Clustering methods for transaction data that consider market dynamics

van Duijnhoven, E.J.

Award date:
2015

Link to publication

Technische Universiteit
Eindhoven
University of Technology

Clustering Methods for Transaction Data that consider Market Dynamics.

Erwin van Duijnhoven

MSc Thesis

Eindhoven University of Technology
Department of Mathematics

Advisors:
dr. J.M.L. Maubach
dr. Alessandro Di Bucchianico
Jeroen Smeets

Eindhoven, September 28, 2016

Abstract

This MSc thesis examines "transaction"-market dynamics. The situation of interest is a company with anonymous name ACME which sells products to customers (businesses, not end users). Some of these customers purchase more and more over time, whereas others purchase less and less. The aim is to cluster customers which grow (in for instance purchased volume) in one cluster and customers which for instance decrease in a different cluster.

Because customer transactions form mathematically a time-series we focus on clustering techniques for time-series. The clustering itself involves the use of distance measures, which implies the need to use distance measures whose values take into account the dynamics in the customer transaction time-series.

Contents

CHAPTER 1

Introduction

This thesis examines transaction *data* of a large *company* which sells products to its *customers*. These customers are businesses rather than individual people. During the examination we focus on possible mathematical tools/approaches for (1) *clustering* (2) *time-dependent data* and (3) *regularity* of the transaction data:

(1) Without any specific prior knowledge on individual customers, a clustering of customers can be formed based on transaction data. For instance two "*clusters*" of customers can be defined: Small and *large customers* which purchase small respectively large *volume*s of a specific product.

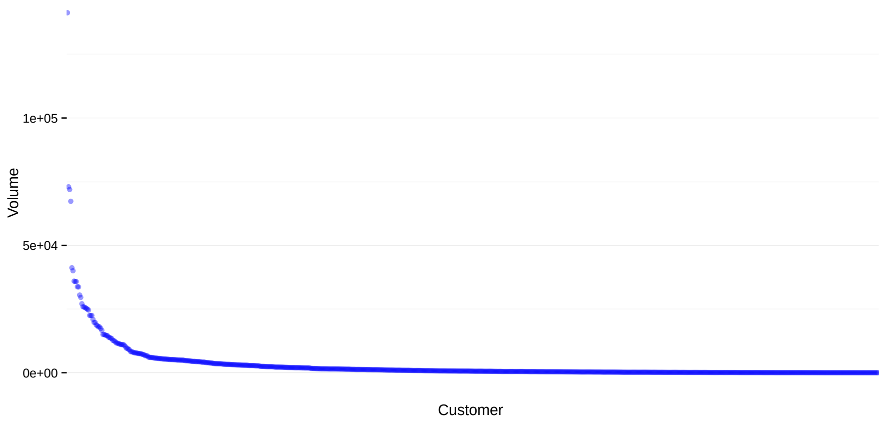


Figure 1.1: (Volume, Customer) Dots ordered on volume.

Typically there are "few" large customers (mostly detached dots in the left of Figure 1.1) which generate large volume sales for the company, and many small customers which each contribute much less to the company's total volume sales. Summarized, *similar customers* could be defined using the transaction data and on that basis clusters can be defined. Chapter 3 focuses on the formation of clusters.

(2) The clustering of the customers (for instance based on large or small volume) does not reflect the dynamics of the market: Some customers typically purchase more and more over the years and grow to be large customers (or the other way around). Thus, it is important to take the time-related aspects such as growth into account.

To demonstrate this a company's transaction volumes of two subsequent years is visualized: We assume that the company sells only one specific product and that each customer $c = 1, \dots, n$ purchases volume v_c^k at times t_c^k, $k = 1, \dots, P_c$. For each customer c Figure 1.2 shows two dots at positions

$$v_c = \sum_{k:t_c^k \text{ in year } i} v_c^k \qquad \text{for } i = 1, 2,$$

where one specific customer is highlighted with the color red.

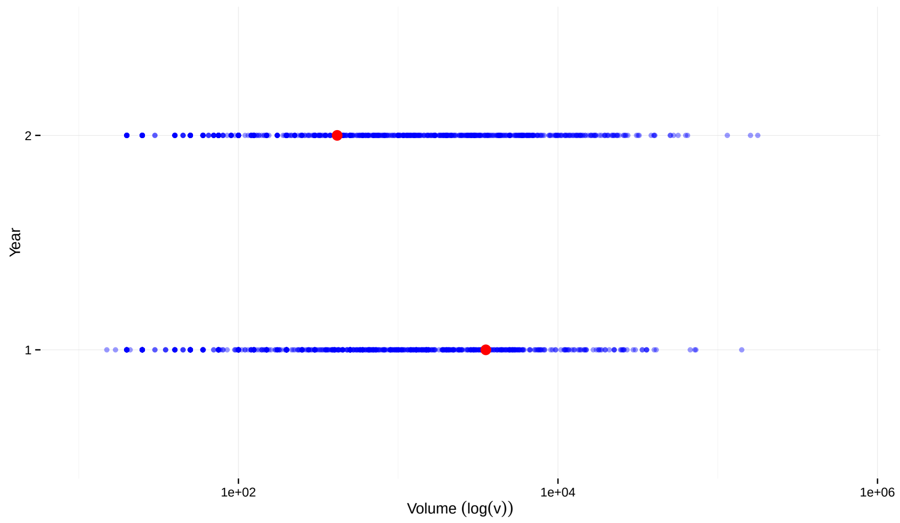


Figure 1.2: Customer data division among total volume on logarithmic scales for 2 years of the ACME transaction data set

Figure 1.2 was generated with data from company ACME (a real company whose name is kept anonymous). The transaction data set has been obtained under the protection of a non-disclosure agreement, which is why we later also work with a self generated data set.

The relative movement of the data over time is shown via the differences between the volume division of the two years Figure 1.2. The red dot in the two figures represent the same customer but in a different year. Note that the relative position of this customer indicated with a red dot with respect to the data cloud is not static.

The red dot is used as an indication of customer movement through time of the data set, where actually most dots move. In Figure 1.3 the movement of a sample of 100 customers is indicated with a line, indicating some data set dynamics. Chapters 4 through 6 focus on dynamic aspects of the transaction data.

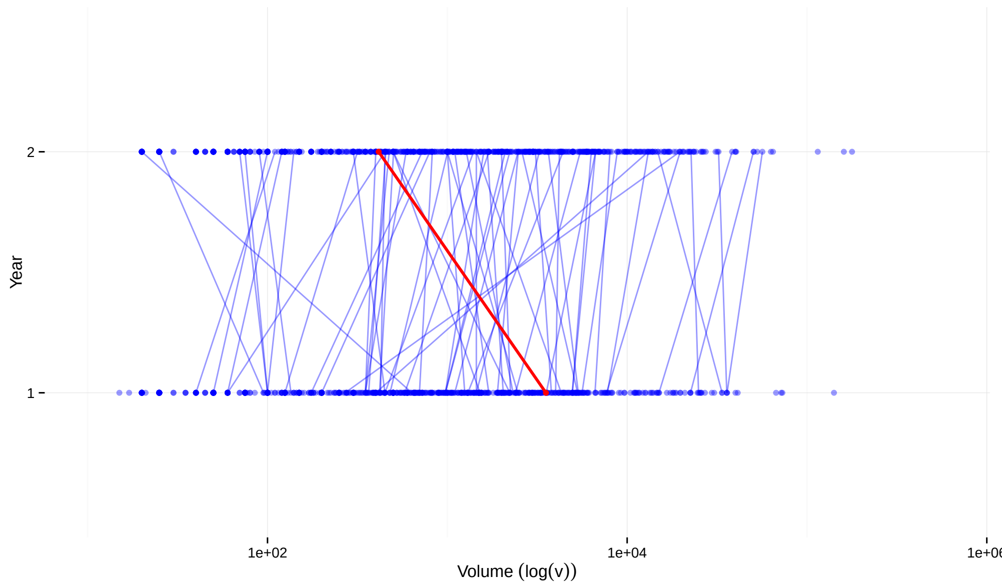


Figure 1.3: Customer movement along total volume for ACME data set over a year.

(3) Developing mathematics for the problems at hand (clustering and preferably taking dynamics into account) relies on the transaction data set properties. Data sets in general contain transactions which are irregular in time but also contain entries which are physically impossible such as selling negative volumes of a product. During this thesis' research many peculiarities were encountered such as:

1. Transactions that contain no volume,
2. Transactions that occur with minimal time separation,
3. Transactions that contain a negative volume,

Most of these peculiar transactions are entered to correct incorrect transactions, at the same time making the transaction data set irregular. The peculiarities 1 and 2 we will process because they are likely. We are not interested in exceptions such as 3 but in the global trends and behaviors. Such peculiarities could be preprocessed, but we did not. Chapters 4 and 6 provide more detail on peculiarities.

1.1 Thesis outline

During a prior internship at a company whose name is kept anonymous due to a NDA, an inventory was made of (some of) the existing clustering methods. In addition a method to cluster based on curves generated by LPC was formulated. Also, recommendations for future research were provided.

For this MSc thesis we, first, created a mathematical model that is used for clustering as well as dynamics analyses. There was no off-the-shelf mathematical model, such that we had to create one ourselves. In Chapter 2 this mathematical model representing the transaction data is given. This chapter and Chapter 4 give methods to mold this model (with minimal effort) to data sets that are used for clustering and dynamics analyses respectively.

Second, we have modeled existing different clustering approaches in accordance with the application. We more elaborately present clustering techniques in Chapter 3.

We, third, examine possibilities to extend these approaches to time series based clustering. Via, one, investigating which distance measure would be beneficial for clustering (of time series) such that dynamics are considered, in Chapter 5. And, two, examining growth indexing functions based on elementary mathematics with the focus on behavior with respect to dynamics, in Chapter 6.

Finally, in Chapter 7 we apply the modeled clustering techniques to data extended with values computed via a growth indexing function.

Where possible we also try to understand whether our example data sets (those of the company) are suited, i.e. approximately satisfy the theoretical conditions for the mathematical methods.

1.2 Literature review

For this Msc thesis approximately 20-30 articles have been examined. This examination started during the prior internship and continued during the course of the first three months of the Msc final project.

The collection of examined articles contain articles that explain the Mean Shift method, which was already used by the internship company. The company was also interested in the use of the R package `LPCM` for which related literature was found.

For this MSc thesis we were interested in clustering that considered transaction data dynamics. We thus expanded this bibliography with articles found searching the internet with the search terms "Time series", "Clustering", "Distances" and "Dynamics". We did however not find any articles that related these search terms with "Transaction", "Market" or "Business" data sets.

