I would like to thank all the members in Emitlab: Xilun Chen, Mithila Nagendra, Parth Nagarkar, Shengyu Huang,Jung Hyun Kim, Mijung Kim, Xinsheng Li, Sicong Liu, and Sriram Rathinavelu. My thank also extend to Rosaria Rossini and Antonio Penta who I have collaborated in previous researches.

And last, I thank my parents for supporting and encouraging me for my studies.

# TABLE OF CONTENTS

CHAPTER Page

# LIST OF TABLES

Table Page

## LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Chapter 1

# INTRODUCTION

Various novel applications of time series data, such as motion capture, disease diagnosis and energy system control, have emerged in recent years where traditional uni-variate time series are not sufficient in describing temporal observations. Instead, researchers have proposed the idea of multi-variate time series to accommodate applications where we simultaneously observe multiple aspects of the system. Moreover, these applications often have the following properties: (1) data is often larger, (2) variates are inter-related, and (3) observations can be presented in multiple resolutions. However, how to effectively and efficiently analyze, search and mine multi-variate time series is still an open question. In this thesis, the main goal is define a multi-variate time series model as well as an algorithm for extracting local salient features, which are suitable for analysis and retrieval, from multi-variate time series data. We start by introducing the basic concepts and characteristics of time series and then present an overview of existing feature extraction algorithms.

## 1.1 Uni-Variate Time Series

Uni-variate time series data (Figure 1.1), typically captured successively at discrete time stamps, is presented as a sequence of scalar observations. To support uni-variate time series indexing [18] and mining (*e.g.* cluster [32] and classification [46] tasks), various global and

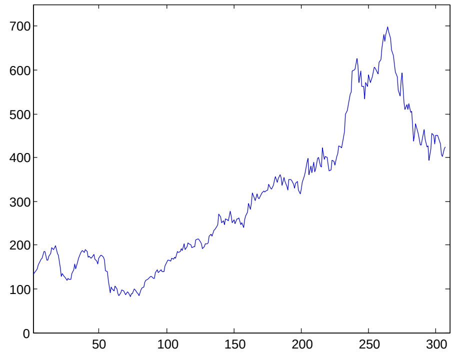


Figure 1.1: A sample uni-variate time series.

local feature extraction algorithms have been proposed.

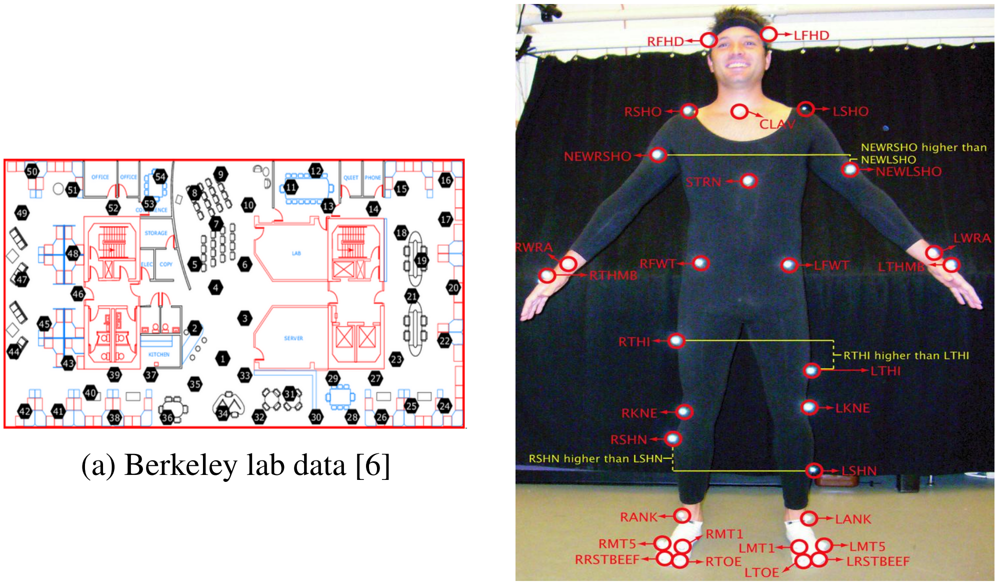
**Global uni-variate time series features:**

Global uni-variate time series features are used for indexing and searching uni-variate time series where the characteristics of the series as a whole are needed for search and analysis. One approach is to use linear transformations, such as discrete Fourier transforms (DFT) [1] and discrete wavelet transform (DWT) [10], to map the given time series into a new basis, which highlight important global characteristics, such as dominant frequency components that make up the time series.

One of the key aspects in understanding uni-variate time series is to find the distance between two time series. Many algorithms have been proposed for this task: *e.g.*, Euclidean distance[1], Dynamic Time Warping (DTW)[18], Edit Distance with Real Penalty (ERP)[11], Longest Common Subsequence (LCSS)[43], Edit Distance on Real Sequence (EDR)[12], and Dictionary Compression Score (DCS)[34]. Intuitively, these algorithms search for alignments of time series in order to calculate the matching distance between them. However, while global features can hint at whether such alignments exists, finding such alignments through global features is not possible. Instead, local uni-variate time series features are needed to support this process.

**Local uni-variate time series features:**

Local uni-variate time series features are widely used in searching for alignments between uni-variate time series, time series segmentation, visualization, clustering and classification tasks. Time series shapelets [47], presented as time series subsequences, is considered as a kind of local feature which maximally represents a given class. Time series snippets [45] is captured using hidden Markov model (HMM) techniques and correspond to shape contours. Frequently repeating patterns can be extracted by motif [37] extraction algorithms. While the above algorithms tend to use supervised learning strategies, sDTW [8] relies on an unsupervised approach for locating local features. Moreover, sDTW [8] aimed at searching local salient features, which include feature location, scope and descriptor, are



(a) Berkeley lab data [6]

(b) Motion capture data [14]

Figure 1.2: Two multi-variate time series examples. In (a), the spatial positions of the temperature sensors (and the locations of the space partitions) relate the observations at different sensors. Similarly, in (b) the structure of the human body relates the positions of the location markers during the motion capture.

robust against multiple kinds of noises. We discuss sDTW (which we refer to as UNI is this thesis) in greater detail in Section 2.2.

## 1.2 Multi-Variate Time Series

Multi-variate time series is essential in various applications, such as Mocap[14] motion capture and smart building energy time series data sets. In many of these applications (a) the resulting time series data are multi-variate, (b) relevant processes underlying these time series are of different scale [42], and (c) the variates (i.e., observation parameters) are dependent on each other in various ways (Figure 1.2).

Analysis of relationships (correlations, transfer functions, and causality) among time series is a well studied problem. H.Akaike 1977 [3] used canonical correlation concept for analysis of variates relations of time series. Silva et al. 2010 [15] proposed a component vector based model in discovering the inter relationship between series that improve fit and forecasts. Eichler 2000 [16] proposed multi-variate time series relationship

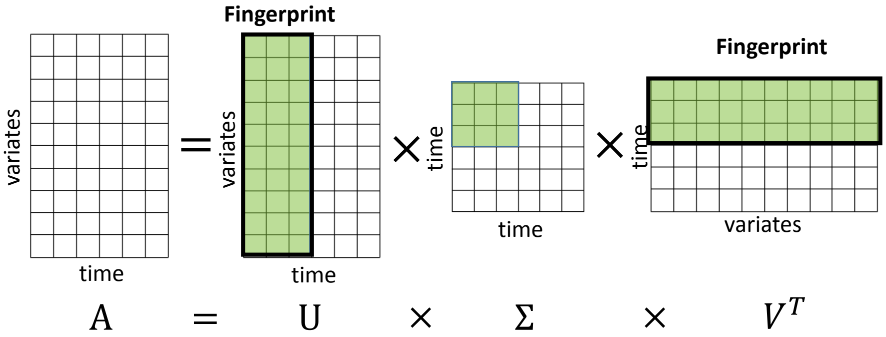


Figure 1.3: Find multi-variate time series Fingerprint with SVD algorithm.

analysis according to granger-causality graphs.

**Global multi-variate time series features**:

The most common approach for extracting features from multi-variate time series is to seek global features, such as correlations, transfer functions, variate clusters, and spectral properties [15]. A common representation of multi-variate data evolving over time is a tensor (multi-dimensional array) stream. The order of a tensor is the number of modes (or ways): a first-order tensor, a vector, is often used to represent uni-variate time series, whereas a second-order tensor can be used for multi-variate series. A 3- or higher-order tensor can be used when each variate itself is multi-dimensional. Tensor deposition algorithms such as CP decomposition [21] and Tucker decomposition [41] can be thus used in seeking global features. When variates are one dimensional, a multi-variate time series can be represented in the form of matrix. Multi-variate time series can thus be analyzed for latent variables and indexed for searching using matrix decomposition methods known as singular value decomposition (SVD) or principle component analysis (PCA). As shown in Figure 1.3, given a multi-variate time series, $A$, it can be factorized as the product of a unitary matrix $U$, a diagonal matrix $\Sigma$, and another unitary matrix $V^T$. Fingerprint of multi-variate time series can either be the first few columns in $U$ or be the first few rows in $V^T$ in such a way that the majority of energy are preserved. CLeVer [49], which using Feature subset selection (FSS) to pre-process the multi-variate time series data, can improve the classification accuracy of the PCA based global feature. Besides, Li et al. 2010 [36] argued that

Linear Dynamical Systems (LDS) can be used for correlation discovery and forecasting in multi-variate time series. They also proposed a PLif algorithm for detecting interpretable features for multi-variate time series.

**Local Multi-variate Time Series Features**:

TClass [30] was proposed to improve classification accuracy of multi-variate time series, where sub-events will be learned from training instance. Each sub-event has a starting time and duration as well as temporal patterns presented by parameters that are learned from existing propositional machine learning techniques (*i.e.* HMM). Different with sub-pattern in TClass, recent temporal pattern [4] presents a list of states (time interval patterns) and the relation (before, at the same time and after) among states. Motion Stream [35] was proposed to detect motion events from multi-variate motion time series (*i.e.* mocap motion). Precision of each motion stream is incrementally improved by comparing and maximizing the difference with existing motion streams that are detected. Though some approaches have been done extracting local features for multi-variate time series, most feature extraction approaches for multi-variate time series focus on learning global relationships among the variates through factorization and decomposition.In this thesis, we observe that besides global features, local salient features also exist in multi-variate time series. Figure 1.4 provides an example where positions and scopes of sample multi-variate features on a multi-variate time series; note that these features are of different temporal length and contain different number of variates.

## 1.3 Challenges and Contributions

Though many studies have been made in multi-variate time series analysis, to the best of my knowledge, none of the existing fingerprint/feature extraction algorithms focus on non supervised learning of local salient features. Therefore, in this thesis I track the following challenges:

**Multi-variate time series modeling:**

In this thesis, I first introduce a multi-variate time series model that involves the rele-

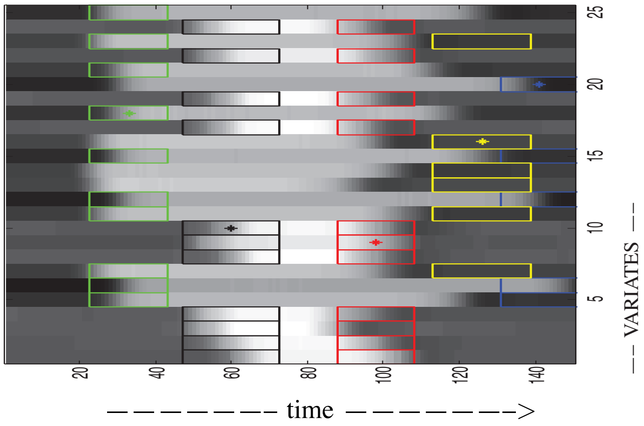


Figure 1.4: A multi-variate time series data set, where each variate is plotted as a row of gray scale pixels and sample multi-variate features identified on the data set (each feature is marked with a different color). More specifically, the figure shows 26 time series of length 150 and 5 local multi-variate features on these time series: note that some of the features correspond to the onset of a rise in amplitude, whereas other correspond to the drop in the series amplitude. For each time series involved in a given multi-variate feature, we plot the corresponding temporal scope (i.e., duration) of that feature. For each feature, the time series marked with a "*" is the series on which that feature is centered. Note that the set of the time series involved in a given feature as well as the position and scope of the feature are automatically detected by the RMT feature extraction algorithm.

vant temporal characteristics while leveraging matadata for variates relationship construction. Applicability of the proposed multi-variate time series model to different applications should be maximized while the assumptions about the structure are minimized.

**Local feature detection challenge:**

Uni-variate time series carry localized temporal features [8] that are scale invariance and robust against multiple types of noise. In multi-variate time series, local salient features corresponding to major changes in temporal dynamics or in the relationships among variates. Thus the second challenge I tackle in this thesis to detect local salient features of multi-variate time series.

As mentioned above, variates are often inter related or correlated. In this thesis, I argue that known relationships among variates can promote accuracy and robustness of the extracted features. Therefore, this thesis aim at extracting features in a way that takes into account any variate relationship knowledge available as a priori.

We propose *robust multi-variate temporal (RMT)* features and develop algorithms for locating and extracting thesis multi-variate features. Experiment results, presented in Chapter 7, confirm that the proposed RMT features are highly effective and efficient in identifying robust temporal features for multi-variate time series and can be used for indexing, visualization, segmentation and clustering/classification.

## 1.4 Organization of the Thesis

Organization of the thesis is as follows: relevant background (existing scale invariant feature extraction algorithm for images and uni-variate time series) will be discussed in Chapter 2. In Chapter 3, the proposed algorithms for identifying and describing *robust multi-variate temporal (RMT)* features will be presented. In Chapter 4, we will discuss the feature alignment process for multi-variate data sets using the proposed RMT features. A visualization system based on the proposed RMT features will be proposed in Chapter 5. Experimental setup, including data sets and evaluation criteria, will be introduced in Chapter 6 and experimental results will be reported and analyzed in Chapter 7. Conclusion of the thesis will be presented in Chapter 8.

# Chapter 2

# BACKGROUND

As described in the introduction, the problem of extracting local features from uni-variate series has already been solved [8]. Similar local features have also been extracted from 2D images to support indexing and object search (SIFT [38][39]). These techniques rely on (a) repeatedly smoothing of the data to generate different versions of the input object corresponding to different scales and (b) comparing neighboring points both in time (or in x and y dimensions) and scale to identify regions where the gradients are large. Once features are located, feature descriptors (in the form of gradient histograms) are extracted to support indexing and search. In this section, we provide overviews of SIFT [38][39] and sDTW [8] algorithms to help us highlight the challenges that make the problem of extracting local features from multi-variate series difficult.

## 2.1 SIFT Feature for 2D images

The scale-invariant feature transform (SIFT) algorithm [38][39] was proposed to identify 2D image keypoints which are robust against scaling, translation, rotation, and partially invariant towards illumination differences and noise. SIFT algorithm relies on four major steps to extract features from 2D image matrix: scale-space extrema detection, keypoint localization, orientation assignment, and keypoint description.

**Scale-space extrema detection:**

To identify features of different scales, SIFT algorithm will first conduct a convolution process of a variable-scale Gaussian: given 2D image, $I(i,j)$, let $L(i,j,\sigma)$ be the smoothed image:

$$L(i,j,\sigma) = G(i,j,\sigma) * I(i,j),$$

where,

$$G(i,j,\sigma) = (1/2\pi\sigma^2)e^{-(i^2+j^2)/2\sigma^2}.$$

In other words, each pixel in the original 2D image matrix is smoothed by its neighbors by several iterations. Afterwards, the difference of Gaussian of the input image can be produced by subtract between nearby scale $\sigma$ and $\kappa\sigma$:

$$D(i,j,\sigma) = L(i,j,\kappa\sigma) - L(i,j,\sigma).$$

Local extrema are detected by searching through each pixel in DoG and compare with its 26 neighbors in the current scale and its adjacent two scales. A pixel is selected as SIFT feature candidate when is is larger or smaller than all of its 26 neighbors.

**Keypoint localization:**

Poorly localized feature candidates that are either with low contrast or large principle curvature are eliminated in this step. More specifically, to prune features that are poorly localized along edge (with large principle curvature), a Hessian matrix [24], $H$, is constructed by computing the secondary derivatives. The principle curvature is determined as follows: Let $\alpha$ is the largest eigenvalue and $\beta$ is the smallest one, the trace and the determinant of the Hessian matrix can be used for relating $\alpha$ and $\beta$:

$$Tr(H) = \alpha + \beta \quad and \quad Det(H) = \alpha\beta.$$

If $\alpha = r\beta$, the ratio $r$ of the two eigenvalues is computed by trace and determinant:

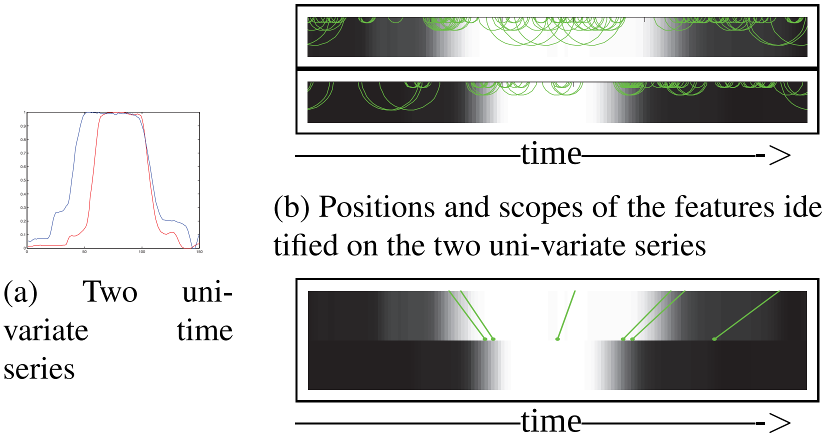
$$\frac{Tr(H)^2}{Det(H)} = \frac{(r\beta+\beta)^2}{r\beta\beta} = \frac{(r+1)^2}{r}.$$

The ratio $r$ should as close to 1 as possible, thus the author claims that given a upper bound $r$, poorly localized features can be identified by evaluating if:

$$\frac{Tr(H)^2}{Det(H)} > \frac{(r+1)^2}{r}.$$

**Orientation Assignment:**

The orientations, $O$, of the remaining features are computed based on the local gradient distributions. More specifically, a gradient direction histogram around the keypoint is generated and the peak of the histogram is selected as the dominant orientation $O$.