We then decided to limit ourselves to the clustering techniques already found by the internship company and focused on articles considering time series distance measures. These articles do provide distance measures but only weakly or not couple these to the "regularity" of the time series involved. Therefore we started with more elementary distance measures and examined these for our transaction data sets. Unfortunately, related literature does not seem to exist.

Next to the literature review we also consulted scientist of the mathematics department of the Eindhoven University of Technology, who are experts in statistics, computational science and analyses. Two CASA multi-hour question meetings were dedicated solely to the topic of this MSc thesis.

CHAPTER 2

Definitions and Data sets

This chapter provides global definitions, notations and functions used in this MSc thesis.

Tuples of data $x = (\cdot, \cdot, \cdot)$ are called *datum* and are typeset using the standard mathroman font (i.e., not bold). The fields of which x consists are called *entries* or *data entries*. The *sets* are denoted by capital letters. For finite sets the n distinct points can be specified,

$$V = \{x_1, x_2, \dots, x_n\}. \tag{2.1}$$

When the order is of any particular need we define the set of points via a sequence or vector

$$V = [x_1, x_2, \dots, x_n]. \tag{2.2}$$

The column sequence is then denoted by

$$V^T = [x_1, x_2, \dots, x_n]^T, \text{ or by } [x_1; x_2; \dots; x_n] \tag{2.3}$$

We use the $|\cdot|$ as the indicator for the number of elements such that

$$|V| = \left|V^T\right| = n \tag{2.4}$$

The standard *norm* $\|\cdot\|$ denotes the Euclidean norm which for points $x = [a_1, a_2, \dots, a_d] \in \mathbb{R}^d$ is defined by

$$\|x\| = \sqrt{\sum_{k=1}^{d} a_k^2}. \tag{2.5}$$

The *maximum norm* $\|\cdot\|_{\max}$ for points $x = [a_1, a_2, \dots, a_d] \in \mathbb{R}^d$ is defined by

$$\|x\|_{\max} = \max_{i=1,\dots,d} a_i. \tag{2.6}$$

The standard *inner product* $\langle \cdot, \cdot \rangle$ denotes the Euclidean inner product which for points $x = [a_1, a_2, \dots, a_d], y = [b_1, b_2, \dots, b_d] \in \mathbb{R}^d$ is defined by

$$\langle x, y \rangle = [a_1, a_2, \dots, a_d] \cdot [b_1, b_2, \dots, b_d]^T = \sum_{k=1}^{d} a_k \cdot b_k. \tag{2.7}$$

2.1 Data set overview

The problem of clustering heavily depends on the considered data set. Therefore this thesis presents several data sets: A corporate (anonymous) data set *ACME*, a self generated large data set called *Contoso* and various special purpose data sets S_1 through S_3.

To perform calculations, this thesis uses the language R (rather than for instance matlab) because R supports access to SQL databases. The ACME and Contoso data set are SQL data sets.

ACME data set

As described in the introduction this is a SQL data set from a real company and (therefore) contains a lot of irregularities. Each transaction contain many more entries (than needed for our clustering analyses). This data set is used to explain the types of problems that we encountered during this project. The ACME data set is anonymized due non-disclosure issues: Customer names, transactions nor units (such as currency) are given.

Contoso data set

The Contoso is manufactured to compare different methods and applications. It is created such that it is regular. This data set contains transactions which contain precisely enough entries for our clustering analysis. The clustering methods introduced in this thesis all work on this data set and result in predictable solutions. The data set is chosen such that the different clustering methods show different results.

The Contoso data set is generated via random samples from the normal and exponential distribution with a fixed seed. We chose a normal distribution because a lot of the clustering methods are theoretically based on this distribution. The exponential distribution is chosen to generate times between transactions, since it is commonly used to generate arrival times.

The manufactured data sets S_1, S_2 and S_3

The data sets S_1, S_2 and S_3 in Chapter 6 are created for specific purposes. These are data sets used to mathematically support an argument. They are situation specific to indicate why something works or fails to work. These data sets contain only variables that are needed in these specific situation. The specifics of these data sets are therefore given in Chapter 6.

2.2 Data set and transaction definition

To find similar customers, the customers have to be compared and analyzed. Since there is no obvious way of comparing customers, data that describes customer behavior is analyzed. These data are a set of points (i.e. *transactions*) x each consisting of multiple data entries describing a variable or value. The data entries can be divided into three *data entry categories*; Customer properties $\hat{C}$, the moment at which the info was obtained $\hat{T}$, Quality data $\hat{D}$, i.e. $x \in \hat{C} \times \hat{T} \times \hat{D}$. Note that each data entry category may consist of multiple data entries.

Customer properties are data entries that describe characteristics or features of a customer. For instance region, market, customer and or product are data entries that are covered by this category. These data entries are mostly indexed and particularly used for grouping data before it is analyzed.

Time of measurement is the time when a transaction x entered the database. Quality data is directly linked to the time of measurement since they are recordings at that moment in time.

Quality data is the category that mostly consists of data entries indicated by numerical values. These values describe for instance contract or order info such as volume. The entries in this category vary with time and are the subjects to be clustered.

Let V denote the set of all transactions. Using the data entry categories a transaction $x \in V$ can be described by a triplet $x = (c, t, d)$, such that V is defined by

$$\begin{aligned} V = \big\{ &(c_1, t_1^1, d_1^1), (c_1, t_1^2, d_1^2), \dots \\ &, (c_2, t_2^1, d_2^1), (c_2, t_2^2, d_2^2), \dots \\ &, \dots \\ &, (c_n, t_n^1, d_n^1), (c_n, t_n^2, d_n^2), \dots \big\} , \end{aligned} \tag{2.8}$$

where equal sub-index i for t_i^k, d_i^k and c_i indicates that these time and quality data are related to the customer properties c_i.

The data set V is therefore a combination of these three data categories, such that

$$V \subset X = C \times T \times D \subset \hat{C} \times \hat{T} \times \hat{D} = \Omega,$$

where $C \subset \hat{C}$, $T \subset \hat{T}$, $D \subset \hat{D}$ and

$$C = \{c_i : x_i = (c_i, t_i, d_i) \in V\}$$

and T, D similarly defined. We need the finite sets C, T and D because SQL queries yield these finite sets and we need $\hat{C}, \hat{T}$ and $\hat{D}$ to (in principle) check mathematical conditions assumed by the clustering techniques.

Example 1

$$\begin{aligned} V = \{&(\text{"}Anon\text{"}, 1427846400, 300.23)\,, \\ &(\text{"}Bocra\text{"}, 1352345100, 623.41)\,, \\ &(\text{"}Creat\text{"}, 1342398100, 239.12)\} \end{aligned}$$

whence

$$\begin{aligned} C &= \{\text{"}Anon\text{"}, \text{"}Bocra\text{"}, \text{"}Creat\text{"}\} && \subset \hat{C} \\ T &= \{1427846400, 1352345100, 1342398100\} && \subset \hat{T} && = \mathbb{N} \\ D &= \{300.23, 623.41, 239.12\} && \subset \hat{D} && = \mathbb{R} \end{aligned}$$

with $\hat{C}$ the set of all strings of finite length.

Definition 2.2.1 (Category Projection)
Define *category projections* as projections that project points onto a category. Let $x \in V$ be a point, the category projection onto category $\hat{C}$ is defined by

$$\begin{aligned} P_C(x) &= P_C(c, t, d) \\ &= c. \end{aligned}$$

Realize that $C = \{P_C(x) : x \in V\}$. P_D and P_T are similarly defined to P_C as the projection onto the data category $\hat{D}$ and $\hat{T}$ respectively.

Using the projections one can group, organize and mold the data set such that it can be used for several clustering applications.

With the operation $|\cdot|$ (2.4) indicating the number of points in a set, note that $|C|, |T|, |D| \leq |V|$. Since the amount of points (transactions) $x \in V$ is finite, it also follows that $|V| < \infty$ and $|C|, |T|, |D| < \infty$.

Example 2 (Contoso)
Contoso, an example company, has a customer file consisting of 1000 customers. The business is described by points (i.e. transactions) having the data entries given in Table 2.1. Next to volume it is desirable to have a second entry, not correlated to volume, to be able to demonstrate clustering for multidimensional data in Chapter 3. The second entry was taken to be *delta*, which is assumed to be uncorrelated to volume.

Data Entry	**Data Category**	**Description**
Customer	C	The customer.
Product	C	The product bought.
Timestamp	T	The time.
Volume	D	The volume delivered.
Delta	D	The difference of volume ordered and volume delivered.

Table 2.1: Contoso data entries.

The Timestamp for Contoso data is given as the Unix time, also known as the Epoch time. This is a method widely used in software applications that indicates the number of seconds that have elapsed since 00:00:00 Coordinated Universal Time (UTC), Thursday, 1 January 1970, not counting leap seconds. As an example, a customer i ordered product A for a volume of 450 and eventually bought a volume of 470 on April 1st 2015. This transaction $x \in V$ is given by the values in Table 2.2.

Data Entry	**Value**
Customer	i
Product	A
Timestamp	1427846400
Volume	470
Delta	20

Table 2.2: Contoso transaction $x \in V$, entry value example.

Contoso, as an example, only sells one product: Product A. Therefore there is no need to consider the data entry product in the customer properties category. Hence, customer is the only data entry left in this category, such that, looking at the data as in (2.8), $c_i \in C$ indicates a unique customer of Contoso.

From now on we assume that $c_i \in C$ indicates a single entry, even if c_i contains multiple data entries. Each customer can still have a region and market as a data entry, but a unique customer is indicated by $c_i \in C$. The number of unique customers n is equal to $|C|$.

The quality data $d_i \in D$ related to $c_i \in C$ is the subject of clustering (Which automatically induces a clustering of the customers $c_i \in C$). Essentially, *clustering* is finding subsets $\tilde{C} \subset C$ such that the data related to $c_i, c_j \in \tilde{C}$ is similar and related to $c_i \in \tilde{C}, c_j \in C \setminus \tilde{C}$ is by definition dissimilar. The transactions of customer c_i are given by

$$P_C^{-1}(c_i) = \{x \in V : P_C(x) = c_i\} \subset V. \tag{2.9}$$

The set $P_C^{-1}(c_i)$ is the result of a filter operation on V (filtering out all data related to c_i). A clustering over the subsets $P_c^{-1}(c_i)$ will therefore result in groups of similar customers with respect to transaction data.

Example 3 (ACME)
The ACME data set is not as clean as the data set described for Contoso. In general real

data sets contain a lot of data entries, whether or not data entries are indeed of importance for customer similarities is a whole different study and will not be addressed in this thesis. We instead assume we only have independent (to some extend) data entries. The relevant quality data entries for ACME are similar to the ones defined in Table 2.2. The exception is the delta variable, which is not a variable in this data set.

For the data category C we also assume that other than customer, no other customer properties are of influence on customer similarities. All transaction data seen is referring to the same product, such that c_i indicates a single customer product combination. We hereby skip a few regularization steps, but these too are a separate study which we do not address to in this thesis.

2.3 Static cdata

Most existing clustering methods base their clusters on the distances between points from a data set. For a lot of these methods the distances are computed using standard distance measures in the space $\mathbb{R}^d$ for $d \geq 1 \in \mathbb{N}$. For our transaction data sets to fit these clustering methods it has to be of the form

$$V = \{(c_1, x_1), (c_2, x_2), \dots, (c_n, x_n)\}\,, \tag{2.10}$$

i.e. a single datum $x_i \in \mathbb{R}^d$ per customer c_i.

We however have a different situation: For every c_i we have a *time series*. A time series, denoted (t, d), is a finite series of transactions ordered on time

$$(t, d) = (t^1, d^1), \dots, (t^P, d^P)$$

of finite implicit length $P \in \mathbb{N}$. Sometimes time series are indexed with subscript "i" if it is related to customer c_i

$$(t, d)_i \overset{\text{notation}}{:=} (t_i^1, d_i^1), \dots, (t_i^{P_i}, d_i^{P_i}) \tag{2.11}$$

for some $P_i \in \mathbb{N}$. This time series $(t, d)_i$ is created from the transactions in $P_C^{-1}(c_i)$. Alternatively, the time series $(t, d)_i$ related to customer c_i could be written as

$$(t, d)_i = (c_i, t_i^1, d_i^1), (c_i, t_i^2, d_i^2), \dots, (c_i, t_i^{P_i}, d_i^{P_i}).$$

More on time series and their characteristics upward of Chapter 4.

Since we have a time series $(t, d)_i$ consisting of multiple data for each customer c_i our original data set is not of the form (2.10). Therefore, before we can use any of these existing clustering methods to find clusters of customers, we need to construct *cdata* where each customer is represented by a single *cdatum* $x_i \in \mathbb{R}^d$ as in (2.10). This cdatum x_i should represent characteristics of the customer time series $(t, d)_i$. The entries of the cdata are called *cdata entries*.

Since V is a set of points measured in time we have to lump data to create the cdata. Define the *cdata function* $g : X \mapsto \mathbb{R}^d$ as the function that maps time series $(t, d)_i$ onto cdata $x_i \in \mathbb{R}^d$ such that

$$(c_i, x_i) = (c_i, g\left((t, d)_i\right))\,, \qquad \text{for } i = 1, \dots, n,$$

which is conform with the desired data set form (2.10).

The cdata function $g : X \mapsto \mathbb{R}^d$ accumulates for each customer all its transactions. This function uses different operators such as sum, minimum, maximum and average on the data entries and aims to represent their characteristics in the cdata. Define $\hat{V} = \{x_1, x_2, \dots, x_n\} \subset \hat{X}$ as the set of cdata points x_i each related to the unique $c_i \in C$.

Example 4 (Contoso)
For Contoso we take the cdata entries as given in Table 2.3. In the cdatum x_i for customer c_i we want two cdata entries that represent the original data entries volume and delta. We

create these cdata entries as the sum of the individual volume and delta values inside a fixed period.

The cdata is calculated over transactions from within a fixed time period, for instance a year. Let $dat_1 < dat_2$ be the limiters to this fixed time period, the cdata function g for Contoso is defined by

$$\begin{aligned} x_i &= g((t,d)_i) \\ &= g\left((t_i^1, d_i^1), (t_i^2, d_i^2), \ldots, (t_i^{P_i}, d_i^{P_i})\right) \\ &= g\left(\left(t_i^1, \text{Volume}_i^1, \text{Delta}_i^1\right), \left(t_i^2, \text{Volume}_i^2, \text{Delta}_i^2\right), \ldots\right) \\ &= \begin{pmatrix} g_1((t,d)_i) \\ g_2((t,d)_i) \end{pmatrix} \qquad (2.12) \\ &= \begin{pmatrix} \text{Total volume}_i \\ \text{Total delta}_i \end{pmatrix}, \qquad (2.13) \end{aligned}$$

where the *cdata entry functions* g_1 and g_2 are defined by

$$g_1((t,d)_i) = \sum_{\left\{k : dat_i \le t_i^k \le dat_2\right\}} \text{Volume}_i^k \qquad (2.14)$$

$$g_2((t,d)_i) = \sum_{\left\{k : dat_i \le t_i^k \le dat_2\right\}} \text{Delta}_i^k. \qquad (2.15)$$

Note that the cdata entries total volume and total delta are a representation of, and therefore different from, the original data entries volume and delta in $(t,d)_i$. Also note that the time elements t_i^k are not playing part in these operators, other than the selection of a time frame. In Figure 2.1 the Contoso cdata set (g_1, g_2) is plotted, indicating the division of customers among this cdata.

cdata entry	**Original data Entry**	**Description**
Total volume	Volume	The total volume delivered.
Total delta	Delta	The difference of the total volume ordered and the total volume delivered.

Table 2.3: Contoso cdata entries.

Example 5 (ACME)
The ACME cdata is created in the same fashion as the Contoso cdata, see (2.12). Prior to the calculation of the cdata the regularity issues given in Example 3 are solved. Next to these regularity issues the data set contains a lot more regularity issues with respect to time which are later described in Chapter 4. The cdata entry functions in (2.12) are discretizations of integral functions (which are regularizing functions) such that no further regulation steps are necessary to manage these time irregularities for the ACME cdata set.

The ACME data set does not contain the information to extract the delta values. Therefore the ACME cdata is (in this case) one dimensional: Only consisting of the cdata entry total volume g_1. Since the clustering methods in the next chapter can be used on multidimensional cdata we do not apply them to the ACME cdata, simply because results will not be "exciting" enough.

The division of total volume g_1 is visualized in Figure 1.1 and 1.2. In these figures we see that the ACME cdata is not as evenly distributed in comparison to the manufactured Contoso cdata plotted in Figure 2.1, especially when considering that the ACME cdata in Figure 1.2 is plotted on logarithmic axis.

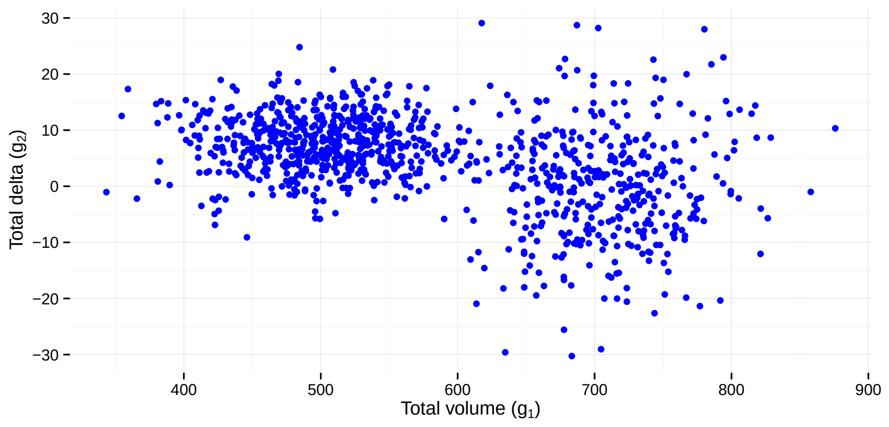


Figure 2.1: Total volume and total delta scatter diagram of Contoso customers.

The start of clustering without the consideration of the time elements t_i^k, as in example 4, is on purpose. The consideration of time elements is paired with a lot of unanswered questions (see Chapter 4) and the effect on the application is not clear. Therefore this problem is addressed to upward of Chapter 4.

The cdata entries that do not consider the time elements t_i^k other than for the selection of the period (like for instance g_1 and g_2 ((2.14) and (2.15))) form a category of cdata entries, from now on referred to as *static cdata entries.* If the cdata contains only static cdata entries it is called *static cdata.*

Note that the static cdata entry functions accumulate the time series $(t, d)_i$ to one datum x_i, such that a lot of data is lost. The choice in cdata entry functions used is therefore of influence on the subsequent analyses. For instance, the amount of transactions and their individual volume are not taken into account by static cdata (2.12). This choice of cdata might lead to considering customers with totally different characteristics as similar. However incomplete, the cdata (2.12) still indicates global and important characteristics of customers. Also, this cdata can be implemented into a lot of existing clustering methods and is therefore easier to use for analyses.

Note that the points plotted in Figure 2.1 need not be points on a graph of a single function, such that no functional relationship between g_1 and g_2 exists. Therefore clustering could be a useful tool, where for Contoso it becomes "as easy as" finding dense clouds inside Figure 2.1. Here the outlines of two clusters can easily be formed via logical reasoning. In practice (for instance on the ACME data set) the clusters are not obvious and a mathematical approach is needed, especially when multiple dimensions are considered.

CHAPTER 3

Clustering Methods

This chapter presents two different clustering methods for cdata in $\mathbb{R}^d$. The related algorithms and their qualities are discussed. These clustering methods are based on kernel density estimation which is explained first.

3.1 Kernel density estimation

The *density function* $f : \mathbb{R}^d \mapsto \mathbb{R}$ is a function that describes the relative likelihood for a random variable to take on a given value. Since for our sample data $\hat{V}$ there is no (known) true density function it is approximated with a technique called density estimation. A highly popular method of density estimation is kernel density estimation which uses kernel functions.

Definition 3.1.1 (Kernel function)
A *kernel function* is a non-negative real-valued integrable function $K : \mathbb{R}^d \mapsto \mathbb{R}$ satisfying the following three requirements:

$$\begin{aligned} K(u) &\geq 0 && \forall u \in \mathbb{R}^d \qquad (3.1) \\ \int_{\mathbb{R}} K(u)du &= 1, && \\ K(-u) &= K(u) && \forall u \in \mathbb{R}^d. \end{aligned}$$

Note that for $h > 0$

$$u \mapsto \frac{1}{h^d} K\left(\frac{u}{h}\right)$$

is also a kernel function.