(a) Two uni-variate time series

(b) Positions and scopes of the features identified on the two uni-variate series

(c) Feature alignments of the two uni-variate time series

Figure 2.1: Uni-variate feature extraction using the sDTW algorithm [8] – in (b) and (c), each time series is visualized as a row of pixels, where lighter colors correspond to larger values.

**Keypoint description:**

In this step, the spatial distributions around each keypoint are sampled into a histogram. To achieve orientation invariance, the 2D matrix around the keypoint is rotated to be normalized against the dominant orientation and to eliminate impact of noise. A weight will be assigned to each of the sample points by Gaussian weighting function. The closer a point on which the gradient is measured to the keypoint, the larger is its weight.

SIFT algorithm for images has been proved highly effective, and can be used in various applications: such as gesture recognition [44], video tracking [5], match moving [25] and etc.

## 2.2 UNI Feature for Uni-Variate Time Series

### *UNI Feature Extraction Algorithm*

Unlike images, which consist of $n \times m$ variables, uni-variate time series are represented as 1D vectors. To identify local scale invariance features from uni-variate time series, a UNI feature extraction algorithm is proposed in sDTW[8]. The UNI algorithm, which forms the core of the sDTW framework for efficient DTW distance computation [8], follows a similar approach to SIFT, but is optimized for 1D time series:

**Step 1:** Local extrema detection: Difference-of-Gaussian function is first created by iteratively smoothing the given time series to create different layers of detail and by taking difference between the consecutive layers. Candidate keypoints are found by comparing a given point with its neighbors in the resulting DoG space. Only local maxima and minima points are treated as candidates and delivered to next step.

Let $L(i,\sigma)$ be the smoothed time series through convolution with the Gaussian of a given time series $X$,

$$L(i,\sigma) \quad = \quad G(i,\sigma) * X(i) = G(i,\sigma) * x_i,$$

where,

$$G(i,\sigma) \quad = \quad (1/2\pi\sigma^2)e^{-(i^2)/2\sigma^2}.$$

Through Gaussian smoothing, a given uni-variate time series can be presented with multiple scales where Different of Gaussian (DoG) will be computed for discovering the largest variation.

$$D(i,\sigma) = L(i,\kappa\sigma) - L(i,\sigma).$$

where $D(i,\sigma)$ is the difference of Gaussian between nearby scale $\sigma$ and $k\sigma$ ($k$ is a constant multiplicative factor). Note that since uni-variate time series is one dimensional, only 8 neighbors are compared in this step. At this stage, non-useful keypoints are pruned by considering the amount of details each feature captures.

**Step 2:** Keypoints descriptor: As the final step in SIFT algorithm, a local image descriptor is computed for each keypoint obtained from previous steps. Unlike 2D image SIFT, UNI feature descriptor is a vector of length $2a \times c$, where $a$ is a user defined factor indicating the size of neighborhood region centered around the time instant and $c$ is the number of direction bins. UNI feature descriptor is extracted by sampling neighborhood gradients into $c$ bins. Figure 2.1(b) shows an example.

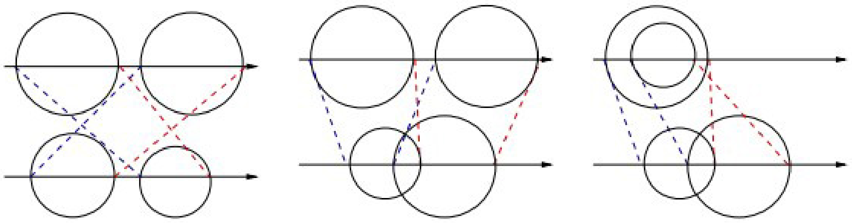
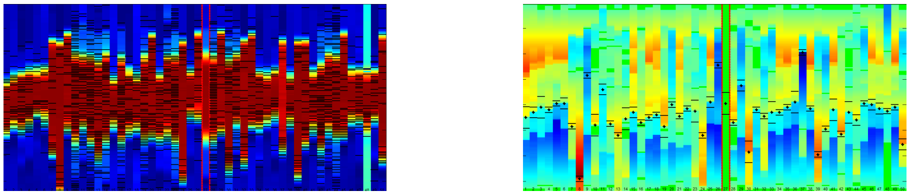


Figure 2.2: Example of scope boundary conflicts [8].

*UNI Feature Alignment*

sDTW[8] also proposed a way to leverage the UNI feature for comparing two given uni-variate time series. In this approach, candidate feature matching pairs are generated by choosing the best degree of matching (in terms of descriptor distance) feature in series *B* for every feature in series *A* as shown in Figure 2.1(b). As we can see, this matching strategy may identify conflicting matching pairs that need to be cleaned. sDTW[8] defines a conflicting matching pair as a pair where the order of feature boundaries do not agree (Figure 2.2) and proposes a greedy algorithm to eliminate these conflicts:

Firstly, the algorithm computes a matching score, $\mu_{comb}$, representing the agreement between two features of all candidate matching pairs. The algorithm then sorts the matching pairs in descending order of scores and gather non-conflicting pairs of feature by going down this list and pruning those pairs that conflict with any of the higher scoring pairs. Figure 2.1(c) shows the maintained matching pairs after the matching pair search process. As we can see, the most discriminating matching pairs are maintained and no conflicts exist. Salient features for uni-variate time series can also be used in visualization tasks. Figure 2.3 shows two local salient temporal feature-based visualization methods for uni-variate time series [9]. Figure 2.3(a) shows a uni-variate time series data set that includes 50 series. Each column corresponds to one uni-variate time series, which is divided into segments by UNI feature boundaries; for a selected feature (shown with a feature center ”*” and two boundaries) in one uni-variate time series (highlighted in rectangular box),



(a) (b)

Figure 2.3: A sample of feature driven visualization for uni-variate time series (Gun data set).

the best matching features in every other series are highlighted and shown in Figure 2.3(b).

Chapter 3

# RMT FEATURES

## 3.1 Inter-related Multi-variate Time Series Model

There are various multi-variate temporal data models, such as the *multi-variate structural time series* model [22] and its variants, including the *vector innovations structural time series* framework [15]. These models describe a multi-variate time series based on various assumptions about its structure, including cyclicity, hysteresis, and known relationships among variates. In order to maximize the general applicability of the RMT features detection algorithms, in this section, we present an *inter-related multi-variate time series* (IMTS) model which minimizes the assumptions that need to be made about the structure of the data (Figure 1.2). In Chapter 7, we experimentally establish the effectiveness of this simplified model in the context of RMT feature extraction.

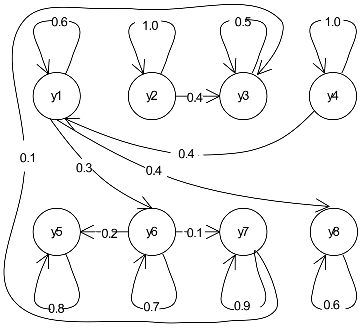
**Definition 1** (IMTS Data Model)**.** *An* inter-related multi-variate time series *(IMTS) data set, $\mathscr{Y} = \langle Y(t), \mathbf{R} \rangle$, is a pair where $Y(t) = \langle Y_1(t), \ldots, Y_m(t) \rangle$ is a multi-variate time series and* **R** *is a dependency/correlation matrix denoting the relationships among the individual time series (variates) in the multi-variate data.*

**Example 1** (Example: Multi-variate Energy Time Series Data Sets)**.** *Smart building energy management systems collect multi-variate temporal data and building models [26]:*

- Observation time series: *These include observations made at different sensors regarding the various energy-related parameters including temperature, heating, ventilation, and air conditioning (HVAC) sensor data.*

- Building models*: These include the spatial geometry describing how different spaces in the building relate to each other.*

In the rest of the paper, we consider two different types of relationships among the variates in $\mathscr{Y}$:



(a) A directed relationship graph

$$\begin{array}{c|cccccccc} & y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 & y_8 \\ \hline y_1 & .6 & 0 & 0 & .4 & 0 & 0 & 0 & 0 \\ y_2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_3 & 0 & .4 & .5 & 0 & 0 & 0 & .1 & 0 \\ y_4 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ y_5 & 0 & 0 & 0 & 0 & .8 & .2 & 0 & 0 \\ y_6 & .3 & 0 & 0 & 0 & 0 & .7 & 0 & 0 \\ y_7 & 0 & 0 & 0 & 0 & 0 & .1 & .9 & 0 \\ y_8 & .4 & 0 & 0 & 0 & 0 & 0 & 0 & .6 \end{array}$$

(b) the corresponding relationship matrix $\mathbf{R}\_directed$

Figure 3.1: A sample relationship graph and the corresponding relationship matrix.

In this thesis, we consider two different types of relationships: among the variates in $\mathscr{Y}$:

**Definition 2** (Variate Dependency Model)**.** *Under the variate dependency model, $Y(t)$ can be described as*

$$Y(t) \quad = \quad \mathbf{R}Y(t-1) + E(t).$$

*Here*

- *$\mathbf{R}$ is a (row-normalized) matrix defining how the values of $Y$ at time $t-1$ impact the values of $Y$ at time $t$, and*
- *$E$ is a time series denoting independent, external inputs on the time series.*

*More specifically, if $i^{th}$ row, $j^{th}$ column of $\mathbf{R}$ is non-zero, then the variate $v_i$ (i.e., the $i^{th}$ time series) is impacted by the value of the variate, $v_j$ (i.e., the $j^{th}$ time series) at the previous time instant.*

**Example 2** (Example: Multi-variate Motion Time Series Data Sets)**.** *The mocap markers (Figure 3.2) are located on the human body and record the markers' coordinate data. The*

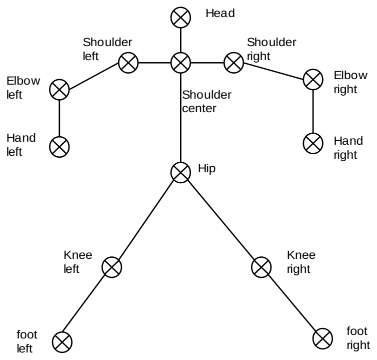


Figure 3.2: Mocap markers' locations on human body.

*markers are inter-dependent on each other due to the inter-connectedness of muscles and tendons, i.e. coordinate changes are observed at time t-1 on the right knee marker. There is a great possibility that coordinate changes on the right foot marker will be observed at time t. Such inter-dependent relations among markers can be described by a relationship graph, which can be further represented by a relationship matrix.*

Figure 3.1 visualizes the relationship graph of a multi-variate time series (with 8 inter-related variates) and the associated relationship matrix. This graph-based representation is a common way of modeling temporal dynamics of multi-variate time series [17, 20, 22, 15].

**Definition 3** (Variate Correlation Model)**.** *Under the variate correlation model, there exists a matrix* $\mathbf{R}$ *such that* $\mathbf{R}[i,j] = \Phi(Y_i,Y_j) \in [0,1]$.

Here, $\Phi$ is an application specific similarity or correlation function. The value of $\Phi(Y_i,Y_j)$ may be computed by comparing (recent) historical data of the time series $Y_i$ and $Y_j$ or may reflect available domain knowledge, such as the distance of the sensors recording the variates or known physical relationships between environmental parameters (such as the amount of cooling and the temperature of a given building zone).

The algorithms presented in the paper are applicable under both relationship models[1] and we use the matrix **R** to denote both relationships. We also assume that **R** is known and fixed; though as we discuss in Chapter 8, this requirement can, in practice, be relaxed.

## 3.2 RMT Feature Extraction Algorithm

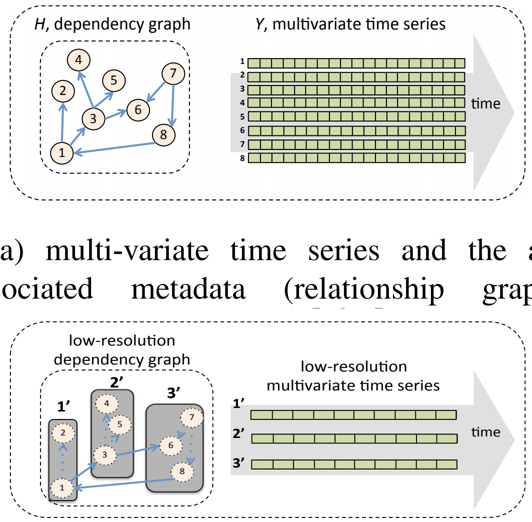
In this paper, we propose an algorithm to extract *robust multi-variate temporal* (RMT) features from inter-related multi-variate time series (IMTS) data sets. This approach has a number of advantages: (a) First of all, the identified salient features are robust against noise and common transformations, such as temporal shifts or dropped / missing uni-variate series. (b) Scale invariance enables the extracted salient features to be robust against variations in speed and enables multi-resolution searches. (c) Also, the temporal and relationship scales at which a multi-variate feature is located give an indication about the *scope* (both in terms of duration and the number of variates involved) of the multi-variate feature.

Since the RMT features can be of different lengths and may cover different number of variates, in order to be able locate features of different sizes, the RMT features are extracted from a scale-space we construct for the given multi-variate time series through iterative smoothing[2]. Intuitively, smoothing of the multi-variate data in time and variates create different resolution versions of the input data and, thus, help identify features with different amount of details both in time and in terms of the number of variates involved (Figure 3.3).

While iterative smoothing techniques are well understood for uni-variate data [8] [42], this is not the case for multi-variate time series. Therefore, before we describe the RMT feature identification and extraction processes, we first propose a novel approach to smoothing a multi-variate time series by leveraging available metadata that describes known relationships among variates.

[1] Thus, without loss of generality, we sometimes focus on the dependency model and, other times, use the correlation model.

[2] This is different from what is known as “multi-variate exponential smoothing”, a forecasting technique where the multi-variate models include the so-called “smoothing parameters” and these are learned to obtain models with a better fit to the data [15].


H, dependency graph
Y, multivariate time series
1
2
3
4
5
6
7
8
time
low-resolution dependency graph
low-resolution multivariate time series
1'
2'
3'
time

(a) multi-variate time series and the associated metadata (relationship graph)

(b) lower resolution view of the same data

Figure 3.3: (a) A multi-variate time series and the associated relationship graph; (b) lower-resolution view of the same data (in terms of both time and variates).

*Metadata Driven Smoothing of a Multi-Variate Time Series*

Let $Y$ be a multi-variate time series. The scale-space of $Y$ is obtained through iterative smoothing across both time and variate relationships, starting with an initial smoothing parameter $\Sigma_0 = \langle \sigma_{time,0}, \sigma_{rel,0} \rangle$ and continuing for $L$ iteration layers obtaining differently smoothed versions of the given multi-variate time series. The values of $\Sigma_0$ and $L$ control the sizes of the smallest and largest features sought in the data (as described in Section 3.2).

**Temporal Smoothing**

Let $\mathsf{Y}_i(t,\sigma)$ indicate a version of the uni-variate time series, $Y_i$, smoothed with parameter $\sigma$:

$$\mathsf{Y}_i(t,\sigma) = G(t,\sigma) * Y_i(t),$$

where $*$ is the convolution operation in $t$ and $G(t,\sigma_{time})$ is the Gaussian function:

$$G(t,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-t^2}{2\sigma^2}}.$$

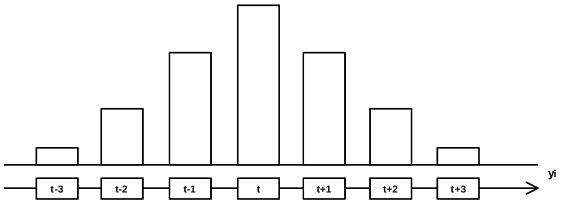


Figure 3.4: Gaussian smoothing in temporal of uni-variate time series $y_i$ at time instant, $t$.

As visualized in (Figure 3.4). Each element in the multi-variate time series is smoothed in temporal by its neighbors in the same variate. We denote $\mathsf{Y}(t,\sigma) = \langle \mathsf{Y}_1(t,\sigma),\ldots,\mathsf{Y}_m(t,\sigma)\rangle$ is a version of the multi-variate time series, where each uni-variate time series is smoothed independently of the rest..

**Relationship Smoothing**

As described above, the temporal smoothing process relies on a convolution operation that leverages the temporal ordering of the time instants in the series. We define the relationship smoothing function relying on an analogous relationship ordering of the variates, described through a relationship distance function $\mathbf{N}$:

**Definition 4** (Relationship Distance Function, $\mathbf{N}$)**.** *Let* $\mathbf{R}$ *be a matrix describing a given dependency/correlation graph. The relationship ordering between the variates is described through the relationship distance function,* $\mathbf{N}$*, where*

- $\mathbf{N}(j,\mathbf{R})$*, for* $j \geq 0$*, is an*

  $m \times m$ $\{0,1\}$*-valued matrix, where for a given variate* $v_i$ *in the relationship graph,* $\mathbf{R}$*, the* $i^{th}$ *row in the matrix,* $\mathbf{N}(j,\mathbf{R}) = 1$

  *only for variates that have relationship distance* $j$ *from the variate* $v_i$*, and*

- $\mathbf{N}(j,\mathbf{R})$*, for* $j < 0$*, is an* $m \times m$ $\{0,1\}$*-valued matrix, where for a given variate* $v_i$*, the* $i^{th}$ *column in the matrix,* $\mathbf{N}(j,\mathbf{R}) = 1$ *only for variates that have relationship distance* $j$ *on the inverted relationship graph, where all relationship edges are inverted (i.e.,* $\mathbf{R}$ *is transposed).*

$$\begin{Bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{Bmatrix}$$

(b) $\mathbf{N}(1,\mathbf{R})$

$$\begin{Bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{Bmatrix}$$

(b) $\mathbf{N}(-1,\mathbf{R})$

Figure 3.5: A sample distance function **N**, with distance $j = 1$ and $-1$. corresponding to the time series and the dependency graph in Figure 3.1 (a).

Intuitively, the cell $[v_1, v_2]$ of the matrix $\mathbf{N}(j,\mathbf{R})$ is true if the variate $v_2$ is within $j$ hopes from $v_1$. When $j$ is positive the hope distance is measured following outgoing edges, whereas when $j$ is negative, incoming edges are followed.

Common applicable definitions of relationship distance include the hop distance (determined by the shortest distance between the nodes on the given graph) or hitting distance [13]. The matrix $\mathbf{N}(j,\mathbf{R})$ can also be defined as $\mathbf{R}_n^j$, where $\mathbf{R}_n$ is the relationship matrix $\mathbf{R}$ where all the diagonal values are set to 0 and non-zero values are set to 1. $\mathbf{N}(1,\mathbf{R})$, and $\mathbf{N}(-1,\mathbf{R})$for the relationship matrix in Figure 3.1 (b) is presented in Figure 3.5.

Given a relationship distance function, we can introduce the concept of (*non-normalized*) relationship smoothing function as follows:

**Definition 5** ( Relationship Smoothing Function)**.** *Let*

- $\mathbf{R}$ *be an $m \times m$ matrix of relationships,*
- $\sigma$ *be a smoothing parameter, and*
- $X = \langle X_1, \ldots, X_m \rangle$ *be an m-vector.*

*If* $\mathbf{R}$ *corresponds to a directed graph, then the* (non-normalized) relationship smoothing function, *$S_{nn}(\mathbf{R},\sigma,X)$, is defined as*

$$S_{nn}(\mathbf{R},\sigma_{rel},X) = G(0,\sigma_{rel})\mathbf{I}X \quad + \quad \sum_{j=1}^{\infty} G(j,\sigma_{rel})\mathbf{N}(j,\mathbf{R})X$$
$$+ \quad \sum_{j=1}^{\infty} G(j,\sigma_{rel})\mathbf{N}(-j,\mathbf{R})X.$$

*If $\mathbf{R}$ corresponds to an un-directed graph, on the other hand, the forward and backward neighborhoods are symmetric:*

$$S_{nn}(\mathbf{R},\sigma,X) = G(0,\sigma)\mathbf{I}X + \sum_{j=1}^{\infty} 2G(j,\sigma)\mathbf{N}(j,\mathbf{R})X.$$

The following example visualizes relationship smoothing:

**Example 3.** *Figure 3.6 shows how we apply Gaussian smoothing over a relationship graph in Figure 3.1. The lower half of the figure shows a node* a *and its forward and backward k-hop neighbors in the relationship graph. As shown in the upper half of the figure, when identifying the contributions of the nodes on* a*, Gaussian smoothing is applied along the hop distance. Since at a given hop distance there may be more than one node, all the nodes at the same distance have the same degree of contribution and the degree of contribution gets progressively smaller as we get further away from the node for which the smoothing is performed.*

Note that the non-normalized smoothing function (for the directional relationship graph) can be equivalently formulated as

$$S_{nn}(\mathbf{R},\sigma,X) = \mathbf{S}_{nn}(\mathbf{R},\sigma)X,$$

where the term $\mathbf{S}_{nn}(\mathbf{R},\sigma)$ can be computed in advance to speed-up the computation of $S_{nn}(\mathbf{R},\sigma,X)$ for different time series vectors, $X$. Moreover, since $G(j,\sigma)$ approaches to 0 quickly as $j$ increases, the term $\mathbf{S}_{nn}(\mathbf{R},\sigma)$ can be approximated efficiently by performing the infinite summations only a finite number, $r$, of times based on $\sigma$ (see Section 3.2 for the relationship of $r$ and $\sigma$).

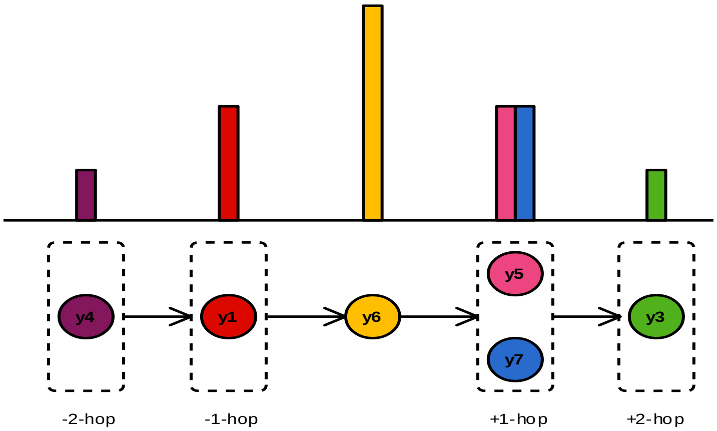


Figure 3.6: Gaussian smoothing along relationship of directed graph in Figure 3.1, for node $y_6$.

Remember from Figure 3.6, however, that, unlike basic Gaussian smoothing, during (non-normalized) relationship smoothing, there may be more than one node at the same distance and all such nodes have the same degree of contribution. As a consequence, the sum of all contributions may exceed 1.0, which means that the smoothing process may undesirably scale the time series. To avoid this, we need a smoothing function, $\mathbf{S}(\mathbf{R},\sigma)$, where the total contribution of all the nodes is normalized back down to 1.0:

**Definition 6** ((Normalized) Relationship Smoothing Function)**.** *Let* $\mathbf{R}$*,* $\sigma$*, and* $X$ *be defined as before. The* (normalized) relationship smoothing function*,* $S(\mathbf{R},\sigma,X)$*, is defined as*

$$S(\mathbf{R},\sigma_{rel},X) = \Big(\mathbf{S}_{nn}(\mathbf{R},\sigma_{rel})X\Big) \div \Big(\mathbf{S}_{nn}(\mathbf{R},\sigma_{rel})1_{(m)}\Big),$$

*where*

- $\mathbf{S}_{nn}(\mathbf{R},\sigma_{rel})$ *is the non-normalized relationship smoothing function corresponding to* $\mathbf{R}$ *and* $\sigma$*,*
- $1_{(m)}$ *is an m-vector where all values are* 1*, and*
- *"÷" is a binary vector operation which applies a pairwise division operation across the elements of two vectors; i.e., if* $C = A \div B$*, then* $\forall i\ C[i] = A[i]/B[i]$*.*

Intuitively, division scales down the contributions for each row in such a way that the total contributions, at most, add up to 1.0.

### Combined TR-Smoothing

We can define time/relationship smoothing (TR-smoothing) of multi-variate time series based on the previous concept of temporal/relationship smoothing functions:

**Definition 7** (TR-Smoothing of a Multi-Variate Time Series)**.** *Let $Y(t) = \langle Y_1(t), \ldots, Y_m(t) \rangle$ be a multi-variate time series and $\mathbf{R}$ be the corresponding dependency/correlation matrix. Let also $\mathbf{S}(\mathbf{R}, \sigma_{rel})$ be the relationship smoothing function corresponding to the matrix $\mathbf{R}$. For a given smoothing parameter, $\Sigma = \langle \sigma_{time}, \sigma_{rel} \rangle$, the TR-smoothed version, $\mathbb{Y}(t, \Sigma)$, of the multi-variate time series $Y(t)$ is defined as*

$$\mathbb{Y}(t, \Sigma) = \mathbf{S}(\mathbf{R}, \sigma_{rel}) * \mathsf{Y}(t, \sigma_{time}),$$

*where, as described earlier, $\mathsf{Y}(t, \sigma_{time})$ is a version of the multi-variate time series, $Y$, where each uni-variate time series is smoothed independently of the rest. More specifically, let $\mathfrak{Y}_t(z, \Sigma)$ denote the version of $\mathsf{Y}$, where at each time instance, $t$, the values $\langle \mathsf{Y}_1(t, \sigma_{time}), \ldots, \mathsf{Y}_m(t, \sigma_{time}) \rangle$ are further smoothed across the relationship space defined by $\mathbf{R}$ using parameter $\sigma_{rel}$:*

$$\mathfrak{Y}_t(z, \Sigma) = S(\mathbf{R}, \sigma_{rel}, \mathsf{Y}_z(t, \sigma_{time})).$$

*Then $\mathbb{Y}(t, \Sigma) = \{\mathbb{Y}_1(t, \Sigma), \ldots, \mathbb{Y}_m(t, \Sigma)\}$ is the multi-variate time series, where the uni-variate time series, $\mathbb{Y}_i(t, \Sigma)$, corresponding to the $i^{th}$ variate is a length $n$ smoothed sequence,*

$$\mathfrak{Y}_1(i, \Sigma); \ldots; \mathfrak{Y}_n(i, \Sigma),$$

*where the value, $\mathfrak{Y}_t(i, \Sigma)$ at time instant $t \in \{1, \ldots, n\}$, is smoothed both in time (in the temporal neighborhood of time instant $t$) and across relationships (in the relationship neighborhood of the $i^{th}$ variate).*

### *Multi-Variate Scale Space Construction*

The first step in identifying multi-variate features is to generate a scale-space representing versions of the given multi-variate series with different amounts of details. In particular, building on the observation that robust localized features are often located where the differences between neighboring regions (also in different scales [39] [8]) are large, we seek RMT features of the given multi-variate time series at the extrema of the scale space defined by the difference-of-the-Gaussian (DoG) series.

More specifically, given a multi-variate time series, $Y = \langle Y_1(t), \ldots, Y_m(t) \rangle$, and the dependency/correlation matrix, $\mathbf{R}$, we compute the difference-of-the-Gaussian (DoG) series, $\mathbb{D}$, from the differences of two nearby scales separated by a multiplicative factor, $\kappa = \langle k_{time}, k_{rel} \rangle$:

$$\mathbb{D}_i(t, \sigma) = \mathbb{Y}_i(t, \kappa\Sigma) - \mathbb{Y}_i(t, \Sigma),$$

where, for a given $\Sigma = \langle \sigma_{time}, \sigma_{rel} \rangle$,

$$\kappa\Sigma = \langle k_{time}\sigma_{time}, k_{rel}\sigma_{rel} \rangle.$$

This repeated smoothing processes for generating the scale-space also produces data needed for obtaining descriptors for features identified at different scales (see Section 3.2).

### Computation of DoG

Let $\Sigma_0 = \langle \sigma_{time,0}, \sigma_{rel,0} \rangle$ be the user provided initial (combined) smoothing parameter and let $s = \langle s_{time}, s_{rel} \rangle$ be a user provided parameter regulating the speed with which time and variate relationships are smoothed. The multi-variate time series is incrementally smoothed (both in time and relationships) starting from the given smoothing parameter $\Sigma_0$, multiplied at each iteration with $\kappa = \langle k_{time}, k_{rel} \rangle$, where $k_{time} = 2^{1/s_{time}}$ and $k_{rel} = 2^{1/s_{rel}}$. As a result, each sequence of $s_{time}$ smoothed-time series corresponds to a doubling of $\sigma_{time}$, also referred to as an “octave” (Figure 3.7); or equivalently to halving of temporal details.

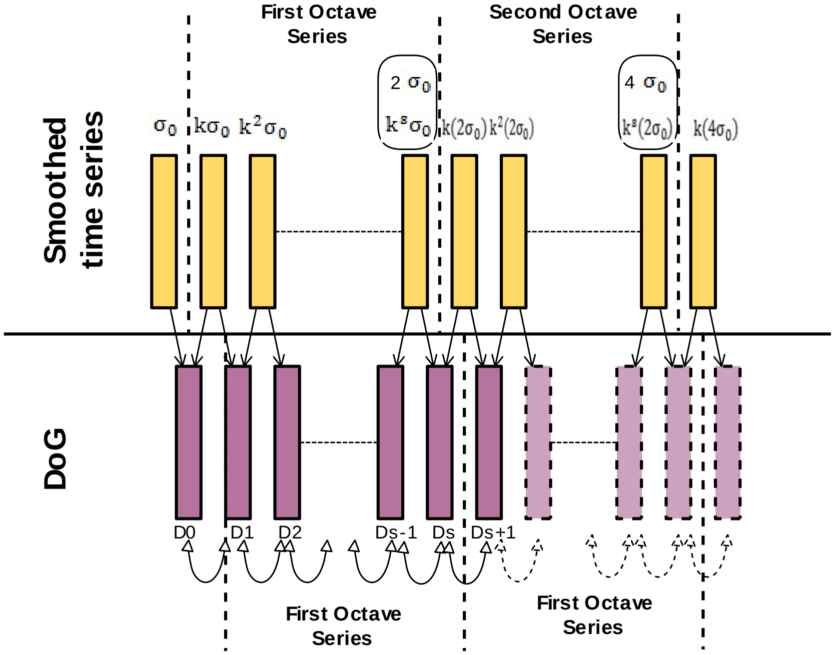


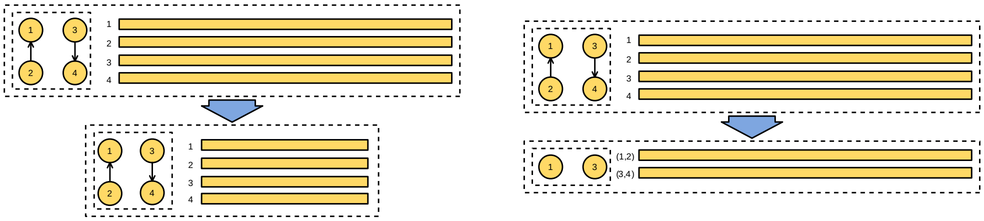
Figure 3.7: Creation of the scale-space and DoG.

Similarly, each sequence of $s_{rel}$ smoothed-time series corresponds to a doubling of $\sigma_{rel}$ and halving of the details across the variates and relationships. The process continues $l$ steps resulting in $l$ layers in the underlying DoG scale-space. Adjacent multi-variate time series are, then, subtracted to obtain the final difference-of-Gaussian (DoG) series.

### Optimizations

Note that, as the multi-variate time series are smoothed, details are lost. As a result, maintaining and using the original length of the time series may be wasteful: instead, it may be more efficient to reduce the length of the time series in a way that matches the amount of details lost during the smoothing process. Thus, as in [39] for images and [8] for univariate time series, we reduce the data resolution to match the loss in details. But, unlike prior work, we leverage the relationship graph to improve the effectiveness of the reduction process for the multi-variate time series.

**Temporal Reduction.**



(a) temporal reduction

(b) variate/relationship reduction

Figure 3.8: Reductions in temporal and relationship resolutions to match the detail losses due to smoothing.

As we can see in Figure 3.7, to produce an octave of DoG series, (not counting the very first time series smoothed with parameter $\sigma_0$), we need to consider $s_{time}+1$ progressively smoothed time series. Moreover, as also seen in the figure, since the scale-space extrema detection involves comparison of each DoG series to the DoG series that come immediately before and immediately after, we need to consider $s_{time}+2$ smoothed time series (including one DoG series before and one DoG series after) to fully cover an octave of DoG series. For example, in Figure 3.7, given the initial time series smoothed with parameter $\sigma_0$, $s+2$ series with smoothing parameters, $k\sigma_0$ through $k^2(2\sigma_0)$ (or equivalently, $k^{s+2}\sigma_0$), are needed to generate the difference of Gaussians series $D_0$ through $D_{s+1}$ that covers the first octave.

Let $Z(t)$ be the first multi-variate series of a new octave (i.e., $(s_{time}+1)^{st}$ series from the beginning of the previous octave). To reduce the amount of work, we reduce the size of $Z(t)$, by resampling it, taking every second time instant of each $Z_i(t) \in Z(t)$; i.e., $\forall_{1 \leq t \leq \lfloor length(Z_i)/2 \rfloor} Z_i'(t) = Z_i(2t)$. The resulting multi-variate series $Z'(t)$ is then used as the input to the subsequent octave.

**Example 4.** *As visualized in Figure 3.8(a), a smoothed multi-variate time series data set with 4 variates is presented on the upper level. To reduce it's size without losing details, we resample by taking every other time instance. The new multi-variate time series (on the lower level), which contains 4 variates that are half the size of the upper one, is used as the input to the subsequent octave.*

**Variate/Relationship Reduction.**

Since the multi-variate data is smoothed both in time and relationships, temporal only reduction would maintain more information than necessary and would potentially waste space and processing time. Variate/relationship reduction reduces resolution in relationship graph and, thus, prevents this waste.