Example 6 (Gaussian kernel (normal kernel))
The *Gaussian kernel* is defined by the function

$$K(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2}$$

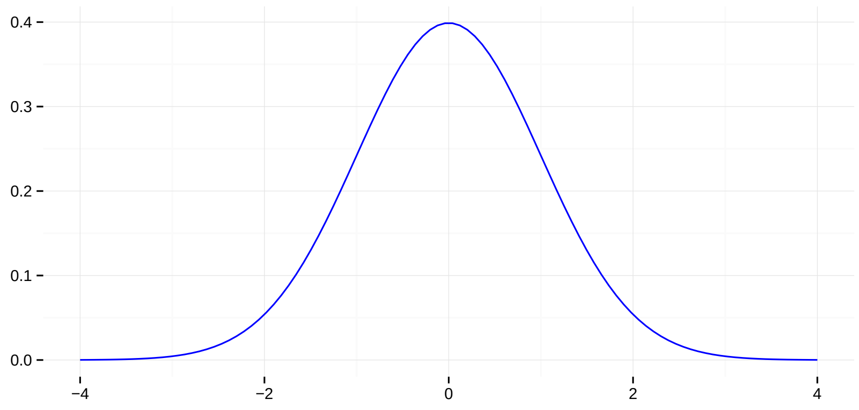


Figure 3.1: The Gaussian kernel.

Let $\mathbf{x}_1, \ldots, \mathbf{x}_n$ be a sample of d-variate random vectors drawn from a common distribution described by the density function f. The *kernel density estimate* is defined by

$$\hat{f}_h(x) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{x - \mathbf{x}_i}{h}\right). \tag{3.2}$$

The first requirement for a kernel (3.1) ensures that the kernel density estimate (3.2) results in a probability density function. The expected value of the kernel density estimate is equal to the sample mean.

We use the Gaussian kernel as a basis for our density estimation because the mathematical theories of the presented clustering methods are all based on the normal distribution. It is also often used to describe, at least approximately, any set of (possibly) correlated real-valued random variables each of which clusters around a mean value, which happens to be the field of interest for clustering methods.

Our cdata set $x_1, \ldots, x_n$ is not independently drawn from a density function f but is a fixed set of values. By replacing the stochastic sample $\mathbf{x}_1, \ldots, \mathbf{x}_n$ with this cdata set $x_1, \ldots, x_n$, the kernel density estimate (3.2) defines a fixed function estimating the cdata's density.

3.1.1 Bandwidth optimization

The value $h > 0$ in (3.2) is a smoothing parameter called the *bandwidth*. The bandwidth is used to scale the kernel functions appropriate to the data. The optimal value of h is depending on the kernel function and the data set variance, intuitively one wants to choose h as small as the data allows.

The kernel density estimate is sensitive to the choice of bandwidth h. When the bandwidth is too small the resulting density estimation will contain spurious data artifacts. On the other hand, when the bandwidth is too large, the resulting density estimation will be over smoothed and will obscure too much of the underlying structure. A bandwidth h is said to be good when the resulting kernel density estimate $\hat{f}_h$ is close to the true density function f from which the sample $\mathbf{x}_1, \ldots, \mathbf{x}_n$ is drawn.

When the true density f is known an optimality criterion for the bandwidth selection is used in the form of the expected squared L_2 risk function, also termed the *mean integrated squared error*

$$\mathrm{MISE}(h) = \mathbf{E}\left[\int \left(\hat{f}_h(x) - f(x)\right)^2 dx\right].$$

This expectation indicates a measure of distance between the true density f and the density estimate $\hat{f}_h$ depending on h. The bandwidth h which minimizes the function MISE is considered to be the *optimal bandwidth*.

Because we do not know f, we can not calculate an h which minimizes MISE(h). However, with a sample $\mathbf{x}_1, \dots, \mathbf{x}_n$ drawn from f the standard normal density function with standard deviation σ and $\hat{f}_h$ based on the Gaussian kernel, the optimal choice for bandwidth h, Silverman (1986), becomes

$$h = \sqrt[5]{\left(\frac{4\sigma^5}{3n}\right)} \approx 1.06\sigma n^{-1/5}. \tag{3.3}$$

This approximation is termed *the normal distribution approximation*, *Gaussian approximation* or *Silverman's rule of thumb*.

Unfortunately our cdata $x_1, \dots, x_n$ is neither standard normally distributed nor independently drawn from a density function f. Nevertheless we use the bandwidth h given by (3.3) in the remainder of this thesis.

3.2 Mean Shift Method

The Mean Shift method, as described in Comaniciu and Meer 2002, is a clustering method based on cdata sets $\hat{V} \subset X = \mathbb{R}^d$. This method uses the kernel density estimate directly. In the kernel density estimate Mean Shift uses a *radially symmetric kernel* K, which have the specific property that the set of points for which $K(x) \neq 0$ is assured to be a d dimensional sphere. Given a *profile* $k : \mathbb{R} \mapsto \mathbb{R}$ and a *normalization constant* $c_k > 0$ such a kernel can be written as

$$K(x) = c_k \cdot k\left(\|x\|^2\right),$$

where $\|\cdot\|$ is the Euclidean norm (2.5).

The normalization constant ensures that K integrates to one over $\mathbb{R}$, as all kernels do. Hence, the kernel density estimate (3.2) can be rewritten for radially symmetric kernels to

$$\hat{f}_h(x) = \frac{c_k}{nh^d} \sum_{i=1}^{n} k\left(\left\|\frac{x - x_i}{h}\right\|^2\right),$$

with h the bandwidth.

We are interested in the modes of the density f which correspond to the maxima of this density and thus are zeros of the gradient of this density. The modes m of the density function f satisfy the equation

$$\nabla f(\mathfrak{m}) = 0.$$

In practice the density f is unknown and its modes are estimated using the kernel density estimate. The Mean Shift method uses the kernel density estimate $\hat{f}_h$ computed from the cdata set $\hat{V}$ to link each point $x_i \in \hat{V}$ to a specific mode m_i for $i = 1, \dots, n$. We set $M = [m_1, \dots, m_n]$ as the sequence of modes specifically related to $x_1, \dots, x_n$. The points connected to the same mode form a cluster, such that the cluster around x_i is the set of points

$$\left\{x_j \in \hat{V} : m_j = m_i\right\}. \tag{3.4}$$

One possible way to link modes to the points $x_1, x_2, \dots, x_n$ is computing the density estimation and analyzing it. When the computed density estimation is plotted it looks like a mountain scenery. In this mountain scenery the modes are the peaks of the mountains and clusters are points on the same mountain. This idea is sketched in Figure 3.2, Comaniciu and Meer 2002.

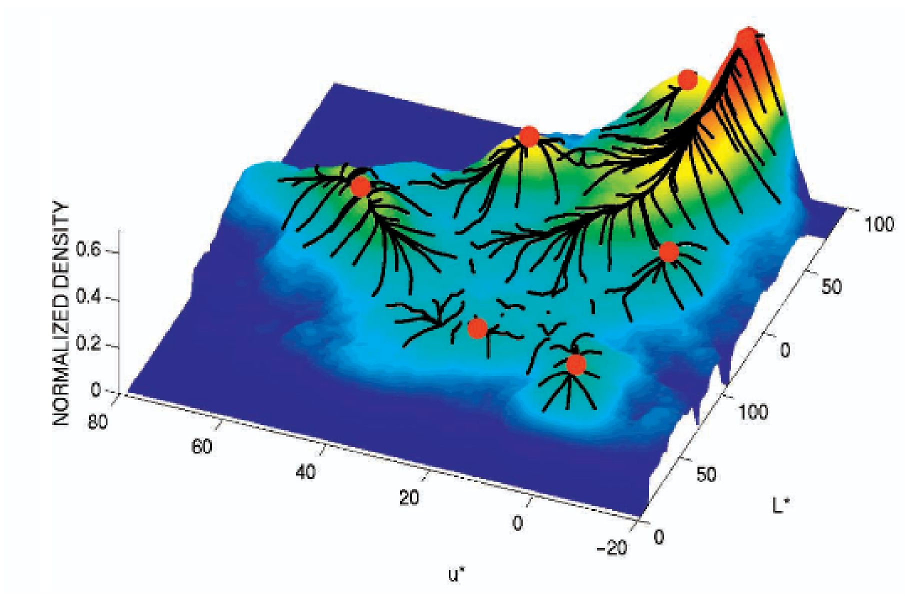


Figure 3.2: Mountains of the kernel density estimate plotted with their modes highlighted.

Example 7
For Contoso cdata the mountain scenery is plotted in Figure 3.3. The kernel density estimate has two mountain tops, which indicate the modes. The mountains themselves indicate two cluster positions.

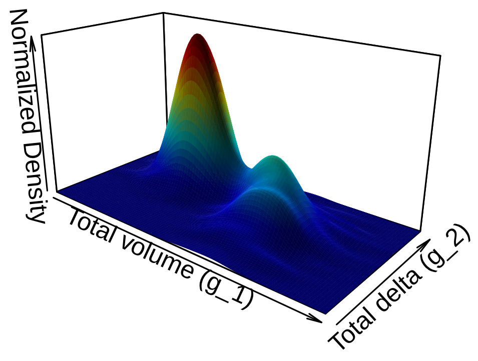


Figure 3.3: Total volume and total delta kernel density estimate plotted as a mountain scenery for Contoso customers.

The calculation and analysis of the kernel density estimate however takes a lot of time, especially for multi-dimensional cdata. The Mean Shift method is therefore created to link each point x_i to a mode m_i for $i = 1, \dots, n$ without computing the full kernel density estimate of the cdata set.