Let $Z(t) = \langle Z_1(t), \ldots, Z_z(t) \rangle$ be the first series of a new octave, where $z$ is the current number of variates, and $\mathbf{R}_{cur}$ be the current relationship matrix. To reduce the amount of work, $Z$ is reduced along the relationships as follows:

Let $\mathscr{C} = \{C_1, \ldots, C_{\lfloor z/2 \rfloor}\}$ be a clustering[3] of the variates in the relationship graph. We reduce $Z$ according to $\mathscr{C}$ as follows:

$$\forall Z_i \in Z \, \forall_{1 \leq t \leq length(Z_i)} Z'_i(t) = \underset{j \in C_i}{AVG}\left(Z_j(t)\right).$$

The relationship matrix is also reduced according to $\mathscr{C}$:

$$\forall_{1 \leq i \leq \lfloor z/2 \rfloor} \forall_{1 \leq h \leq \lfloor z/2 \rfloor} \mathbf{R}'[i,h] = \frac{\sum_{j \in C_i; u \in C_h} (\mathbf{R}_{cur}[j,u])}{\|C_h\|}.$$

The resulting reduced multi-variate time series, $Z'$, and relationship graph, $\mathbf{R}'$, are then used as inputs to the next octave.

**Example 5.** *The Variate/relationship reduction process is visualized in Figure 3.8(b). A smoothed multi-variate time series data set with 4 variates and its variate relationship graph are presented on the upper level. To reduce its size along relationship, we divide the variates into 2 clusters based on the relationship graph and combine the uni-variate time series in the same cluster. In this case, we combine series 1 and series 2; series 3 and series 4 to form the new multi-variate time series, which contains 2 variates and has the same temporal length as original. The new series, on the lower level, is treated as the input to the next octave.*

[3] $\mathscr{C}$ can be obtained either operating directly on the relationship graph and applying a clustering algorithm, such as k-means, or by leveraging domain knowledge (such as reducing the resolution of the space in which sensors are distributed).

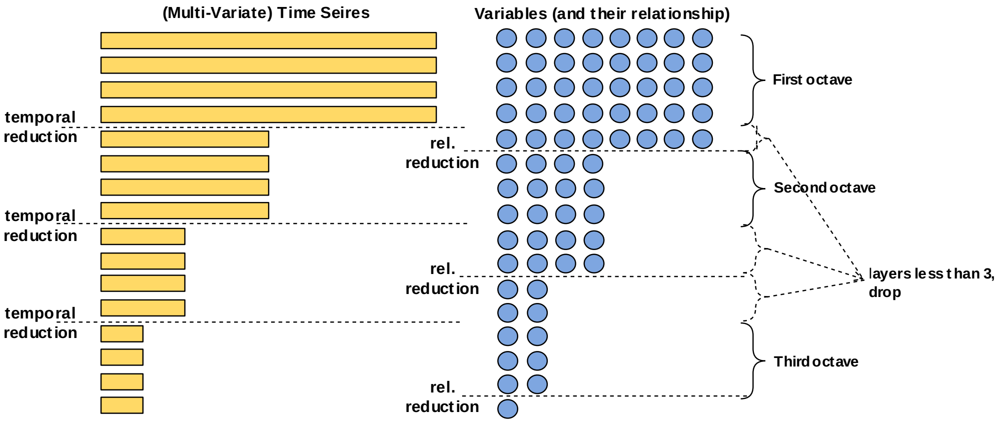


Figure 3.9: Asynchronous reduction of time and variates / relationships.

### Asynchronous Reductions

As shown in Figure 3.9, time reduction and variate/relationship reduction steps and iterations are carried out asynchronously. In this example, $s_{time}$ is smaller than $s_{rel}$, indicating that the multi-variate data set is smoothed faster along time than along the relationships. This might for example be when the length of the time series is much longer than the diameter of the variate relationship graph.

In cases where $s_{time} = s_{rel}$ or when, by coincidence, the temporal and variate / relationship reductions overlap in the same iteration, without loss of generality, we first apply temporal reduction on the given time series, followed by the relationship reduction.

Note that the octaves are achieved by determining the reduction factor in regards to time or relationship. When the DoG series in a single octave is less than 3, we treat it as inefficient because, as we will discuss in the next section, at least 3 DoG layers are needed for searching feature candidates.

### *Identify RMT Feature Candidates*

In order to detect RMT feature candidates using the difference-of-Gaussians (DoG) multivariate series, $\mathbb{D}$, the value of $\mathbb{D}$ for each $\langle i,t,\Sigma \rangle$ triple is compared to its neighbors (both in time and relationships) in the same scale as well as the scales above and below $\Sigma$ and triple, $\langle i,t,\Sigma \rangle$, is selected as a candidate only if it is an extremum; i.e., it is larger or smaller

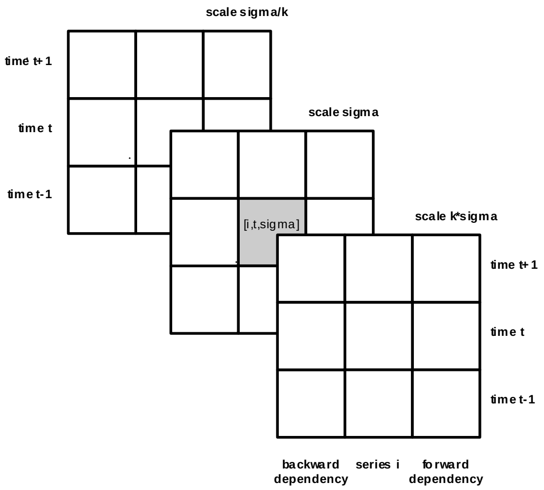


Figure 3.10: 26 neighbors of a triple $\langle i,t,\Sigma\rangle$ (in time, relationship and scale).

than all of them. More specifically, whether the given $\langle i,t,\Sigma\rangle$ triple is a local maximum is identified by comparing $\mathbb{D}_i(t,\Sigma)$ against $max\_neighbor(i,t,\Sigma)$, defined as the maximum of the 26 neighboring triples of $\langle i,t,\Sigma\rangle$ in time, scale, and relationships[4] shown on Figure 3.10:

$$max \left\{ \begin{array}{ccc} \mathbb{D}_i(t-1,\Sigma/\kappa) & \mathbb{D}_i(t-1,\Sigma) & \mathbb{D}_i(t-1,\kappa\Sigma) \\ \mathbb{D}_i(t,\Sigma/\kappa) & & \mathbb{D}_i(t,\kappa\Sigma) \\ \mathbb{D}_i(t+1,\Sigma/\kappa) & \mathbb{D}_i(t+1,\Sigma) & \mathbb{D}_i(t+1,\kappa\Sigma) \\ \mathscr{H}[i,t-1,\Sigma/\kappa] & \mathscr{H}[i,t-1,\Sigma] & \mathscr{H}[i,t-1,\kappa\Sigma] \\ \mathscr{H}[i,t,\Sigma/\kappa] & \mathscr{H}[i,t,\Sigma] & \mathscr{H}[i,t,\Sigma/\kappa] \\ \mathscr{H}[i,t+1,\Sigma/\kappa] & \mathscr{H}[i,t+1,\Sigma] & \mathscr{H}[i,t+1,\kappa\Sigma] \\ \mathscr{H}'[i,t-1,\Sigma/\kappa] & \mathscr{H}'[i,t-1,\Sigma] & \mathscr{H}'[i,t-1,\kappa\Sigma] \\ \mathscr{H}'[i,t,\Sigma/\kappa] & \mathscr{H}'[i,t,\Sigma] & \mathscr{H}'[i,t,\Sigma/\kappa] \\ \mathscr{H}'[i,t+1,\Sigma/\kappa] & \mathscr{H}'[i,t+1,\Sigma] & \mathscr{H}'[i,t+1,\kappa\Sigma] \end{array} \right\},$$

where the term $\Sigma/\kappa$ is defined as

$$\Sigma/\kappa = \left\langle \frac{\sigma_{time}}{k_{time}}, \frac{\sigma_{rel}}{k_{rel}} \right\rangle.$$

[4] The number of neighboring triples may be less than 26 if the triple is at the boundary in terms of time, scale, or relationship graph.

$$\mathbf{F} = \begin{array}{c|cccccccc} & y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 & y_8 \\ \hline y_1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ y_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_3 & 0 & 0.8 & 0 & 0 & 0 & 0 & 0.2 & 0 \\ y_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_5 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ y_6 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_7 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ y_8 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

(a) **F** matrix obtained from **R** in Figure 3.1

$$\mathbf{B} = \begin{array}{c|cccccccc} & y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 & y_8 \\ \hline y_1 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0.5 \\ y_2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ y_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_6 & 0 & 0 & 0 & 0 & 2/3 & 0 & 1/3 & 0 \\ y_7 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ y_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

(b) **B** matrix obtained from **R** in Figure 3.1

Figure 3.11: **F** and **B** matrices corresponding to **R** in Figure 3.1.

and the terms, $\mathscr{H}[i,t,\Sigma]$ and $\mathscr{H}'[i,t,\Sigma]$, denote the values of the forward and backward relationship neighbors of the triple respectively (in an undirected graph, $\mathscr{H}' = \mathscr{H}$). $\mathscr{H}[i,t,\Sigma]$ is defined as

$$\mathscr{H}[i,t,\Sigma] = (\mathbf{F}\mathbb{D}(t,\Sigma))\,[i].$$

Here $\mathbf{F} = \mathit{row\_normalize}(\mathbf{R} - \mathit{diag}(\mathbf{R}))$, accumulates the contributions of all forward related variates. Note that, unlike **R** which also encodes the self-dependency of the variates, **F**, encodes only forward relationships across variates. Moreover, the contributions of the variates with forward relationships are normalized to 1.0 (compare Figure 3.1(a) vs. Figure 3.11(a)).

The term $\mathscr{H}'[i,t,\Sigma]$ is defined similarly,

$$\mathscr{H}'[i,t,\Sigma] = (\mathbf{B}\mathbb{D}(t,\Sigma))\,[i],$$

using backward relationships; $\mathbf{B} = \mathit{row\_normalize}(\mathbf{F}^T)$, accumulates the (normalized) contributions of all backward related variates. Again, unlike **R** which also encodes the self-

(a) a poorly localized feature (b) a better localized feature

Figure 3.12: (a) A poorly localized feature has a large principal curvature, but a small one in perpendicular direction, whereas (b) for a well localized feature, both curvatures are large.

dependency of the variates, **B**, encodes only backward relationships across variates (compare Figure 3.1 (a) and Figure 3.11 (b)). The term $min\_neighbor(i,t,\Sigma)$ is defined similarly (using the *min* function instead of *max*) for identifying the local mimima. The triple $\langle i,t,\Sigma \rangle$ is selected as a candidate if $\mathbb{D}_i(t,\Sigma)$ is larger than $max\_neighbor(i,t,\Sigma)$ or it is less than $min\_neighbor(i,t,\Sigma)$.

*Eliminating Poor Feature Candidates*

Local extrema of DoG can include candidate triples that are poorly localized and the well-localized candidates can be identified by considering the ratio of the eigenvalues of the $2 \times 2$ Hessian matrix, describing the local curvature of the scale-space in terms of the second-order partial derivatives [24, 39]. As shown on Figure 3.12, a poorly defined peek in the difference-in-Gaussians will have a large principal curvature in the scale space in one direction, but a small one in the perpendicular direction. To apply this observation to the problem of identifying poorly localized features in multi-variate time series, we construct the $2 \times 2$ time/relationships Hessian matrix, $\mathfrak{D}^{TR}_{i,t,\Sigma}$, on a given point $\langle i,t,\Sigma \rangle$ at the corresponding scale $\Sigma$:

$$\mathfrak{D}^{TR}_{i,t,\Sigma} = \begin{bmatrix} D_{T,T} & D_{T,R} \\ D_{R,T} & D_{R,R} \end{bmatrix},$$

where

- $D_{T,T} = D_T D_T$ is the second derivate along time at the location and scale of $\langle i,t,\Sigma \rangle$,

- $D_{R,R} = D_R D_R$ is the second derivative along "relationships" at the location and scale of $\langle i,t,\Sigma \rangle$,
- $D_{T,R} = D_T D_R$ is the partial derivative along time of the partial derivate along relationships (at the location and scale of $\langle i,t,\Sigma \rangle$), and
- $D_{R,T} = D_R D_T$ is the partial derivative along relationships of the partial derivate along time (at the location and scale of $\langle i,t,\Sigma \rangle$).

To construct this time/relationships Hessian matrix, we estimate the derivatives along time and relationships by taking differences of neighboring sample points:

$$\begin{aligned}
D_T(i,t,\Sigma) &= \mathbb{Y}_i(t+1,\Sigma) - \mathbb{Y}_i(t-1,\Sigma), \\
D_R(i,t,\Sigma) &= \begin{cases} \Big(\mathbf{F}\mathbb{Y}(t,\Sigma)\Big)[i] - \Big(\mathbf{B}\mathbb{Y}(t,\Sigma)\Big)[i] \\ \qquad \text{for } \textit{directed relationships} \\ \\ \Big(\mathbf{F}\mathbb{Y}(t,\Sigma)\Big)[i] - \mathbb{Y}_i(t,\Sigma) \\ \qquad \text{for } \textit{undirected relationships} \end{cases}
\end{aligned}$$

Note that if, to save work, time series are reduced at each octave as described in Section 3.2, we use the reduced time series and relationship matrices instead.

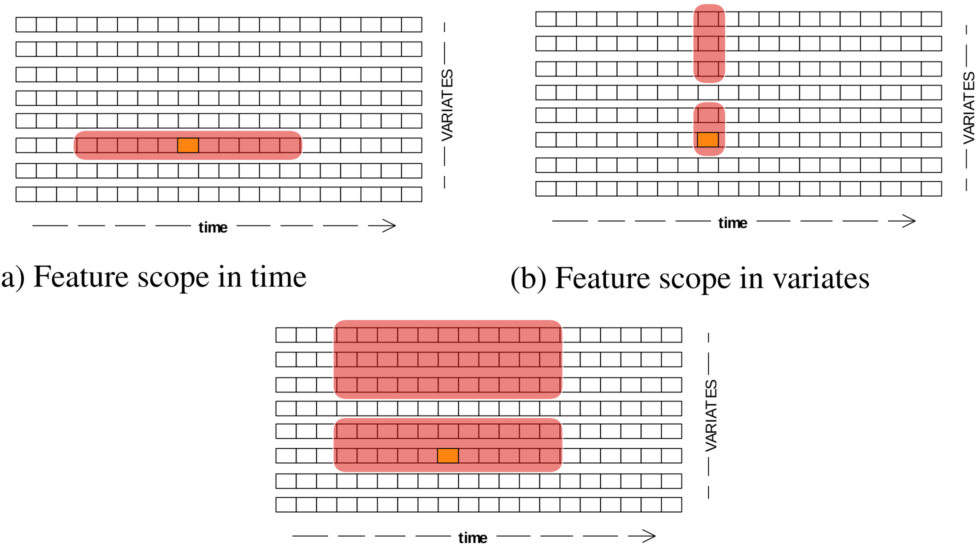
Once this Hessian matrix, $\mathfrak{D}^{TR}_{i,t,\Sigma}$, is constructed for the triple $\langle i,t,\Sigma \rangle$, whether the triple is poorly localized can be checked using eigenvalue-based techniques [24, 39].

### *Extracting RMT Features*

Given a triple $\langle i,t,\Sigma \rangle$ identified in the previous steps, the corresponding feature descriptor is created by considering the gradients around the feature in the scale space.

#### Scope of an RMT Feature

Each multi-variate feature, $\langle i,t,\Sigma \rangle$, has an associated scope, defined by the scale, $\Sigma$, in which it is identified. More specifically, for a given $\Sigma = \langle \sigma_{time}, \sigma_{rel} \rangle$, the radii along time



(a) Feature scope in time

(b) Feature scope in variates

(c) Scope of an RMT feature

Figure 3.13: Scope of an RMT feature: feature center is highlighted in orange.

and scope are $3\sigma_{time}$ and $3\sigma_{rel}$, respectively, since, under Gaussian smoothing, 3 standard deviations would cover $\sim 99.73\%$ of the original observations that have contributed to the identified feature. Intuitively, the larger the scale, the bigger the feature scope.

This means that the four parameters, $\sigma_{temp,0}$, $\sigma_{rel,0}$, $k_{time}$, and $k_{rel}$, can be used for controlling the sizes of the smallest and largest features (in time and relationship spaces) identified in the data. In particular, given a scale-space generation process with $L$ iterations layers,

- the smallest radius of any feature will be $3 \times \sigma_{time,0}$ in time and $3 \times \sigma_{rel,0}$ in the relationship space, and
- the largest feature radius will be
  - $3 \times \sigma_{time,0} \times k_{time}^{L}$ ($\sim 3 \times \sigma_{time,0} \times 2^{\left\lfloor \frac{L}{s_{time}} \right\rfloor}$ ) in time and
  - $3 \times \sigma_{rel,0} \times k_{rel}^{L}$ ($\sim 3 \times \sigma_{rel,0} \times 2^{\left\lfloor \frac{L}{s_{rel}} \right\rfloor}$) in the relationship space.

The identified feature triple, $\langle i, t, \Sigma \rangle$, will form the center of the feature both in time and in relationship. Naturally, as we have seen in Section 3.2, observations closer in time and relationships to the triple will have significantly larger contributions to the feature than the points closer to the boundaries of the scope. Figure 3.2 (a) highlights a sample RMT feature (marked in orange) and its scope in time (highlighted in red). As we discussed, the majority temporal neighbors (more than 99%) that contribute to the feature will be covered. Similarly, as shown in Figure 3.2 (b), the neighbors along relationship that contribute to the feature are also highlighted. Figure 3.2 (c) highlights the scope of a given RMT feature along time and relationship. Note that the RMT feature descriptor, as we will introduce in next section, will also cover the entire highlighted region.