The Mean Shift method is based on the *gradient of the density estimate*, defined by

$$\nabla \hat{f}_h(x) = \frac{2c_k}{nh^{d+2}} \sum_{i=1}^{n} (x - x_i)\, k' \left(\left\| \frac{x - x_i}{h} \right\|^2 \right),$$

where k' is the derivative of the kernel profile k. Define the function

$$p(x) = -k'(x).$$

Assume that this derivative of the kernel profile k exists for all $x \in [0, \infty)$, except for a finite set of points. With a new normalization constant c_p, use the function p as a profile for a new kernel function

$$P(x) = c_p p \left(\|x\|^2 \right).$$

Note that when K is a Gaussian kernel, the kernel P has the same expression. (Comaniciu and Meer (2002))

With the use of the function p the gradient of the density function can be rewritten.

$$\begin{aligned} \nabla \hat{f}_h(x) &= \frac{2c_k}{nh^{d+2}} \sum_{i=1}^{n} (x_i - x)\, p \left(\left\| \frac{x - x_i}{h} \right\|^2 \right), \\ &= \frac{2c_k}{nh^{d+2}} \left[\sum_{i=1}^{n} p \left(\left\| \frac{x - x_i}{h} \right\|^2 \right) \right] \left[\frac{\sum_{i=1}^{n} x_i p \left(\left\| \frac{x - x_i}{h} \right\|^2 \right)}{\sum_{i=1}^{n} p \left(\left\| \frac{x - x_i}{h} \right\|^2 \right)} - x \right]. \end{aligned} \tag{3.5}$$

The first term of Equation (3.5) is proportional to the density estimate at x computed with kernel P. The second term is called the *Mean Shift term*, denoted by

$$m_h(x) = \frac{\sum_{i=1}^{n} x_i p \left(\left\| \frac{x - x_i}{h} \right\|^2 \right)}{\sum_{i=1}^{n} p \left(\left\| \frac{x - x_i}{h} \right\|^2 \right)} - x.$$

The Mean Shift term is the vector that maps x onto its *local weighted mean*. This vector therefore always points in the direction of maximum increase in density. The intuitive interpretation of the Mean Shift term $m_h(x)$ is that it shifts x toward the region in which the majority of the points reside.

The Mean Shift algorithm exploits this property of the Mean Shift vector and computes successive locations of the local weighted mean, defined by

$$\mu_h(x) = \frac{\sum_{i=1}^{n} x_i p \left(\left\| \frac{x - x_i}{h} \right\|^2 \right)}{\sum_{i=1}^{n} p \left(\left\| \frac{x - x_i}{h} \right\|^2 \right)}.$$

The sequence y consisting of successive locations of the local weighted mean starting at point x is defined by

$$\begin{aligned} y^{(0)} &= x, \\ y^{(i+1)} &= \mu_h(y^{(i)}). \end{aligned} \tag{3.6}$$

The Mean Shift term has smooth trajectory properties which state that these iterative steps do not fluctuate between multiple points. Comaniciu and Meer 2002 states that the sequence y converges due to the smoothness together with the convexity of K and the monotonically decreasing profile k. The Mean Shifts iterative method therefore stops when the sequence y is converged. This convergence point m is then a mode of the kernel density estimate.

Algorithm 1: Mean Shift algorithm process

```
Function $[\mathfrak{m}_1, \mathfrak{m}_2, \ldots \mathfrak{m}_n]$ =MS $([x_1, x_2, \ldots, x_n])$
for $i := 1$ to $n$ step $1$ do
    $k := 0$;
    $y_i^{(k)} := x_i$;
    while $\left\| y_i^{(k)} - \mu_h\left(y_i^{(k)}\right) \right\| > \varepsilon$ do
        $k = k + 1$;
        $y_i^{(k)} = \mu_h\left(y_i^{(k-1)}\right)$;
    end
    $\mathfrak{m}_i = y_i^{(k)}$;
end
```

Example 8 (Contoso)
The Mean Shift algorithm, Algorithm 1, applied to the Contoso cdata divides it into two clusters. In Figure 3.4 these clusters are indicated by color and its modes are indicated by a red dot. The division of cdata points and the modes are conform with the mountain scenery plotted in Figure 3.3. Note that with only two cluster the cdata division is coarse.

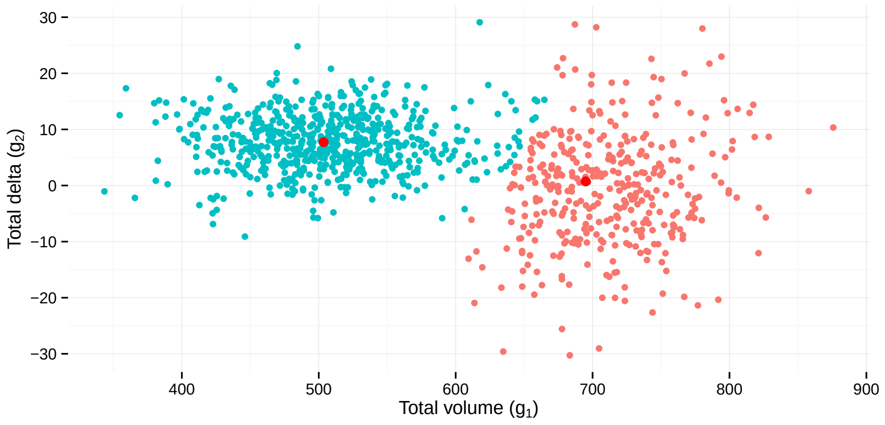


Figure 3.4: Clusters on Contoso cdata computed with the Mean Shift method (Each cluster differently colored, centers indicated).

In general, the clusters computed via the Mean Shift method, tend to be coarse on dense data sets. Because most cdata sets analyzed are dense, as for instance the ACME cdata set, the Mean Shift method will result in little and large clusters.

3.3 Local Principal Curve Methods

In this section we examine a data representation method called *local principal curves* (*LPC*) Einbeck et al. 2005. LPC is a method that represents data sets by a curve. The curve should be such that it describes main characteristics of the data set.

In Einbeck et al. (2005) the history and motivation of finding curves describing points $x_1, x_2, \ldots, x_n$ is discussed. These curves can for instance be used for dimension reduction or simply as a description or representation of the data set.

The LPC method is based on principal component analysis. Via *principal component analysis* a set of observations of possibly correlated variables are converted into a set of values of linearly uncorrelated variables through a statistical procedure called orthogonal transformation.

The principal components analyses can be extended to *principal curves*. Principal curves are defined as one-dimensional smooth curves passing through the "middle" of a d-dimensional data set. The notion of a principal curve is intuitively clear, but there is a lot flexibility in how to define the "middle" of a data set. The first applications of principal curves uses the concept of *self-consistency*, meaning that each point of the principal curve is the average over all points that project there.

The currently existing principal curve algorithms can be divided into two families. First the algorithms with a *"top-down"- strategy*. These algorithms use a line or curve as a starting point and by modifying it the algorithms try to dwell out this line till the curve satisfactorily passes through the middle of the data. Unfortunately the dependence on the initial line leads to technical problems and a lack of flexibility.

The second family of algorithms are the *"bottom-up" algorithms*. These consider in every step only data in a local neighborhood of the currently considered point. The first principal curve approach which uses the "bottom-up" strategy was proposed in Delicado (2001). Based on this strategy, Einbeck et al. (2005) proposes a relatively quick method that creates local principal curves based on local information.

3.3.1 Principal curve of oriented points

Local principal curves are principal curves that are created using the "bottom-up" strategy. The curves are defined in the d-dimensional space $\mathbb{R}^d$. The method described in Delicado (2001) therefore is defined for cdata $\hat{V} \subset \hat{X} = \mathbb{R}^d$.

The method introduced by Delicado is based on local analytics on an arbitrary point $x \in X$. The combination of x with a *unit vector*

$$b \in \{w \in X : \|w\| = 1\} \tag{3.7}$$

defines a *hyperplane* that has normal b and support point x. Denote a orthonormal basis $b_2, \ldots, b_d$ of the hyperplane such that $b, b_2, \ldots, b_d$ is a orthonormal basis for $\mathbb{R}^d$.

The points $x_i \in V$ are, as in the kernel density estimate, the only reference to approximate the data division on a hyperplane $H(x, b)$. To approximate the data division on hyperplane $H = H(x, b)$ all points $x_1, x_2, \ldots, x_n$ are projected onto the hyperplane. The *projection* of x_i is defined by,

$$x_i^H = x_i + \langle x - x_i, b\rangle\, b,$$

where $\langle\,\cdot\,,\,\cdot\,\rangle$ is the Euclidean inner product (2.7). The vector $x_i^H - x$ should than be an vector laying within the hyperplane H and therefore be perpendicular to b.

$$\begin{aligned}
x_i^H - x \perp b &\Leftrightarrow \left\langle x_i^H - x, b\right\rangle = 0 \\
\left\langle x_i^H - x, b\right\rangle &= \langle x_i + \langle x - x_i, b\rangle\, b - x, b\rangle \\
&= \langle x_i - x, b\rangle + \langle \langle x - x_i, b\rangle\, b, b\rangle \\
&= \langle x_i - x, b\rangle + \langle x - x_i, b\rangle\, \langle b, b\rangle \\
&= \langle x_i - x, b\rangle + \langle x - x_i, b\rangle \\
&= \langle x_i - x, b\rangle - \langle x_i - x, b\rangle \\
&= 0
\end{aligned}$$

These projections x_i^H are weighted with a function that weighs the distance of x_i to the hyperplane H. The weight w_i^H of point x_i is defined by

$$w_i^H = w\left(|\langle x - x_i, b\rangle|\right) = w\left(\left\|x_i^H - x_i\right\|\right),$$

With w a decreasing positive function, for example a kernel function.

Example 9 (Contoso)
As an example we define a hyperplane on the Contoso cdata and visualize the projection onto the hyperplane. We take $x = (\text{total volume } = 450, \text{total delta} = 0) \in \mathbb{R}^2$ as support point and $b = (1, 0)$ as the normal of the hyperplane $H(x, b)$. Note that $b_2 = (0, 1)$ is an orthonormal basis of hyperplane $H(x, b)$. The projections of $x_i \in \hat{V}$ onto this hyperplane are visualized in Figure 3.5 where the weights w_i^H are indicated by color.

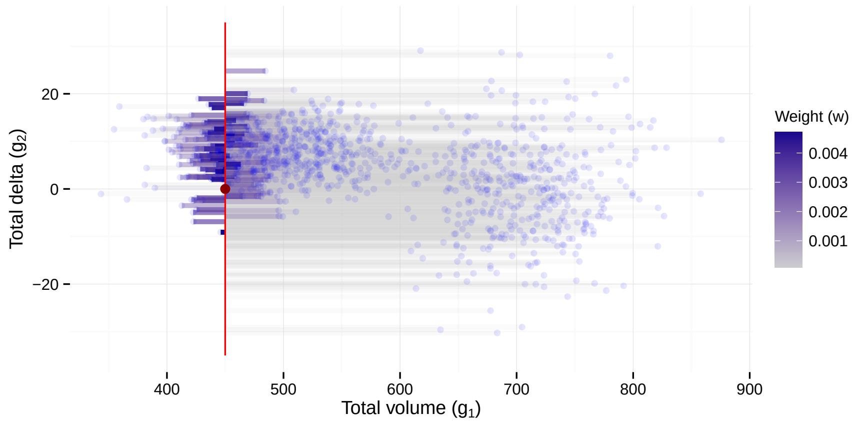


Figure 3.5: Projection of Contoso cdata (g_1, g_2) on hyperplane $H((450, 0), (1, 0))$ given by the red line.