### RMT Feature Descriptor

To describe the RMT features in a form that is indexable and searchable, we rely on high-dimensional gradient histograms.

**Gradient Histograms** If the input data object were a 2D matrix (such as an image), a gradient histogram around given point $\langle x,y \rangle$ on the matrix could be constructed by computing a gradient for each element in the neighborhood of the point [39]; to give less emphasis to gradients that are far from the point $\langle x,y \rangle$, a Gaussian weighting function is often used to reduce the magnitude of elements further from $\langle x,y \rangle$. The resulting gradients are then quantized into $c$ orientations. Finally a $2a \times 2b$ grid is superimposed on the neighborhood region centered around the point and the gradients for the elements that fall into each cell are aggregated into a $c$-bin gradient histogram. This process leads to a feature descriptor vector of length $2a \times 2b \times c$. In the case of multi-variate time series, however, we cannot directly apply these techniques. Instead, we first need to construct an *extractor matrix* to enable the gradient extraction process.

**Extractor Matrix** To identify gradients across time and relationships, we construct a $2N \times 2M$ matrix $W_{i,t,\Sigma}$:

**Definition 8** (Extractor Matrix). *Let $Y = \langle Y_1(t), \ldots, Y_m(t) \rangle$ be a multi-variate time series and $\mathbf{R}$ be a matrix describing how the variates relate each other. Let us be given a triple $\langle i, t, \Sigma \rangle$ and let $\mathbb{Y}_i$ be the time series $Y_{i,\Sigma}$ at scale $\Sigma$; then,*

- *if the relationship graph is directed, then for all $-N < u \leq N$ and $-M < v \leq M$*

$$W_{i,t,\Sigma}[u,v] \quad = \quad \begin{cases} \text{if } v > 0 & \left(\mathbf{F}^v \mathbb{Y}_{i,\Sigma}\right)[t+u] \\ \text{if } v = 0 & \mathbb{Y}_{i,\Sigma}(t+u) \\ \text{if } v < 0 & \left(\mathbf{B}^v \mathbb{Y}_{i,\Sigma}\right)[t+u] \end{cases}$$

- *if the relationship graph is undirected, then for all $-N < u \leq N$ and $0 \leq v \leq M$*

$$W_{i,t,\Sigma}[u,v] \quad = \quad \begin{cases} \text{if } v > 0 & \left(\mathbf{F}^v \mathbb{Y}_{i,\Sigma}\right)[t+u] \\ \text{if } v = 0 & \mathbb{Y}_{i,\Sigma}(t+u). \end{cases}$$

The values of $N$ and $M$ should be selected to roughly cover the scope of the feature; i.e., $N \sim 3\sigma_{time}$ and $M \sim 3\sigma_{rel}$.

**Descriptor Extraction** Given this extractor matrix, $W_{i,t,\Sigma}$, the feature descriptor is created as a $c$-directional gradient histogram of this matrix, sampling the gradient magnitudes around the salient point using a $2a \times 2b$ grid (or $2a \times b$ grid for undirected relationship graphs) superimposed on the matrix, $W_{i,t,\Sigma}$. To give less emphasis to gradients that are far from the point $\langle i, t \rangle$, a Gaussian weighting function is used to reduce the magnitude of elements further from $\langle i, t \rangle$.

This process leads to a feature descriptor vector of length $2a \times 2b \times c$ (or $2a \times b \times c$ for undirected graphs). The descriptor size must be selected in a way that reflects the temporal characteristics of the time series; if a multi-variate time series contains many similar features, it might be more advantageous to use large descriptors that can better discriminate.

## 3.3 RMT Feature Set of a Time Series

Given the above, the salient features of an IMTS multi-variate time series $\mathscr{Y} = \langle Y(t), \mathbf{R} \rangle$ is defined as a set, $\mathscr{F}$, of RMT features, where each feature, $f \in \mathscr{F}$, extracted from $Y(t)$, is a pair of the form, $f = \langle position, descriptor \rangle$:

- $position = \langle i, t, \Sigma \rangle$ is a triple denoting the *position* of the feature in the multi-variate time series, where $i$ is the index of the uni-variate time series $Y_{i,\Sigma}$, at scale $\Sigma$, on which the feature is *centered*, $t$ is the time instant around which the duration of the feature is *centered*, and $\Sigma = \langle \sigma_{time}, \sigma_{rel} \rangle$ is the temporal and relationship scales in which the feature is identified.

- $descriptor$ is a vector of length $2a \times 2b \times c$ for directed relationship graphs and $2a \times b \times c$ for undirected graphs.

This feature set can be used for various applications, including alignment of multi-variate series (as discussed in next Chapter), indexing, visualization, classification, and change detection as discussed next.

## Chapter 4

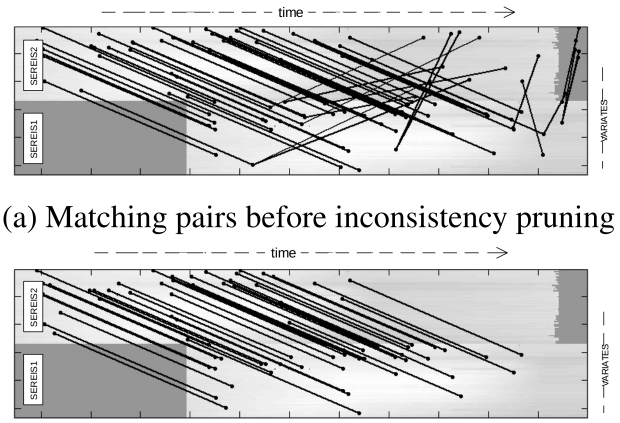
## RMT FEATURE ALIGNMENT

Let us be given two multi-variate time series and let us assume that we are asked to identify how these two series align with each other. One way to achieve this is to locate the best match for each RMT feature of one of the series (let us call this the query series) on the other series (let us call this the data series). As shown in Figure 4.1 (a), however, this strategy may include many conflicting matching pairs that would imply incompatible stretching of time or other inconsistencies. If we want to ensure that the feature matches imply a transformation which potentially stretches time and may also slightly differ in the features that are involved, but does not alter the order of features. We need to eliminate those matching pairs that are less dominant, in terms of the degree of match and temporal / relationship alignments, than those it conflicts with.

Motivated by this observation, we propose a pruning process that involves two steps: (a) matching score calculation and (b) inconsistency elimination.

Note that we can search for different types of alignments between a pair of multi-variate time series:

- Synchronized / asynchronized: this is when we have a common temporal dimension shared by the two time series and we want to preserve the alignments of the two series with respect to this common time dimension, we refer to this as synchronized alignment; otherwise we refer to it as asynchronized alignment.
- Paired / un-paired: this is when we know which variate in one series correspond to which variate in the other series and we want to preserve the mappings between the variates across the two series, we refer to this as the paired alignment, otherwise we refer to it as the unpaired alignment.



(a) Matching pairs before inconsistency pruning

(b) Matching pairs after inconsistency pruning

Figure 4.1: Alignment of two multi-variate time series based on the matching pairs of local RMT features. Each time series contain 53 variates.

In this chapter, we will discuss the strategies for computing pair scores and the elimination strategies for all possible cases.

## 4.1 Overall Matching of a Given Pair of Features

Given two multi-variate time series A and B; and a feature pair, $\left(f_i, f_j\right)$, where $f_i \in A$ and $f_j \in B$, we compute the following scores:

**Similarity score:**

$$\mu_{sim}\left(f_i, f_j\right) = \mu_{desc} \times \left(1 - \triangle_{amp}\left(f_i, f_j\right)\right), \tag{4.1}$$

where $\mu_{desc} = \frac{1}{1+EuclideanDistance\left(f_{i,desc}, f_{j,desc}\right)}$ is the matching score between the descriptors of $f_i$ and $f_j$, and $\triangle_{amp}$ is the percentage difference between the overall amplitudes of the features within their corresponding scopes. This formula reflects the observation that we prefer matching feature pair which have similar descriptors and similar average amplitudes.

**Alignment score**:

$$\mu_{align}\left(f_i, f_j\right) = \frac{\left(\mu_{scope,time} + \mu_{scope,depd}\right)/2}{\left(\mu_{cdist,time} + 1\right) \times \left(\mu_{cdist,depd} + 1\right)}, \tag{4.2}$$

here,

$$\mu_{cdist,time} = \begin{cases} |center(f_i) - center(f_j)| & ,synchronized \\ 1 & ,asynchronized \end{cases} \tag{4.3}$$

$$\mu_{cdist,depd} = \begin{cases} \frac{|f_{i\_set} \cap f_{j\_set}|}{|f_{i\_set} \cup f_{j\_set}|} & ,paired \\ 1 & ,un-paired \end{cases} \tag{4.4}$$

where $\mu_{scope,time}$, $\mu_{scope,depd}$ are the average time / dependency scope of $f_i$, $f_j$ respectively; $\mu_{cdist,time}$ describe how close the two features are in terms of time; and $\mu_{cdist,depd}$ indicates the relationship similarity between two features using Jaccard set similarity [29], where $f_{i\_set}$, $f_{j\_set}$ correspond to the variates that are involved in $f_i$ , and $f_j$ respectively.

Given $\mu_{sim}$ and $\mu_{align}$, we can further combine these two scores as $\mu_{comb}(f_i, f_j)$ which we computed using F-measure:

$$\mu_{comb}(f_i, f_j) = 2 \times \frac{\mu_{align}(f_i, f_j) \times \mu_{sim}(f_i, f_j)}{\mu_{align}(f_i, f_j) + \mu_{sim}(f_i, f_j)}. \tag{4.5}$$

Note that this feature pair score will be used in computing similarity score of time series.

## 4.2 Temporal Inconsistency Pruning

We call two pairs of matches temporally inconsistent, if the temporal relationships between the features in one time series are inconsistent with the temporal relationships between the corresponding features in the second series. In this section we will consider all matching pairs in descending order of their $\mu_{comb}$ scores and follow the inconsistency definition in sDTW [8]:

- Given a feature matching pair $M = \langle (st_{1,i}, end_{1,i}); (st_{2,j}, end_{2,j}) \rangle$, in which $(st_{1,i}, end_{1,i})$ and $(st_{2,j}, end_{2,j})$ correspond to the scopes of the two involved features in query series and data series.

- We *attempt* to insert the time points $st_{1,i}$ and $end_{1,i}$ into a list, $temporalscope\_boundary\_order_1$, which is in decreasing order of time. Similarly we *attempt* to insert the time points $st_{2,i}$ and $end_{2,i}$ into the list, $temporalscope\_boundary\_order_2$, which is also in decreasing order of time.
- Let $rank(st_{1,i})$, $rank(st_{2,i})$, $rank(end_{1,i})$, and $rank(end_{2,i})$ be the corresponding ranks of the four time points in their respective time ordered lists.
- If $rank(st_{1,i}) = rank(st_{2,i})$ and $rank(end_{1,i}) = rank(end_{2,i})$, then we confirm the insertion and we keep the pair.
- Else, we drop the pair and eliminate these scope boundaries from consideration.

## 4.3 Variate-Relationship Inconsistency Pruning

As we have argued, each RMT feature includes a variates set, which we call it the scope in relationship (Section 3.2). Here we denote $set_{1,i}$, $set_{2,i}$ as the variate scopes of two RMT features in query series. We consider three distinct scenarios for $set_{1,i}$ and $set_{2,i}$ (as we see in Table 4.2, we cluster the 13 relationships between a pair of intervals [2] into 3 symmetric relationships):

Table 4.1: Three distinct variate relationships between two RMT features' variate scopes.

| **Relation** | **Interpretation** |
|---|---|
| (a). Disjoint | $set_{1,i}$ and $set_{2,i}$ have no common variates |
| (b). Intersect | $set_{1,i}$ and $set_{2,i}$ are **partially** intersect with each other |
| (c). Equal/Contain | $set_{1,i}$ equals to $set_{2,i}$, or $set_{1,i}$ contains $set_{2,i}$, or $set_{2,i}$ contains $set_{1,i}$ |

We call two pairs of matches inconsistent, if the variate relationships between the features in one time series are inconsistent with the variate relationships between the corresponding features in the second series. Again, we consider all matching pairs in descending order of their $\mu_{comb}$ scores:

- Given a matching pair, $M = \langle set_{2,i}; set_{2,j} \rangle$, and a matching pair list that are committed, *commit_list*, we compare $M$ with every matching pairs on *commit_list*. We denote $N = \langle set_{1,i}; set_{1,j} \rangle$ as one of the matching pairs on *commit_list*.

- Get the relation , $r_1$, between $set_{2,i}$ and $set_{1,i}$ and relation, $r_2$, between $set_{2,j}$ and $set_{1,j}$ according to Table 4.1.

- If $r_1 = r_2$, then we confirm that matching pairs $M$ and $N$ are not conflict to each other. And we continue compare $M$ with other matching pairs on *commit_list*.

- After comparing $M$ with all the matching pairs on *commit_list*, if no conflict occurs, we commit $M = \langle set_{2,i}; set_{2,j} \rangle$ by inserting it into *commit_list*. Otherwise, we prune it as inconsistent matching pair.

Matching pairs are considered in descending order of their $\mu_{comb}(f_i, f_j)$ scores so that the dominant matching pairs are kept while less dominant pairs are eliminated when conflict occurs. Figure 4.1 (b) shows the maintained matching pairs after the elimination of inconsistencies.

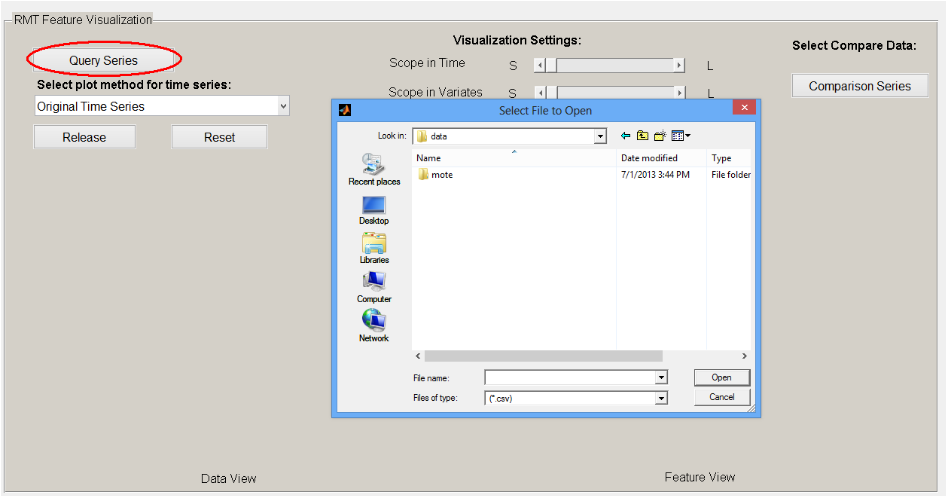
Chapter 5

# RMT FEATURE VISUALIZATION

Since RMT features contain local information (position and scope), a better interpretation of the given multi-variate time series can be made by properly visualizing these features. Therefore, we build a RMT feature visualization system to help users understand a given multi-variate time series and to compare two multi-variate time series at once.

## 5.1 RMT Feature Selection Interface

In this interface, a user can select and explore a multi-variate time series data set and the RMT features extracted from it. For the user, the first step is to select a multi-variate time series using the multi-variate time series selection interface (shown on Figure 5.1). Once the time series is selected, the user is provided with two screens, as shown in Figure 5.2 (a). The left-hand side view provides the standard curve-plot style visualization of all individual variates, in which the x-axis corresponds to the time and the y-axis corresponds to the amplitude. The right-hand side window provides a heat-map style visualization, in which the y-axis corresponds to the different variates and the amplitude is denoted by the different shades.



Select plot method for time series: Original Time Series

Look in: data

| Name | Date modified | Type |
|---|---|---|
| mote | 7/1/2013 3:44 PM | File folder |

File name:

Files of type: (*.csv)

Figure 5.1: RMT feature selection interface (click button in red circle to select multi-variate time series).

The user can use a mouse to interact with both sides of the interface. When the user moves the mouse along the time-axis on the curve-plot, a temporal-focus interval (whose boundaries are marked by the blue-dashed, vertical lines, and whose size is selected by the user) moves along the time axis. Moreover, on the heat-map visualization, all RMT features whose temporal scopes intersect with the temporal-focus interval are displayed as shown in Figure 5.2 (a). The user can also interact with the heat-map interface on the right. As shown in Figure 5.2 (b), as the user moves the pointer along the right-hand side window, only the features whose scopes contain the pointer are highlighted and the variates involved in these features are displayed on the plot-style window on the left.