With these weighted projections the data division of hyperplane H is approximated. Statistical analyses on these projections therefore can approximate characteristics of the hyperplane H. Let μ^H denote the *weighted expectation* such that, by definition, it belongs to $H(x, b)$

$$\mu(x, b) = \mu^H = \frac{\sum_{i=1}^n w_i^H x_i^H}{\sum_{i=1}^n w_i^H} \in \mathbb{R}^d. \tag{3.8}$$

For each basis vector b_j for $j = 2, \dots, d$ of hyperplane $H(x, b)$ the *directional weighted expectation* is defined by

$$\mu_{b_j}^H = \frac{\sum_{i=1}^n w_i^H \langle x_i^H, b_j \rangle}{\sum_{i=1}^n w_i^H}.$$

With this expectation $\mu_{b_j}^H$ the *directional weighted variance* for each basis vector b_j is defined by

$$\text{var}_{b_j}^H = \frac{\sum_{i=1}^n w_i^H \left(\langle x_i^H, b_j \rangle - \mu_{b_j}^H \right)^2}{\sum_{i=1}^n w_i^H}.$$

The *total variance* on this hyperplane is defined by the sum of all directional variances for the vectors $b_2, \dots, b_d$ that together define a basis for $H(x, b)$,

$$\phi(x, b) = \phi^H = \sum_{j=2}^d \text{var}_{b_j}^H \in \mathbb{R}. \tag{3.9}$$

The total variance can also be defined by the trace of the *covariance matrix* Σ^H defined on the hyperplane H. The covariance matrix is defined by the covariances between all directions $b_2, \dots, b_d$. The *weighted covariance* between two directions b_j and b_k is defined by

$$\text{cov}_{b_j, b_k}^H = \frac{\sum_{i=1}^n w_i^H \left(\langle x_i^H, b_j \rangle - \mu_{b_j}^H \right) \left(\langle x_i^H, b_k \rangle - \mu_{b_k}^H \right)}{\sum_{i=1}^n w_i^H}.$$

The covariance matrix is defined by

$$\Sigma^H = \begin{bmatrix} \mathrm{cov}^H_{b_2,b_2} & \cdots & \mathrm{cov}^H_{b_2,b_d} \\ \vdots & \ddots & \vdots \\ \mathrm{cov}^H_{b_d,b_2} & \cdots & \mathrm{cov}^H_{b_d,b_d} \end{bmatrix}.$$

Hence, the trace of Σ^H is equal to the total variance (3.9).

Example 10 (Contoso)
For Contoso we represent the division of cdata on the hyperplane H given in Example 9 via a density function. The density is estimated using the kernel density estimate (3.2), where $\mathbf{x}_i$ is x_i^H, the bandwidth h is depending on the directional variance $\mathrm{var}^H_{(0,1)}$ and K is the Gaussian kernel (Example 6). In Figure 3.6 this density estimate, together with the weighted mean $\mu^H_{(0,1)}$ and standard deviation $\sqrt{\mathrm{var}^H_{(0,1)}}$, for Contoso cdata on hyperplane H is plotted.

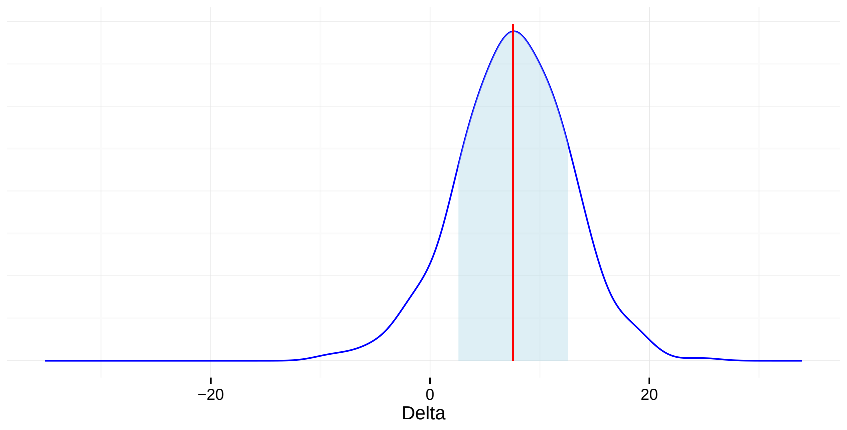


Figure 3.6: Density (blue line), mean (red line) and standard deviation (blue bar) of the weighted projection of Contoso cdata on the hyperplane.

The method of Delicado searches for the hyperplane $H(x,b)$ that minimizes the total variance ϕ^H (3.9). This is done by minimizing the total variance over the unit vectors (3.7) with vector b^* defined by

$$b^* = \arg\min_{b \in \{w \in \mathbb{R}^d : \|w\|=1\}} \phi(x,b). \tag{3.10}$$

The expectation of the hyperplane $H(x,b^*)$ passing through x with minimal total variance defines a function $\mu(x,b^*) = \mu^*(x)$. The fixed points x^* of μ^*, $x^* \in \{x \in \mathbb{R}^d : x = \mu^*(x)\}$, are then called *principal oriented points*. A curve passing through a subset of principal oriented points is called a *principal curve of oriented points*, Delicado 2001.

The method of Delicado uses Algorithm 2 to compute a principal oriented point starting at point $x \in X$. By doing so for multiple x a set of principal oriented points is described. With more unique principal oriented points the principal curve, which has to pass through all these points, is approximated more precise.

To systematically get different principal oriented points that are distinct Delicado proposed a principal curve algorithm, Algorithm 3. This algorithm uses the direction b^* to define new starting points to compute principal oriented points. Let δ be a predefined fixed variable depending on the variance of the data set, the new starting point is defined by

$$x = x^* + \delta b^*.$$

The vector b^* points a direction of relatively high variance, the chances of finding new principal oriented points in this direction is therefore high. Algorithm 3 terminates when the successive values of x^* remain (approximately) constant, this mostly happens when the edge of the data set is reached.

To create a clear view of the algorithms the cdata set $\hat{V}$ is omitted as input in Algorithms 2 and 3, it however is used in most calculations.

Algorithm 2: Principal Oriented Points algorithm

```
**Function** $\{\mu\}$ = POP $(x)$
$k := 0$;
$y^{(k)} := x$;
Compute $b^*$ that defines the Hyperplane $H\left(y^{(k)}, b^*\right)$ with minimal total variance;
**while** $\left\|y^{(k)} - \mu\left(y^{(k)}, b*\right)\right\| > \varepsilon$ **do**
    $k = k + 1$;
    $y^{(k)} = \mu\left(y^{(k-1)}, b^*\right) = \mu^*\left(y_i^{(k)}\right)$;
    Compute $b^*$ that defines the Hyperplane $H\left(y_i^{(k)}, b^*\right)$ with minimal total variance;
**end**
$\mu = y^{(k)}$;
```

Algorithm 3: Principal Curve of Oriented Points algorithm

```
**Function** $\{\alpha_1, \alpha_2, \dots, \alpha_m\}$ = PCOP $(\delta)$
$k := 1$;
Choose $y^{(k)}$ as a suitable starting point;
Set $\alpha_k =$ POP $\left(y^{(k)}\right)$;
**while** $\|\alpha_k - POP(\alpha_k + \delta \cdot b^*)\| > \varepsilon$ **do**
    $k = k + 1$;
    $\alpha_k =$ POP $(\alpha_{k-1} + \delta \cdot b^*)$;
**end**
```

For the theoretical situation where the data are normally distributed stochastic variables the convergence of Algorithm 2 is proven. In this theoretical situation the iteration this Algorithm will converge in one step, for data sets with unknown density function the convergence is assumed but not yet proven, Delicado (2001).

3.3.2 Local Principal Curve

The local principal curve (LPC) method proposed in Einbeck et al. 2005 uses some of the main principal given in Delicado (2001).

One of the main differences between the LPC method and Delicado's method is that local center of mass is used instead of principal oriented points.

Definition 3.3.1 (Local center of mass)
Let $x_1, \dots, x_n$ be the cdata set $\hat{V}$ and K the d-dimensional Gaussian kernel function with bandwidth h, the *local center of mass* around point $x \in X$ is defined by

$$\mu(x) = \mu^x = \frac{\sum_{i=1}^{n} K(x - x_i)x_i}{\sum_{i=1}^{n} K(x - x_i)}.$$

Another difference is that the LPC method uses the first principal component of the local covariance matrix rather than the vector b^* (3.10).

Definition 3.3.2 (Local covariance matrix)
Let weights w_i be defined by

$$w(x, x_i) = w_i = \frac{K(x - x_i)}{\sum_{i=1}^{n} K(x - x_i)},$$

the (j, k)-th entry $(1 \leq j, k \leq d)$ of the *local covariance matrix* around point x, denoted by $\Sigma^x = (\sigma^x_{jk})$, is then defined by

$$\sigma^x_{jk} = \sum_{i=1}^{n} w_i (x_{ij} - \mu^x_j)(x_{ik} - \mu^x_k).$$

The matrix Σ^x may also be defined via matrix multiplication: Let matrix Δ be defined by

$$\Delta = \begin{bmatrix} x_{11} - \mu^x_1 & \cdots & x_{n1} - \mu^x_1 \\ \vdots & \ddots & \vdots \\ x_{1d} - \mu^x_d & \cdots & x_{nd} - \mu^x_d \end{bmatrix},$$

then the local covariance matrix Σ^x is defined by

$$\Sigma^x = \Delta \cdot \operatorname{diag}(w_1, w_2, \ldots, w_n) \cdot \Delta^T.$$

Define the *first principal component* γ^x as the first eigenvector of Σ^x corresponding to the greatest (absolute) eigenvalue λ^x_1. Note that the greatest eigenvalue of Σ^x is equal to the first singular value of Σ^x.

With the local center of mass and the local covariance matrix we define the LPC algorithm as given in Algorithm 4. Notice the similarities between this algorithm and Algorithm 3.