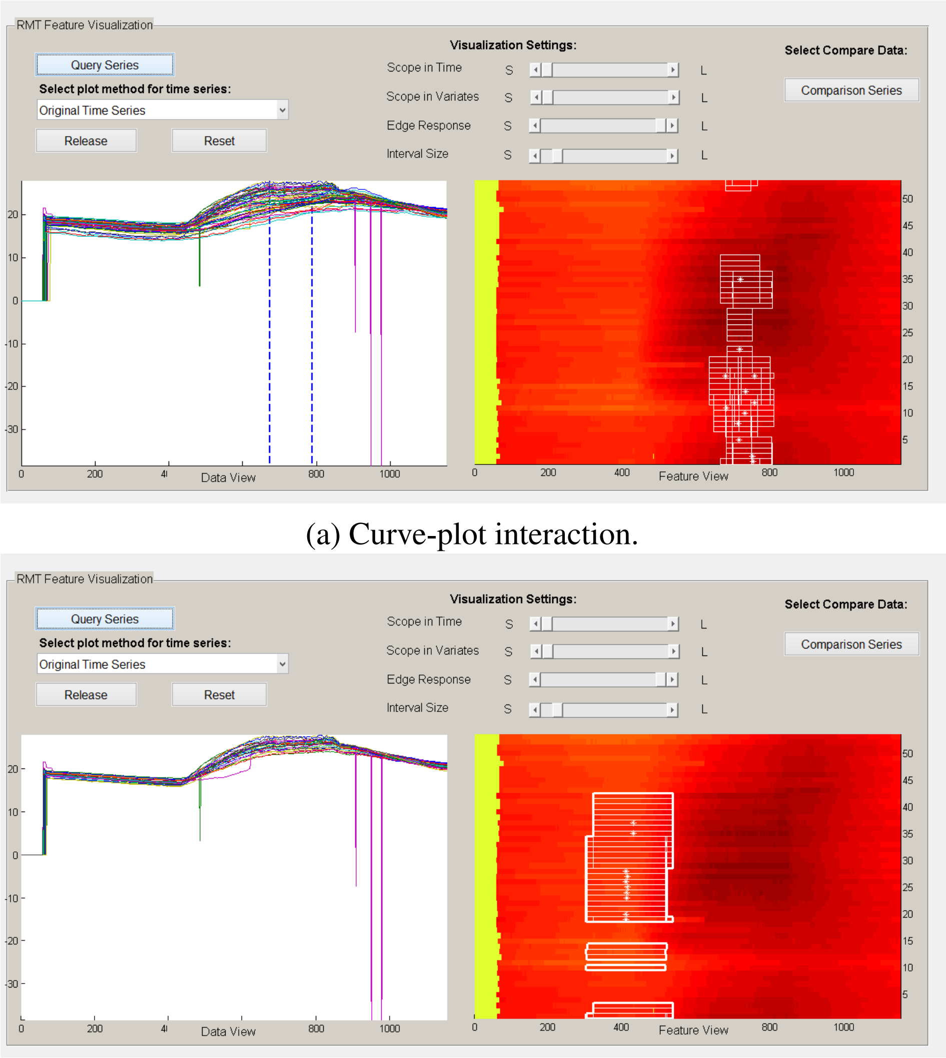
Note that the interface also allows the user to filter features based on their sizes in time and the number of variates they involve, as well as on how well-defined and well-localized the features are on the time series.

As the user explores the RMT features on the time series using this interface, they may also choose to study a particular feature in more details. When the user clicks on the center of a feature of interest, information of the selected feature will be displayed in the feature visualization window, which we will introduce in the next section.

## 5.2 RMT Feature Interface

Given a feature, $f = \langle position, descriptor \rangle$, where $position = \langle i, t, \Sigma \rangle$. The position of the center, in scale $\Sigma$, is determined by $i$ and $t$. As described in Chapter 3, $i$ is the index of the uni-variate time series $Y_i$, in which the feature is *centered* and $t$ is the time instant around which the duration of the feature is *centered*.

In the detailed RMT feature visualization interface, the selected feature is displayed with the ”center” of the feature being marked as ”*”, the temporal scope being marked by white dashed-lines, and variates that are involved being marked as gray-dashed lines. The temporal scope corresponds to the temporal duration of the feature, in which most contributions are covered. The relationship scope corresponds to all of the uni-variate time series



Select plot method for time series: Original Time Series

(a) Curve-plot interaction.

(b) Heat-map interaction.

Figure 5.2: RMT feature selection interface. Berkeley Mote Data set: 1024 readings in temporal and 53 variates.

that are involved in the feature.

For better understanding of the relationship among the variates involved in the feature, this interface allows the user to shrink and expand the feature around the center variate(s) based on the hop distance. Figure 5.4 (a) shows all variates involved in the feature with a hop distance lesser than or equal to 5, whereas Figure 5.4 (b) shows variates with a hop distance lesser than or equal to 1.

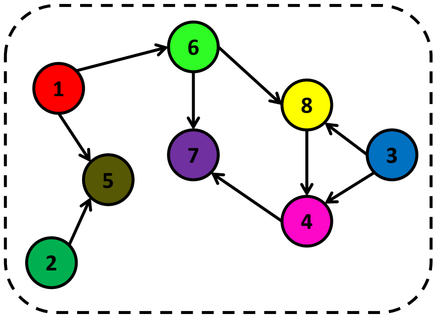
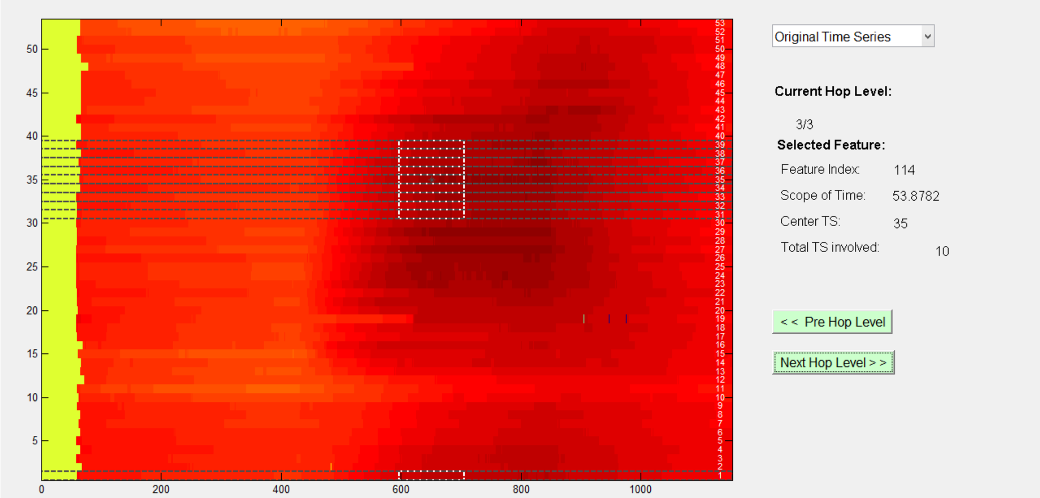


Figure 5.3: A sample relationship graph, R.

**Example 6.** *Figure 5.3 shows a sample relationship graph. Suppose we have an RMT feature with variate $v_8$ as its center. Feature center, $v_8$, is treated as 0-hop; uni-variate time series, $v_3, v_4, v_6$, that have only one step away from the center $v_8$, are 1-hop; and uni-variate time series, $v_1, v_7$, that have two steps away from the center $v_8$, are 2-hop neighbors of $v_8$.*

## 5.3 RMT Feature Matching

The visualization interface also allows the user to select two multi-variate data series (a query series and a data series) and explore them in tandem, based on the matching RMT features. As shown in Figure 5.5, when the user moves the mouse over the query heat-map on the left, the features that the mouse highlights will be shown in that window. Moreover, in the heat-map corresponding to the data series on the right, the best matching features to those highlighted on the query heat-map will also be shown. As before, the corresponding variates involved in the data will also be highlighted in the plot-style views.



Current Hop Level:
3/3

Selected Feature:

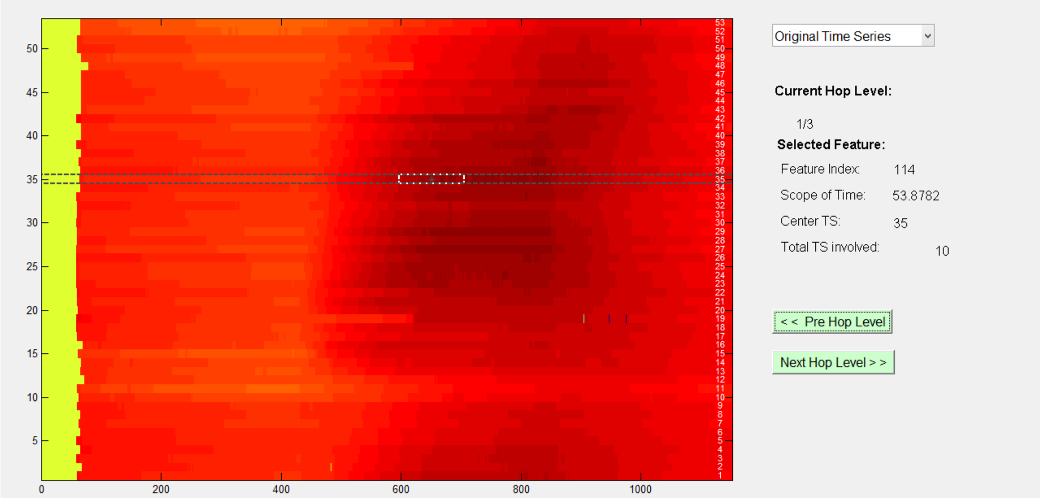
Feature Index: 114

Scope of Time: 53.8782

Center TS: 35

Total TS involved: 10

(a) A sample RMT feature with all involved variates shown.



Current Hop Level:
1/3

Selected Feature:

Feature Index: 114

Scope of Time: 53.8782

Center TS: 35

Total TS involved: 10

(a) A sample RMT feature with only feature center variate shown.

Figure 5.4: RMT feature interface. Berkeley Mote Data set: 1024 readings in temporal and 53 variates.

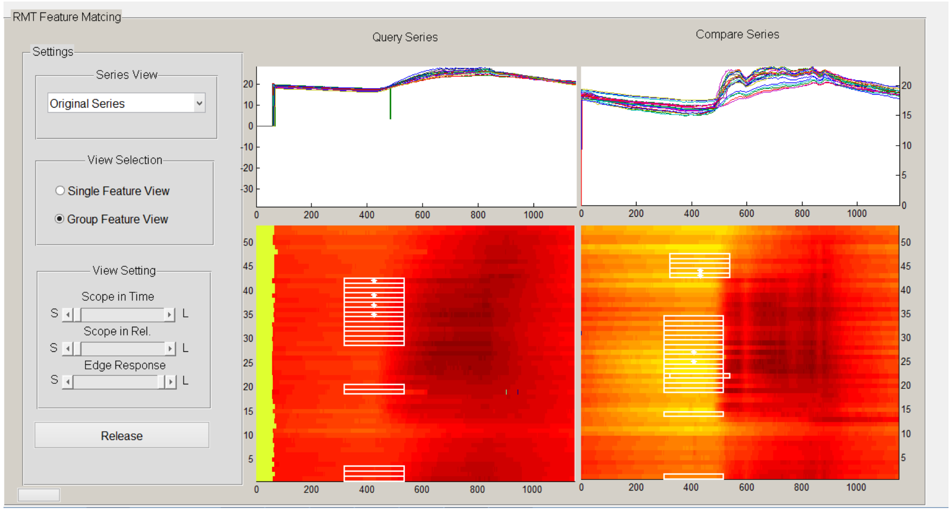


Figure 5.5: RMT feature selection interface. Berkeley Mote Data set: 1024 readings in temporal and 53 variates.

Chapter 6

# EXPERIMENTAL SETUP

In this chapter, we introduce the setup we used for evaluating the RMT features and RMT feature extraction algorithms presented in this thesis. We introduce the settings of the experiments, the data sets we will use, and the evaluation measures.

## 6.1 Settings

All experiments were run on 4-core Intel Core i5-2400, 3.10GHz machines with 8GB memory, running 64-bit Windows 7 Enterprise. The codes were executed using Matlab 7.11.0 (2010b).

## 6.2 Data Sets

We use two distinct data sets in the experiments: Berkeley Mote data set and Mocap Data set. High level characteristics of the two data set are presented in Table 6.1.

### *Berkeley Mote Data Set*

Berkeley mote data set [6] consists of sensor readings between February 28th and April 5th, 2004 from a set of 54 Mica2Dot sensors that are spatially distributed in a $40m \times 30m$ laboratory environment as shown in Figure 6.1. Each sensor is treated as a single variate.

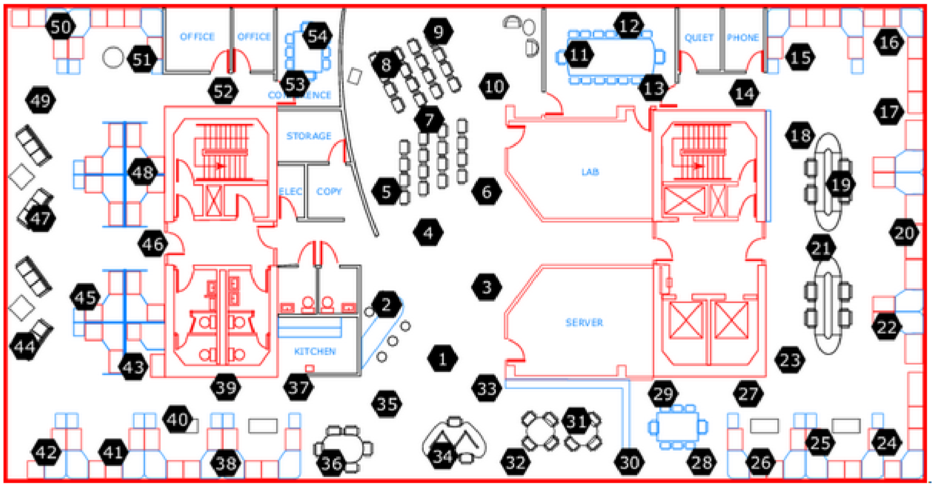


Figure 6.1: Berkeley Mote Sensors arrangement [6].

Table 6.1: Characteristics of the data sets.

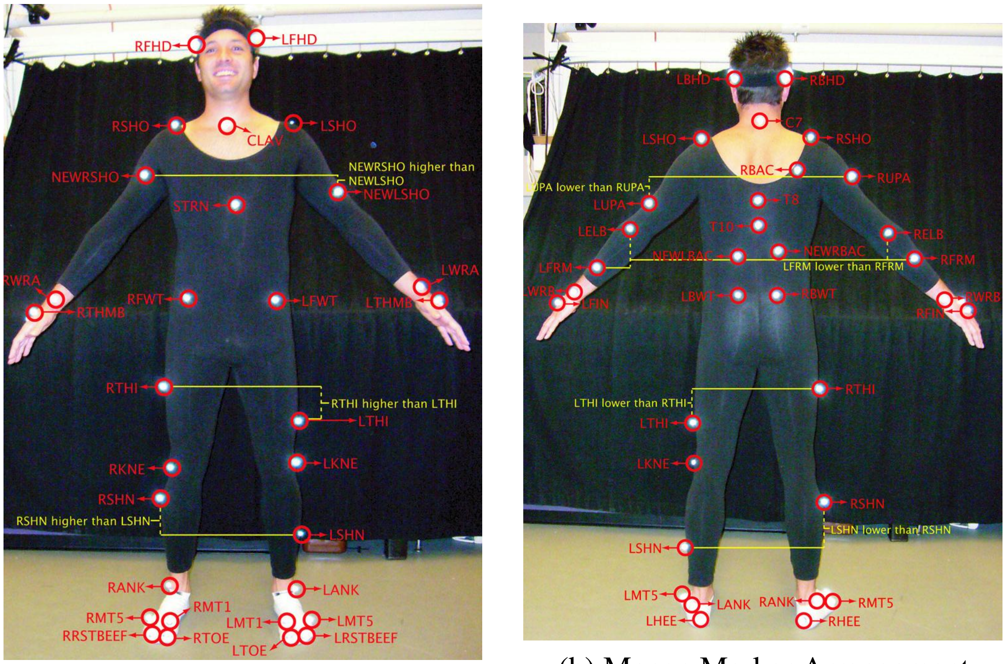
| **Berkeley motes data set** | | |
|---|---|---|
| # classes | # variates | series length |
| 1 | 53 | variable (576 to 1440) |

| **Mocap data set 1** | | |
|---|---|---|
| # classes | # variates | series length |
| 8 | 62 | $\sim$130 to $\sim$1000 |

Sensor 5 is discarded due to bad data reading. Thus the total number of variates for Berkeley Mote data set is 53. The sensors collect topology information, along with humidity, temperature, light and voltage data. We chose the humility data over the 30 days to generate the multi-variate time series data set. The series length is variable in the sense that we can choose different observation intervals (0.5 minute, 1.25 minutes, 2.5 minutes etc.). The smaller the time interval that we choose, the longer the multi-variate time series will be. The spatial distribution of the motes in the laboratory is used to create the underlying correlation matrix. Each pair of sensors within 6 meters is assumed to be correlated with each other, which gives an *intentionally rough* indication of sensor correlation as it ignores other environmental characteristics.

### *Mocap Data Set*

The Mocap data set [14] is created using 41 markers placed on the human body in order to capture coordinate data for different types of motions.The coordinates of the markers are collected as the human actors perform various motions. The so-called ASF / AMC data contain angle information from 62 joints which means that we have 62 variates in the mocap multi-variate time series data set. The relative positions of the joints on the human body are used to create the underlying correlation matrix. More specifically, joints are mapped in 2D space, in such a way that joints that closely coordinates with each other will have smaller distance. For example, since the joints on the left hand will display similar variates due to the interconnectedness of muscles and tendons, they will be mapped closely to each other than other joints.



(a) Mocap Marker Arrangement (Front)

(b) Mocap Marker Arrangement (Back)

Figure 6.2: Marker arrangement of Mocap markers. Figure provided by CMU Mocap lab [14].

## 6.3 Alternative Algorithms

In the experiment, we compare the proposed RMT feature extraction algorithm with two alternative feature extraction algorithms: the UNI feature extraction algorithm for uni-variate time series [8] and the SVD fingerprint. We also consider direct use of the DTW for measuring differences between pairs of multi-variate time series:

### Local, uni-variate, paired (UNI)

We treat each variate as an individual time series and assume that the pairing of the variates in the query and in the database are known in advance.

Given two multi-variate time series, $A$ and $B$, their distance is computed as

$$\underset{A_i \in A, B_i \in B}{\mathrm{AVG}} \left( \underset{f_j \in A_i}{\mathrm{AVG}} \, mindist(f_j, B_i) \right),$$

where $mindist()$ returns the smallest distance (in terms of Euclidean distance) between the given feature $f_j$ in the uni-variate series $A_i$ and any feature in the series $B_i$.

### Raw data, paired (DTW)

In classification task where we compare whole time series to each other, we also use a raw data based strategy where distances are computed through DTW algorithm. Similar to the strategy for paired UNI, when comparing two time series by DTW algorithm, we assume that we know which variate in one series corresponds to which variate in the other.
Given two multi-variate time series, A and B, the degree of match is computed as

$$\underset{A_i \in A, B_i \in B}{\mathrm{AVG}} DTW(A_i, B_i),$$

where $DTW()$ returns the DTW distance between the uni-variate series $A_i$ and corresponding uni-variate series $B_i$. For this purpose, we used the DTW code available at [33].

### Global, multi-variate, non-paired (SVD)

Given two multi-variate time series, $A$ and $B$ and their SVD decompositions, $A = U_A S_A V_A^T$ and $B = U_B S_B V_B^T$, the distance is computed as

$$\underset{u_j \in U_A}{\mathrm{AVG}} mincolumndist(u_j, U_B),$$

where $u_j$ is a column vector in $U_A$ and $mincolumndist(u_j, U_B)$ returns the smallest matching distance (in terms of Euclidean distance) between the column vector $u_j$ and any column vectors in $U_B$. Note that this feature does not need to assume that the variate pairings are known in advance.

### Local, multi-variate, (RMT)

Here the proposed RMT algorithm will be used. Given two multi-variate time series, A and B, the distance is computed as

$$\underset{f_j \in A}{\mathtt{AVG}}\, mindist(f_j, B),$$

where $mindist()$ returns the smallest distance (in terms of Euclidean distance) between the given feature $f_j$ in the multi-variate series $A$ and any feature in $B$. Against the non-paired strategy SVD, we use non-paired RMT. Whereas against paired strategies UNI and DTW, we use paired RMT to measure the distance; in the case of pairing multi-variate feature matches are ignored unless at least 50% of the variates are common.

We also consider an alternative similarity measure for multi-variate time series based on RMT features. In this case, we consider the pairwise alignment scores of a consistent subset of RMT features for the two series being compared.

In this case, given two multi-variate time series A and B, and a set $consistent\_matches_{A,B}$, of matching pairs with their corresponding $\mu_{comb}$ scores, we compute the overall matching score of match as:

$$MatchScore_{A,B} = \sum_{\left(f_i, f_j\right) \in consistent\_matches_{A,B}} \mu_{comb}\left(f_i, f_j\right)$$

We evaluate this alternative in Chapter 7.

## 6.4 Evaluation Tasks

We consider two types of tasks for assessing the effectiveness of extracted RMT features: (a) classification tasks and (b) partial search tasks.

### *Classification Tasks*

For the classification task, we use the Mocap data set where the time series are pre-labeled based on activity types (Table 6.1). As the accuracy measure, we use top-5 precision; i.e, the ratios of series that are of the same class as the query among the top 5 nearest neighbors.

### *Partial Search Tasks*

In the case of Berkeley mote data, for partial search task, observations for each day is treated as a different multi-variate series. Partial time series search queries are generated by picking a random date from the data set and using that series for two queries:

- *Temporal snippet search:* a random time interval during the day is selected and the rest of the series are cropped.
- *Sensor subset search:* sensors in a random portion of the lab space are selected and the rest are dropped.

Accuracy is measured by checking, in the result, the rank of the time series selected to formulate the query. The closer the rank is to 1, the more accurate is the result. Accuracy results are reported both as mean; median accuracy is also reported when outliers skew the mean. A similar process is also used for the Mocap data. One key difference is that in the Mocap data set, multi-variate series are labeled with the type of human motions. Therefore, accuracy is not only measured by checking the rank of the query, but also the average rank of the series with the same label as the query motion.

# Chapter 7

# EXPERIMENTAL RESULT

## 7.1 Classification Task Evaluation

Table 7.1: Top-5 classification accuracy (Mocap data set).

| **Top-5 Precision Classification Accuracy** | | | | | | |
|---|---|---|---|---|---|---|
| Class | # series | RMT | SVD | RMT | UNI | DTW |
| | | (non-paired) | | (paired) | | |
| climb | 18 | 83.3% | 52.2% | 88.9% | 82.2% | 68.9% |
| dribble(basketball) | 14 | 54.3% | 28.6% | 87.1% | 47.1% | 84.3% |
| jumping | 30 | 99.3% | 82.0% | 100% | 98% | 100% |
| running | 19 | 92.6% | 100% | 93.3% | 100% | 100% |
| salsa | 30 | 94.0% | 59.3% | 97.4% | 100% | 87.1% |
| soccer | 6 | 73.3% | 30.0% | 63.3% | 93.3% | 96.7% |
| walk | 36 | 100% | 89.4% | 100% | 100% | 100% |
| walk(uneven terrain) | 31 | 100% | 58.7% | 100% | 98.7% | 98.7% |
| Overall | 184 | 92.2% | 88.8% | 95.4% | 93.5% | 93.3% |

(a) Classification Accuracy

| **Classification Time** | | | | | |
|---|---|---|---|---|---|
| | (non-paired) | | (paired) | | |
| | RMT | SVD | RMT | UNI | DTW |
| Total feature extraction time for the data set | 886.4s | 15.2s | 886.4s | 95.7s | NA |
| Pairwise distance computation time for the data set | 375.8s | 88.0s | 538.3s | 48.6K s | 4Ms |
| Total cost | 1.3K s | 103.3 s | 1.4K s | 48.8K s | 4M s |

(b) Classification cost (in seconds)

Table 7.1 (a) compares the classification accuracy (Mocap data set) of the proposed RMT feature extraction algorithm with three alternative algorithms, SVD, UNI and DTW. Here we use RMT with *mindist*() (as in Section 6.3) to calculate the distance between two given time series, in which inconsistency are not pruned and series are considered as asychronized. Table 7.1 (b) compares the classification cost for each corresponding algorithm in terms of feature extraction cost, feature matching cost and total cost. Algorithms can be further divided into two categories: non-paired algorithms: non-paired RMT, SVD; and paired algorithms: paired RMT, UNI, DTW.

Table 7.2: Default configuration for RMT, UNI, and SVD (relevant subset of this is also used as the configuration for the uni-variate feature extraction [8]).

| **RMT** | |
|---|---|
| # iterations, $L$ | 6 |
| # of octaves, $o$ | 2 |
| smallest temporal feature radius ($3\sigma_{time,0}$) | $\sim 15$ |
| smallest rel. feature radius ($3\sigma_{rel,0}$) | $\sim 2$ |
| candidate prunning threshold, $\omega_{\top}$ | 10 |
| descriptor size, $2a \times 2b \times c$ | $(4 \times 4 \times 8 =)$ 128 |
| relationship reduction algorithm | k-means |
| **UNI** | |
| # iterations, $L$ | 6 |
| # of octaves, $o$ | 2 |
| smallest temporal feature radius ($3\sigma_{time,0}$) | $\sim 15$ |
| descriptor size | 128 |
| **SVD** | |
| degree of energy preservation | 95% |

As we can see, paired RMT provide best overall top-5 precision accuracy result. Non-paired RMT perform almost as good as other pairing algorithms and it can be used when the pairing information is not available. UNI and DTW provide good classification accuracy, but the overall costs are much higher than the proposed RMT algorithm. From these results, we confirm that RMT feature extraction algorithm is highly effective and efficient in the classification tasks. Note that the configurations for RMT, SVD and UNI are presented on Table 7.2.

## 7.2 Feature-based Partial Time Series Search Task Evaluation

In this section, we first discuss properties of the proposed RMT feature. Afterwards, we compare (non-paired) RMT, (non-paired) SVD, and (paired) UNI by partial series search tasks on both Berkeley Mote and Mocap data sets. At last, we analyze the effectiveness of the proposed RMT feature alignments searching strategies, as in Chapter 4, by partial query tasks.

*RMT Feature Properties*

**Benefit of Leveraging Metadata (Known Variate Relationships) during Feature Extraction:** Figure 7.1 (a) presents mean and median result ranks (aggregated for all experimented configurations for the Berkeley mote data set) for RMT leveraging the spatial distribution of sensors and the version of RMT where the correlation matrix **R** is assigned randomly, ignoring the underlying sensor distribution. This chart confirms that RMT is able to leverage the correlation information for the variates to identify highly effective features.

RMT algorithm returns on the average 99.3 features when using the correct relationship matrix and only 59.2 features when using the random relationship matrix. A paired t-test shows that there is around 0 probability that this difference in the number of features is by chance. This confirms the observation in Section 7.3 that significant changes in the relationship structure of the data would lead to statistically significant changes in the number of RMT features.

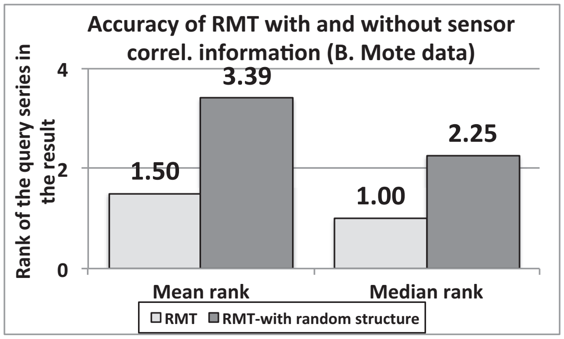


Figure 7.1: Impact of correlation information.

**RMT Feature Extraction Time:** Figure 7.2 shows how the feature extraction work is distributed over sub-tasks. As we can see, most of the time is spent on easily parallelizable tasks, extrema detection and descriptor generation.

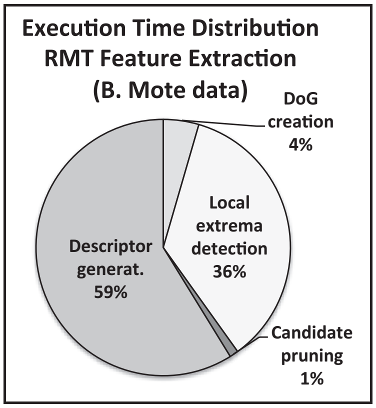


Figure 7.2: RMT feature extraction time.

Table 7.3: Average number of features, feature vector lengths, and extraction times for different feature types.

| | **Berkeley motes data set** | | | **Mocap data set** | | |
|---|---|---|---|---|---|---|
| | RMT | UNI | SVD | RMT | UNI | SVD |
| # features | 99.3 | 12818.6 | 16 | 393.8 | 26712 | 16.8 |
| length | 128 | 128 | 576 | 128 | 128 | 800 |
| ext. time | 1.6s | 0.68s | 0.02s | 4.9s | 0.52s | 0.08s |

Table 7.4: Matching time for a pair of multi-variate series (excluding feature extraction – see Figure 7.3 for the one-time offline feature extraction costs).

| **Berkeley motes data set** | | | **Mocap data set** | | |
|---|---|---|---|---|---|
| RMT | UNI | SVD | RMT | UNI | SVD |
| 0.001sec | 0.5sec | 0.002sec | 0.011sec | 1.43sec | 0.003sec |

**Feature Characteristics:** Table 7.3 shows the characteristics of the three different types of features. RMT leads to significantly less features than UNI; this is because RMT is able to leverage the relationship information among the variates to prune redundant features. SVD leads to a smaller number of features than RMT, but the SVD feature vectors are long (length of the time series). As a result, RMT performs better than SVD both in terms of matching time and accuracy. As expected, for this data size (without any parallelizations) RMT takes more time than SVD to extract features; but low-dimensionality and faster matching times would pay off in large data sets.

**Matching Performance and Scalability:** The two tables in Table 7.4 show that the time to match the features from two multi-variate data series is smallest for RMT, followed by SVD. Since features on each uni-variate series have to be considered, UNI generates redundant features and, thus, takes significantly more time to compute the degree of match.

Figure 7.3 compares the total feature size for RMT and SVD as the length of the time series grows: as the figure shows, the total feature size stays more or less constant for RMT, whereas for SVD the feature size grows (since the length of the feature vectors grow with the length of time series).

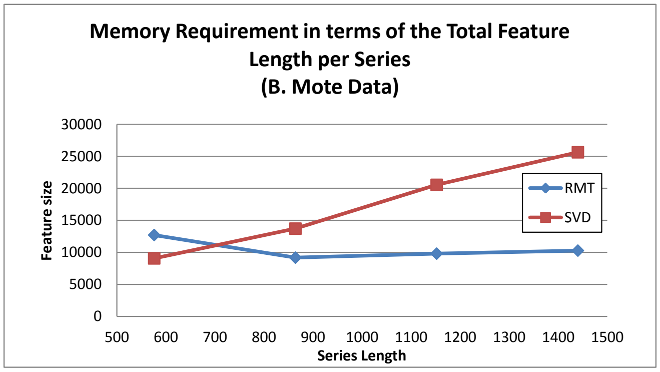


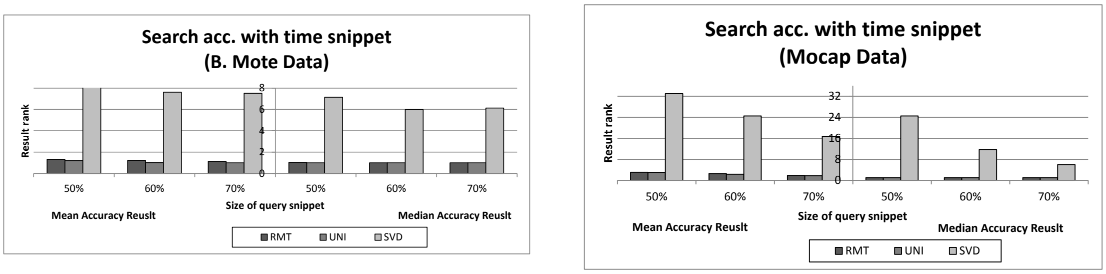
Figure 7.3: Total feature size for RMT and SVD.

*RMT vs. SVD vs. UNI*

As in the classification tasks, (paired) RMT provides the best accuracy, (non-paired) RMT is competitive and cheap, whereas (paired) DTW is much costlier than (paired) UNI. Thus, we compare (non-paired) RMT, (non-paired) SVD, and (paired) UNI features for partial snippet searches when the whole series are not available. Again, we calculate the distance between two time series using RMT features without prune inconsistencies and series are considered as asynchronized.

### Time Snippet Search

Figure 7.4 compares the effectiveness of the three features types for the temporal snippet search scenario described above. As we see here, due to the (temporally) local nature of the queries, both RMT and UNI perform better than SVD-based global feature. In the



(a) Berkeley Mote Data Set

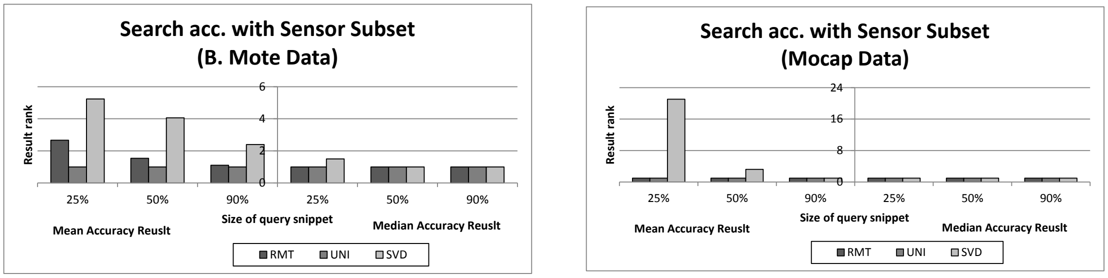
(a) Mocap Data Set

Figure 7.4: Accuracy for time snippet search (the lower the rank, the higher the accuracy).

Mocap data set, RMT also outperforms UNI (moreover, as discussed earlier, RMT costs significantly less to identify matches Figure 7.4). The SVD-based global feature, on the other hand, performs poorly both in mean and median rank accuracy, which indicate that it is ineffective for temporal snippet search.

**Sensor Subset Search**

Figure 7.5 compares the effectiveness of the three feature types for the sensor subset search scenario described above. As we see here, due to its variate-paired nature, as expected UNI performs well (but requires costly matches, Figure 7.4). Among the two non-paired schemes, RMT performs much better than SVD, especially on the Berkeley motes data set and when the search ranges are very small, which indicate that RMT is more robust.



(a) Berkeley Mote Data Set

(a) Mocap Data Set

Figure 7.5: Accuracy for sensor subset search (the lower the rank, the higher the accuracy).

Table 7.5: RMT_noalign vs. RMT_time vs. RMT_var vs. RMT_both.

| | Search acc. with Time Snippet | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | RMT | | | RMT_time | | | RMT_var | | | RMT_both | | |
| | 50% | 60% | 70% | 50% | 60% | 70% | 50% | 60% | 70% | **50%** | **60%** | **70%** |
| Mean | 1.32 | 1.23 | 1.12 | 1.31 | 1.2 | 1.06 | 1.67 | 1.36 | 1.33 | **1.28** | **1.14** | **1.12** |
| Median | 1.03 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | **1** | **1** | **1** |
| Correct rank # | 14 | 19 | 24 | 28 | 27 | 28 | 25 | 28 | 28 | **29** | **29** | **29** |

(a)Accuracy comparison of four strategies for temporal snippet search.

| | Search acc. with Sensor Subset | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | RMT_align | | | RMT_time | | | RMT_var | | | RMT_both | | |
| | 25% | 50% | 90% | 25% | **50%** | **90%** | 25% | 50% | 90% | **25%** | 50% | 90% |
| Mean | 2.67 | 1.53 | 1.1 | 1.8 | **1.06** | **1** | 4.7 | 5.5 | 2.67 | **1.63** | 3.76 | 1.33 |
| Median | 1 | 1 | 1.03 | 1 | **1** | **1** | 1 | 1 | 1 | 1 | 1 | 1 |
| Correct rank # | 17 | 26 | 27 | 26 | **28** | **30** | 16 | 22 | 17 | **26** | 24 | 24 |

(b)Accuracy comparison four strategies for sensor subset search.