Algorithm 4: Local Principal Curve algorithm

```
Function $\{\alpha_1, \alpha_2, \ldots, \alpha_m\} = \text{LPC}(\delta, \{x_1, x_2, \ldots, x_n\})$
$k := 1$;
Choose $y^{(k)}$ as a suitable starting point;
$\alpha_k = \mu\left(y^{(k)}\right)$;
Perform a principal component analysis locally at $y^{(k)}$, to obtain $(\lambda_1^{y^{(k)}}, \gamma^{y^{(k)}})$;
while $\left\| \alpha_k - \mu\left(\alpha_k + \delta \cdot \gamma^{y^{(k)}}\right) \right\| > \varepsilon$ do
    $k = k + 1$;
    $y^{(k)} = \alpha_{k-1} + \delta \cdot \gamma^{y^{(k-1)}}$;
    $\alpha_k = \mu\left(y^{(k)}\right)$;
    Perform a principal component analysis locally at $y^{(k)}$, to obtain $(\lambda_1^{y^{(k)}}, \gamma^{y^{(k)}})$;
end
```

Example 11 (Contoso)
As an example we apply Algorithm 4 to the Contoso cdata set. This creates a local principal curve that describes the data set. The curve and the data set are both plotted in Figure 3.7.

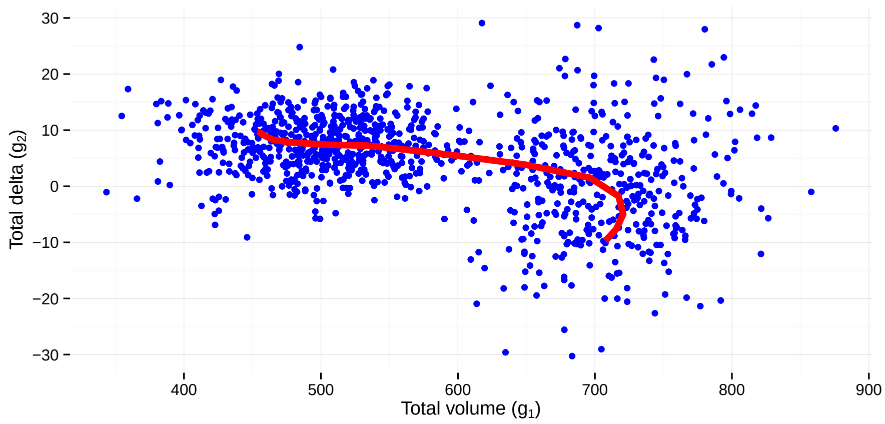


Figure 3.7: Local principal curve through Contoso cdata.

Coverage

In Einbeck et al. 2005 there is also a notion of coverage suggested. *Coverage* is a function that depends on points P_α that describe the principal curve α and a value τ. The coverage is defined by the percentage of points that is within a distance τ from any point $p \in P_\alpha$,

$$\mathfrak{C}_m(\tau) = \# \{x \in \{x_1, \ldots, x_n\} | \exists p \in P_\alpha \text{with} \, \|x - p\| \leq \tau\} /n.$$

The smaller the minimum value of τ such that $\mathfrak{C}_m(\tau)$ is at a certain threshold, the better the principal curve describes the data set.

The derivative of function $\mathfrak{C}_m$ can also be used as an indication of the *goodness of fit* for a set of points P. When the function increases fast the points P_α lie in a dense part of the data set, therefore giving a good description of the data set.

The function $\mathfrak{C}_m$ can thus be used to describe the characteristics of the local principal curve α. It also can be used to describe characteristics of *cluster centers*. Clusters centers can for instance be defined by centers of mass or accumulation points of clusters or may be method specific as the modes in Mean Shift clusters. Implementing the centers as P_α into the function $\mathfrak{C}_m$ will then describe a goodness of fit for the clusters.

Clustering

With the use of local principal curves we have a great tool to represent data. This representation generates insight in data that can be used to optimize analyses. However, local principal curves do not result in clusters, which is the desired result. To this end I developed a *LPC clustering method* that uses the notion of coverage to create clusters.

The LPCM package Einbeck and Evers (2013) has a build in function that approximates the principal curve with a spline. In this function one can define a predefined number of points k that are used to indicate this spline. We define the *k-spline* as the spline defined by k points.

Using the coverage function $\mathfrak{C}_m$ the goodness of fit for the k-spline to the data set can be indicated. Note that the coverage of the local principal curve indicates a *maximum coverage increase* for a k-splines.

The clustering method searches for the minimal k such that such that the distance between the coverage of the k-spline and the maximum coverage increase is within a certain threshold. The k points are then used as cluster centers and each point $x_i \in \hat{V}$ is linked to the nearest cluster center with respect to the the standard Euclidean distance measure. The points linked to the same cluster center then form a cluster.

Example 12 (Contoso)
We apply the method of clustering using local principal curves to the Contoso cdata set. In Figure 3.8 the coverage $\mathfrak{C}_m$ is plotted for different k-splines as well as for the local principal curve.

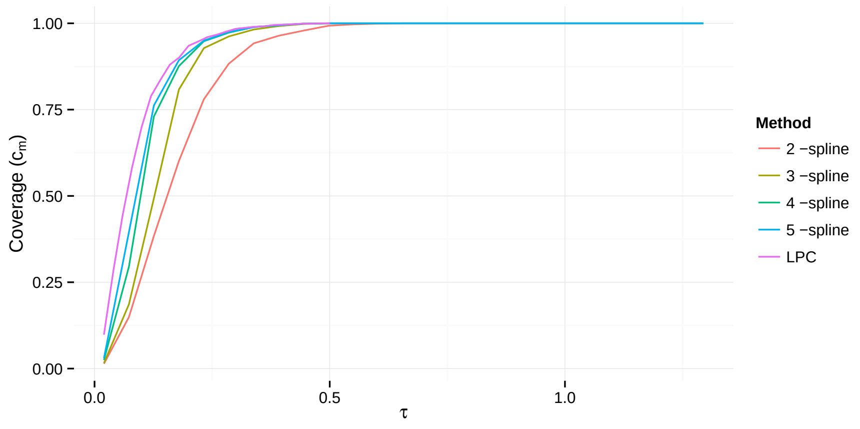


Figure 3.8: Coverage $\mathfrak{C}_m$ of the Contoso cdata in percentage for different k-splines and the local principal curve.

Using Figure 3.8 as reference conclude that the 5-spline is a good representation, in terms of coverage, of the Contoso cdata. To create clusters the 5 points describing the 5-spline are used as cluster centers. In Figure 3.9 these centers are indicated with a red dot, the 5 clusters they define on the Contoso cdata set are colored.

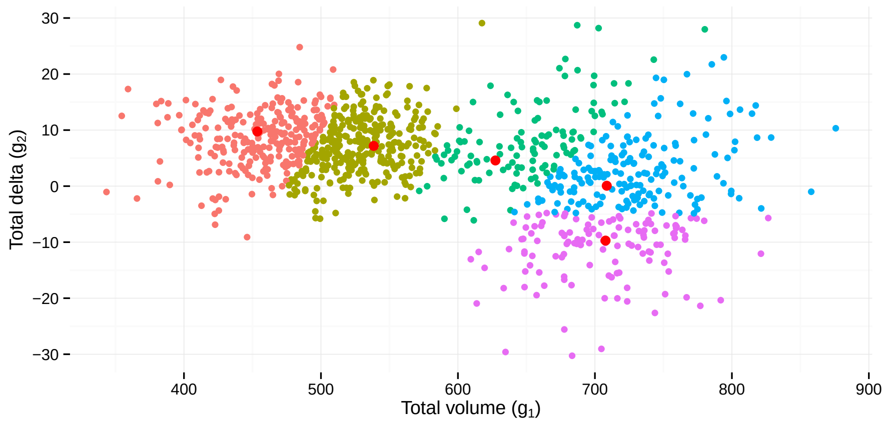


Figure 3.9: Clusters on Contoso cdata computed with the LPC clustering method (Each cluster differently colored, centers indicated).

3.4 Comparison

The main difference between the Mean Shift and the LPC method is their foundation. the LPC method searches for a curve best describing the cdata set. From that curve the clusters are defined using k-splines. The Mean Shift method on the other hand searches for modes. Each point is linked to a mode via Mean Shift iteration.

In Figure 3.10 the coverage $\mathfrak{C}_m$ of the Mean Shift cluster centers is plotted against the coverage of the k-splines and the LPC. The Mean Shift gives a better coverage than the 2-spline. However, in comparison with k-splines with $k > 2$ the Mean Shift has a lower coverage increase. In terms of coverage the LPC method is therefore better.

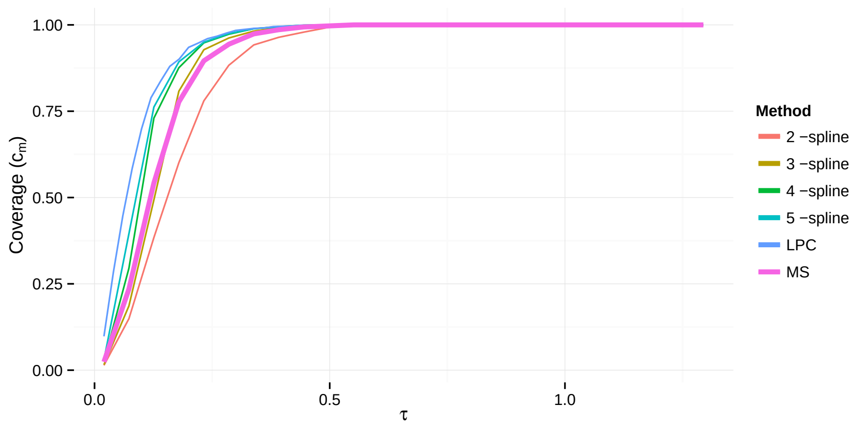


Figure 3.10: Coverage $\mathfrak{C}_m$ of the Contoso cdata in percentage for different k-splines, the local principal curve and the Mean Shift cluster centers.

The definition of clusters defines that points from the same cluster have to be similar and points from different clusters are to be dissimilar. The LPC method does not cope with the second condition as well as the Mean Shift method. However, the local principal curve can consist of multiple disconnected parts ensuring that elements of obviously different clouds are in different clusters.

The last difference is speed. The calculation of the curve through a data set is fast and afterwards, with little calculation time, each point x_i can be linked to one of the created cluster centers. Finding an optimal k with the use of coverage does however take more time. For the Mean Shift method the modes are derived via an iterative process. This iterative process has to be done for every point $x_i \in \hat{V}$. Altogether this makes the Mean Shift method slower in comparison to the LPC Method.

Both clustering techniques still have a lot of possible optimizations. Both use predefined variables, for which the optimal values are unknown and differ per data set. The search or calculation of these optima however is a different study and is not discussed in this thesis.

CHAPTER 4

Time Series

As stated in Chapter 2, a setback of the static cdata is that it does not represent dynamic behavior of the original data entries in the time series. The amount of dynamics lost is visualized in the figures in Figure 4.1. In each of these figures the cumulative volume over time is plotted for two different customers. Each customer shows a different buying behavior but ends up with the same total volume g_1 at the end of the visualized period. By creating a static cdata for these customers, as described in (2.12), their representation will be equal, such that via similarity checks and distance measures they will be considered equal.