### *RMT Alignment Analysis*

In Chapter 4, we proposed two alignment score computation methods, synchronized / asynchronized score and paired / unpaired score; and two inconsistency pruning strategies, temporal inconsistency pruning and variate inconsistency pruning. In this section, we will evaluate the effectiveness of the proposed alignment score computation methods and inconsistency pruning strategies through partial time series search tasks. When searching for alignments, query series in time snippet search task are considered as asynchronized and paired, since we randomly select time intervals from original series; query series in sensor subset search task are considered as synchronized and unpaired, since an unknown subset of sensors are selected.

In this section, we compare RMT without inconsistency pruning (RMT_noalign), RMT with temporal inconsistency pruning (RMT_time), RMT with variate inconsistency pruning (RMT_var) and RMT with temporal & variate inconsistency pruning (RMT_both) through time snippet search task and sensor subset search task.

### Time Snippet Search

As shown on Table 7.5 (a), RMT_both benefits from both RMT_time and RMT_var, thus, it outperforms the other strategies in time snippet search task in terms of mean, median and correct rank amount (correct rank #). As we discussed in Section 6.3, the best result rank for the query snippet should as close to 1 as possible, *correct rank #* counts for queries (out of 30) that return the correct rank. So the higher *correct rank #*, the better accuracy. Here we can claim that RMT_both gives stable and highly accurate results.

**Sensor Subset Search**

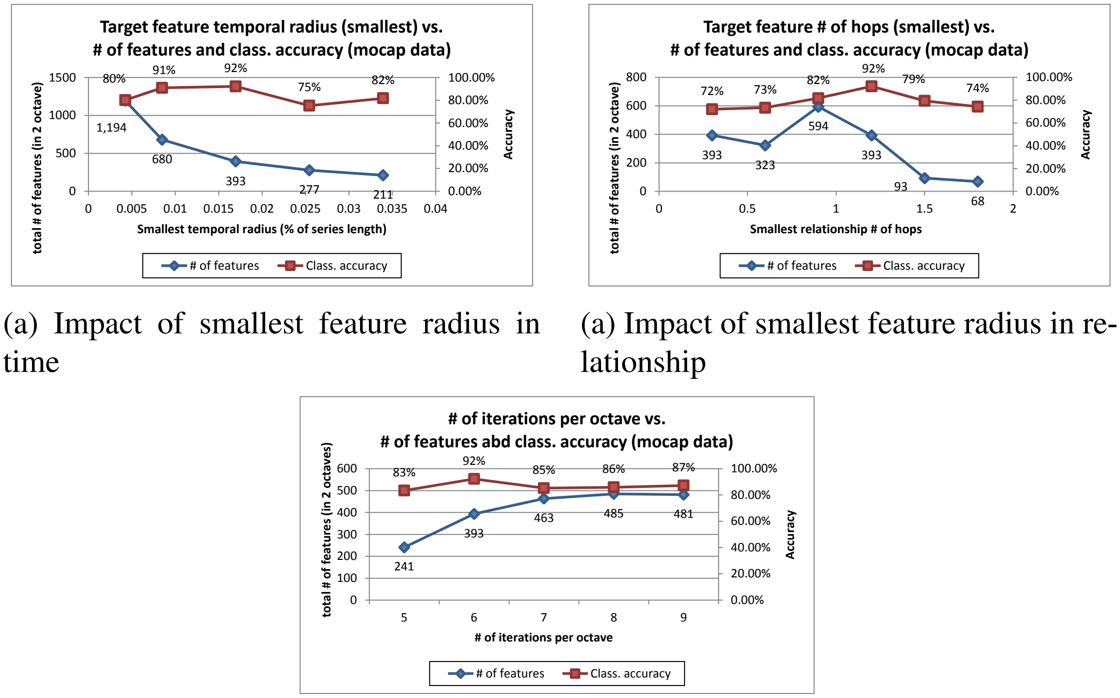
Table 7.5 (b) provides the sensor subset results for the four scenarios, here we preserve sensors in 90%, 50% and 25% of the lab spaces (the rests are dropped). As we expected, for sensor subset search, RMT_time performs the best. RMT_var is not suitable for subset search since when losing sensors, the variate relationships are not comparable any more.

## 7.3 Sensitivity Test

In this section, we examine the robustness of the proposed RMT feature and also explain how the default parameters for the RMT feature are chosen. More specifically, we focus on the impacts of following parameters: number of smoothed series $s$ in each octave ($s_{time} = s_{rel}$); smallest temporal radius, $\sigma_{time}$; and smallest relationship radius, $\sigma_{rel}$, on the Top-5 precision classification results (Mocap data).

Figure 7.8 (a) shows the Top-5 classification precision results by changing the smallest feature radius from 0.5% to 3.5% of the series' temporal length. As we observed, when choosing radius around 1.5% , we get the best classification accuracy results with relatively smaller amount of features. Similarly, as shown in Figure 7.8 (b), the accuracy is also influenced by the feature scope in relationship (here we denote it as # of hops). The best accuracies occur when # of hops equals around 1.2 to 1.3. Moreover, since the speed of the smoothing process is controlled by the # of iterations in each octave, we test the impact of smoothing speed by changing # of iterations from 5 to 9. And when choose 6 iterations, we get the best accuracy result.

Therefore default parameters for the proposed RMT feature extraction algorithms are determined by the presented experiments. Meanwhile, we claim that the accuracies are relatively robust for a large range of parameter values, since in most cases, we get results



(a) Impact of smallest feature radius in time

(a) Impact of smallest feature radius in relationship

(c) Impact of iteration # (number of smoothed series per octave)

Figure 7.6: Top-5 classification precision results. Mocap data set. Red line shows the accuracy and blue line shows the number of RMT features that are extracted. Note that the more features, the higher cost.

in terms of accuracy larger than 80%.

## Chapter 8

## FUTURE DISCUSSIONS AND CONCLUSION

**Parallel and Online Feature Extraction.** The various steps of the RMT candidate feature detection process (smoothing, DoG computation, and extrema detection) can all be parallelized by mapping different portions of the multi-variate data and/or different scales to different processing units. The candidate pruning step can also be trivially parallelized by mapping different subsets of the candidate features to different units. Similarly, in an online setting where the time-series grows continuously with new observations, both the candidate detection and candidate pruning steps can be performed incrementally as new observations arrive.

**RMT Feature Indexing.** RMT descriptors are high-dimensional and therefore feature search and nearest-neighbor based tasks would benefit from locality-sensitive hashing (LSH) based indexing structures [27], which have been shown to perform well when the data is embedded in high dimensional vector spaces.

**Change Detection on Streaming Data.** RMT also provides efficient and effective ways to detect the points in which significant structural changes occur in the data: As we experimentally validate in Chapter 7, the number of features identified in the multi-variate data changes significantly when the dependency/correlation matrix used for RMT feature detection does not reflect the true structure of the data: this is because, when supposedly *nearby* observations are not correlated, this leads to smaller features that can be removed by the process as noise. Therefore, any statistically significant change in the number of features from the historical norms may indicate a shift in the underlying dependency/correlation structure of the data, thereby necessitating re-assessment of the dependency/correlation relationships among the variates. Moreover, frequently occurring or co-occurring features can be used as evidences for strengthening existing variate relationships and/or weakening others.

In this thesis, we presented the robust multi-variate temporal (RMT) Feature extraction algorithm by adjusting the scale invariance process which is used in extracting features for images (SIFT) and uni-variate time series (sDTW). We proposed the inter-related multi-variate time series (IMTS) model and leveraging the variate relationship information to extract features during the whole process: multi-variate time series smoothing, difference of Gaussian (DoG) scale-space constructing, feature candidate detecting, poorly localized feature eliminating and feature descriptor generating. We also built a RMT feature visualization system that enable users to visualize and understand RMT features and IMTS model. In addition, we tested the efficiency and effectiveness of RMT feature and compared it with other alternative feature extraction algorithms. Results show that RMT features are robust against noise and outperform SVD in terms of execution time and accuracy. Also, we demonstrated that RMT algorithm performs almost as good as UNI in most tasks but with less cost. In sum, the proposed RMT feature is highly effective in multi-variate time series searching, alignment and classification tasks.

# REFERENCE

[1] Agrawal, R., Faloutsos, C., & Swami, A. (1993). Efficient similarity search in sequence databases (pp. 69-84). Springer Berlin Heidelberg.

[2] Allen, J. F. (1983). Maintaining knowledge about temporal intervals. Communications of the ACM, 26(11), 832-843.

[3] Akaike, H. (1977). CANONICAL CORRELA TION ANALYSIS OF TIME SERIES AND THE USE OF AN INF ORMA TI ON CRITERION. Computational Methods for Modeling of Nonlinear Systems, 126, 27.

[4] Batal, I., Fradkin, D., Harrison, J., Moerchen, F., & Hauskrecht, M. (2012, August). Mining recent temporal patterns for event detection in multivariate time series data. In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 280-288). ACM.

[5] Battiato, S., Gallo, G., Puglisi, G., & Scellato, S. (2007, September). SIFT features tracking for video stabilization. In Image Analysis and Processing, 2007. ICIAP 2007. 14th International Conference on (pp. 825-830). IEEE.

[6] Berkeley Mote Data Set: http://db.csail.mit.edu/labdata/labdata.html

[7] Berndt, D., & Clifford, J. (1994, July). Using dynamic time warping to find patterns in time series. In KDD workshop (Vol. 10, No. 16, pp. 359-370).

[8] Candan, K. S., Rossini, R., Sapino, M. L., & Wang, X. (2012). sDTW: Computing DTW Distances using Locally Relevant Constraints based on Salient Feature Alignments. PVLDB 5(11).

[9] Candan, K. S., Rossini, R., Sapino, M. L., & Wang, X. (2012, October). STFMap: query-and feature-driven visualization of large time series data sets. InProceedings of the 21st ACM international conference on Information and knowledge management (pp. 2743-2745). ACM.

[10] Chan, K. P., & Fu, A. W. C. (1999, March). Efficient time series matching by wavelets. In Data Engineering, 1999. Proceedings., 15th International Conference on (pp. 126-133). IEEE.

[11] Chen, L., & Ng, R. (2004, August). On the marriage of lp-norms and edit distance. In Proceedings of the Thirtieth international conference on Very large data bases-Volume 30 (pp. 792-803). VLDB Endowment.

[12] Chen, L., Özsu, M. T., & Oria, V. (2005, June). Robust and fast similarity search for moving object trajectories. In Proceedings of the 2005 ACM SIGMOD international conference on Management of data (pp. 491-502). ACM.

[13] Chen, M., Liu, J., & Tang, X. (2008, July). Clustering via Random Walk Hitting Time on Directed Graphs. In AAAI (Vol. 8, pp. 616-621).

[14] CMU Mocap Data: http://mocap.cs.cmu.edu/

[15] De Silva, A., Hyndman, R. J., & Snyder, R. (2010). The vector innovations structural time series framework a simple approach to multivariate forecasting. Statistical Modelling, 10(4), 353-374.

[16] Eichler, M. (2000). Granger-causality graphs for multivariate time series. Preprint, University of Heidelberg.

[17] Eichler, M. (2007). Granger causality and path diagrams for multivariate time series. Journal of Econometrics, 137(2), 334-353.

[18] Faloutsos, C., Ranganathan, M., & Manolopoulos, Y. (1994). Fast subsequence matching in time-series databases (Vol. 23, No. 2, pp. 419-429). ACM.

[19] Fu, A. W. C., Keogh, E., Lau, L. Y., Ratanamahatana, C. A., & Wong, R. C. W. (2008). Scaling and time warping in time series querying. The VLDB JournalThe International Journal on Very Large Data Bases, 17(4), 899-921.

[20] Granger, C. W. (1969). Investigating causal relations by econometric models and cross-spectral methods. Econometrica: Journal of the Econometric Society, 424-438.

[21] Harshman, R. A. (1970). Foundations of the PARAFAC procedure: models and conditions for an" explanatory" multimodal factor analysis.

[22] Harvey, A. C. and Koopman, S. (1997) Multivariate Structural Time Series Model, in *System Dynamics in Economic and Financial Models*, John Wiley and Sons, pp. 269-296.

[23] Harvey, A. C. (1991). Forecasting, structural time series models and the Kalman filter. Cambridge university press.

[24] Harris, C., & Stephens, M. (1988, August). A combined corner and edge detector. In Alvey vision conference (Vol. 15, p. 50).

[25] Hritier, M., Gagnon, L., & Foucher, S. (2009). Places clustering of full-length film key-frames using latent aspect modeling over sift matches. Circuits and Systems for Video Technology, IEEE Transactions on, 19(6), 832-841.

[26] Hopkins, A. S. (2011). Simulating a nationally representative housing sample using EnergyPlus.

[27] Indyk, P., & Motwani, R. (1998, May). Approximate nearest neighbors: towards removing the curse of dimensionality. In Proceedings of the thirtieth annual ACM symposium on Theory of computing (pp. 604-613). ACM.

[28] Itakura, F. (1975). Minimum prediction residual principle applied to speech recognition. Acoustics, Speech and Signal Processing, IEEE Transactions on,23(1), 67-72.

[29] Jaccard, P. (1912). The distribution of the flora in the alpine zone. 1. New phytologist, 11(2), 37-50.

[30] Kadous, M. W. (2002). Temporal classification: Extending the classification paradigm to multivariate time series (Doctoral dissertation, The University of New South Wales).

[31] Keogh, E. J., & Pazzani, M. J. (2000). A simple dimensionality reduction technique for fast similarity search in large time series databases. In Knowledge Discovery and Data Mining. Current Issues and New Applications (pp. 122-133). Springer Berlin Heidelberg.

[32] Keogh, E., Lin, J., & Truppel, W. (2003, November). Clustering of time series subsequences is meaningless: Implications for previous and future research. In-Data Mining, 2003. ICDM 2003. Third IEEE International Conference on (pp. 115-122). IEEE.

[33] Keogh, E., Zhu, Q., Hu, B., Ha,Y.,. Xi,X., Wei, L., and Ratanamahatana, C. The UCR time series classication/clustering homepage. http://www.cs.ucr.edu/eamonn/time series data/ (collected in 2011).

[34] Lang, W., Morse, M., & Patel, J. M. (2010). Dictionary-based compression for long time-series similarity. Knowledge and Data Engineering, IEEE Transactions on, 22(11), 1609-1622.

[35] Li, C., & Prabhakaran, B. (2005, August). A similarity measure for motion stream segmentation and recognition. In Proceedings of the 6th international workshop on Multimedia data mining: mining integrated media and complex data (pp. 89-94). ACM.

[36] Li, L., Prakash, B. A., & Faloutsos, C. (2010). Parsimonious linear fingerprinting for time series. Proceedings of the VLDB Endowment, 3(1-2), 385-396.

[37] Lonardi, J. L. E. K. S., & Patel, P. (2002). Finding motifs in time series. InProc. of the 2nd Workshop on Temporal Data Mining (pp. 53-68).

[38] Lowe, D. G. (1999). Object recognition from local scale-invariant features. InComputer vision, 1999. The proceedings of the seventh IEEE international conference on (Vol. 2, pp. 1150-1157). Ieee.

[39] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints.International journal of computer vision, 60(2), 91-110.

[40] Sakoe, H., & Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. Acoustics, Speech and Signal Processing, IEEE Transactions on, 26(1), 43-49.

[41] Tucker, L. R. (1966). Some mathematical notes on three-mode factor analysis. Psychometrika, 31(3), 279-311.

[42] Vespier, U., Knobbe, A., Nijssen, S., & Vanschoren, J. (2012). MDL-Based analysis of time series at multiple time-scales. In Machine Learning and Knowledge Discovery in Databases (pp. 371-386). Springer Berlin Heidelberg.

[43] Vlachos, M., Kollios, G., & Gunopulos, D. (2002). Discovering similar multidimensional trajectories. In Data Engineering, 2002. Proceedings. 18th International Conference on (pp. 673-684). IEEE.

[44] Wang, C. C., & Wang, K. C. (2008). Hand Posture recognition using Adaboost with SIFT for human robot interaction. In Recent progress in robotics: viable robotic service to human (pp. 317-329). Springer Berlin Heidelberg.

[45] Wang, X., & Candan, K. S. (2010, July). Relevant shape contour snippet extraction with metadata supported hidden Markov models. In Proceedings of the ACM International Conference on Image and Video Retrieval (pp. 430-437). ACM.

[46] Wei, L., & Keogh, E. (2006, August). Semi-supervised time series classification. In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 748-753). ACM.

[47] Ye, L., & Keogh, E. (2009, June). Time series shapelets: a new primitive for data mining. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 947-956). ACM.

[48] Yi, B. K., & Faloutsos, C. (2000). Fast time sequence indexing for arbitrary Lp norms. VLDB.

[49] Yoon, H., Yang, K., & Shahabi, C. (2005). Feature subset selection and feature ranking for multivariate time series. Knowledge and Data Engineering, IEEE Transactions on, 17(9), 1186-1198.