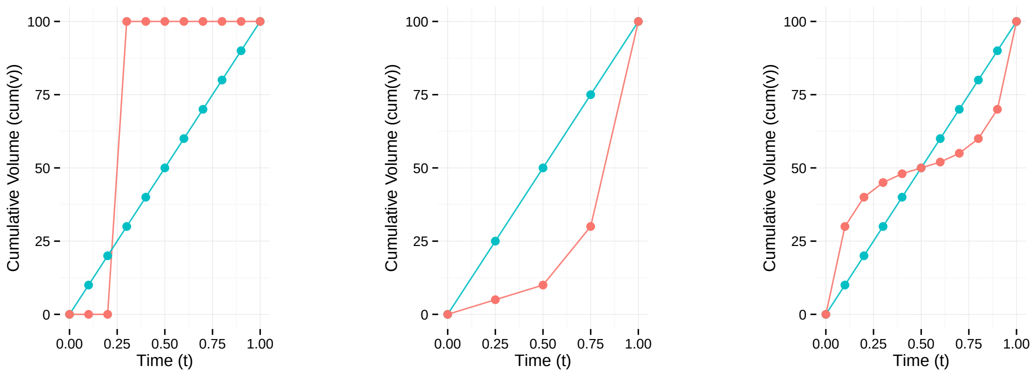


Figure 4.1: 3 pairs of two customers with different characteristics but equal total volume.

For the remainder of this thesis, v_i^k denotes the volume purchased by customer c_i at time t_i^k. An array of volumes is denoted by v, such that

$$v = \left[v^1, v^2, \ldots, v^P\right].$$

Definition 4.0.1 (Cumulative volume)
The array of *cumulative volumes cum(v)* is defined by

$$\operatorname{cum}(v) = \left[v^1, \textstyle\sum_{l=1}^{2} v^l, \ldots, \sum_{l=1}^{P} v^l\right]. \tag{4.1}$$

In this chapter we more extensively define customer time series $(t,d)_i$ and describe a time series visualization method. After this properties of time series in the ACME data set are stated, focusing on data entry dynamics.

4.1 Time series

Since we want to consider the dynamics of data entries over time, the static cdata in $\hat{V}$ (as computed by g (2.12)) are not sufficient for future analyses. To get better insight in the specifics of dynamics the original points $x \in V$ themselves have to be analyzed. Therefore we analyze the customer time series $(t,d)_i$ rather than the created static cdata.

Define

$$\tilde{V} = \{(t,d)_1, (t,d)_2, \ldots, (t,d)_n\} \subset \tilde{X}$$

as the set of n time series each representing a unique customer $c_i \in C$.

The time series $(t,d)_i \in \tilde{V}$ are strictly increasing with respect to the time of measurement t_i^k, such that

$$t_i^k < t_i^{k+1} \qquad \text{for } k = 1, \ldots, P_i - 1. \tag{4.2}$$

This to prevent division by zero when differentiating over time.

The time series as given in this thesis are truncated outside the time period of interest. The period is indicated by the same period delimiters dat_1 and dat_2 defined in Chapter 2. The timestamps are then transformed to non-dimensional time(stamps) $\hat{t} \in [0,1]$ using the operation

$$\hat{t} = \frac{t - dat_1}{dat_2 - dat_1}.$$

The notation $\hat{t}$ is omitted such that t indicates the normalized time.

The quality data entries are not non-dimensionalized to keep in touch with the magnitudes instead. Also, during the given analyzes the quality data entries will be normalized anyway such that non-dimensionalizing would be obsolete.

Definition 4.1.1 (Time series path)
We define the function $p_i : [t_i^1, t_i^{P_i}] \mapsto \hat{D}$ as the piece wise linear curve constructed by linear interpolation between the quality data points d_i^k in time series $(t,d)_i$. The piece wise linear function for $p_i(t) : [t_i^1, t_i^{P_i}] \mapsto \hat{D}$ is defined by

$$p_i(t) = p\left((t,d)_i, t\right) = d_i^k + \frac{t - t_i^k}{t_i^{k+1} - t_i^k} \cdot (d_i^{k+1} - d_i^k) \qquad \text{if } t_i^k \leq t \leq t_i^{k+1} \qquad \text{for } k = 1, \ldots, P_i - 1. \tag{4.3}$$

We use the term "path" to refer to this piece wise linear function p_i.

Example 13 (Volume)
The *volume time series* $(t,v)_i$ is the time series $(t,d)_i$ containing only volume data , such that

$$d_i^k = v_i^k.$$

Its path, equal to (4.3), is redefined by

$$vp_i(t) = p\left((t,v)_i, t\right) = v_i^k + \frac{t - t_i^k}{t_i^{k+1} - t_i^k} \cdot (v_i^{k+1} - v_i^k) \qquad \text{if } t_i^k \leq t \leq t_i^{k+1} \qquad \text{for } k = 1, \ldots, P_i - 1.$$

Henceforth we call this function vp_i the *volume path.*

Example 14 (Cumulative volume)
The *cumulative volume time series* $(t, \mathrm{cum}(v))_i$ is the time series $(t, d)_i$ containing only cumulative volume data, such that

$$d_i^k = \mathrm{cum}(v_i)^k = \sum_{l=1}^{k} v_i^l$$

Its path, equal to (4.3), is redefined by

$$cvp_i(t) = p\left((t, \mathrm{cum}(v))_i, t\right) = \sum_{l=1}^{k} v_i^l + \frac{t - t_i^k}{t_i^{k+1} - t_i^k} \cdot v_i^{k+1} \qquad \text{if } t_i^k \leq t \leq t_i^{k+1} \qquad \text{for } k = 1, \dots, P_i - 1.$$

Henceforth we call this function cvp_i the *cumulative volume path.*

Figure 4.2 shows a pair of time series $(t, d)_i, i = 1, 2$ for which the volume paths vp_i, shown in the right figure, are equivalent. The customer indicated by the red time series however eventually bought more volume in the given time period, which is better indicated by the cumulative volume paths in the right figure. From a business-point of view we think we should nevertheless distinguish the behavior of both time sequences (we would like them to eventually be in different clusters). We do not know how to proceed (best), but obviously one possibility is to use the cumulative volume paths in Figure 4.2.

Note that the value of the cumulative volume path cpv_i at t_i^k is given by

$$cpv_i(t_i^k) = \sum_{l=1}^{k} v_i^l \qquad \text{for } k = 1, \dots, P_i - 1.$$

Assuming equidistant time steps

$$t_i^k = \Delta t \cdot k, \qquad k = 1, \dots, P_i,$$

with $\Delta t = \dfrac{1}{P_i}$ such that $t_i^k \in [0, 1]$ and $t_i^{P_i} = 1$. The integral of the volume path vp_i is

$$\begin{aligned} \int_0^{t_i^k} vp_i(t)dt &\approx \sum_{l=1}^{k} v_i^l \cdot \Delta t, \\ &= \mathrm{cum}(v_i)^k \cdot \Delta t, \\ &= cvp_i(t_i^k) \cdot \Delta t, \end{aligned}$$

for $k = 1, \dots, P_i$. Hence the cumulative volume path approximates a (Δt^{-1})-scaled version of the integrand of the volume path. Our ACME data is (unfortunately) such that many of its volume time series $(t, v)_i$ are highly irregular (see below), which implies that

$$cvp_i(t) \neq P_i \cdot \int vp(t)dt$$

in general.

The cumulative volume time series can be created via a simple operation on the volume data entry, see (4.1). Therefore all its dynamics are also found in the original volume data entry. Hence, we from now on only consider original data entries as in volume time series.

Example 15 (Contoso)
In Table 4.1 the data entries are given for points inside the time series for the Contoso data set. The data entries are equal to the original point entries, Table 2.1. The customer and product entries however become superfluous because the data contains only one product such that each time series $(t, d)_i$ represents a unique customer c_i.

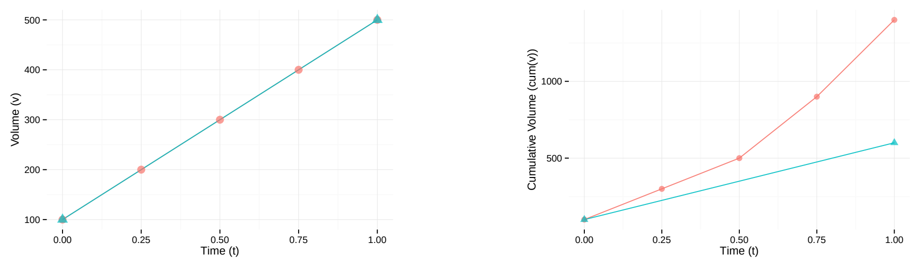


Figure 4.2: Volume path (left) and cumulative volume path (right) for two time series.

Data entry	Data Category	Description
Timestamp	T	The non-dimensional time.
Volume	D	The volume delivered.
Delta	D	The difference of volume ordered and volume delivered.

Table 4.1: Contoso dynamic data entries used for points in time series.

Example 16 (ACME)
The entries of the ACME data time series are similar to those described in Table 4.1, with the exception of the delta variable which is non existing in the ACME data set.

To construct well defined time series from the ACME data set it needs to be regularized. Next to the regularization steps as described in Example 3, some time series specific regularization is needed. The combinations of c_i and t_i^k have to be unique, such that no two transactions can occur at the same time for the same customer. To meet this conditions (4.2) the transactions occurring at the same time from the original ACME data set had to be accumulated. To this end we take the sum of their volumes as the volume of a new transaction.

All together, the ACME time series give an almost exact representation of the original data set meaning that it can be used to show and analyze its dynamics.

4.2 Dynamics of time series

In this section we examine characteristics of the time series. Since time is a new variable to be considered the behavior of the data set with respect to time becomes essential. To this end we study whether the data sets contain regularity issues.

Definition 4.2.1 (Regularity issues)
Regularity issues of a data set depend on the values $t_i^{k+1} - t_i^k$, $d_i^{k+1} \pm d_i^k$.

Example 17 (Contoso)
The Contoso data set is created such that it does not contain any regularity issues. At the same time it represents some of the possible time series that occur in the ACME data set. The exclusion of irregularities is such that future results on the ACME data set are understandable without the explaining effects of outliers.

The Contoso time series have lengths varying between 2 and 9 transactions. The times between transactions are computed using the exponential distribution with a normal distributed λ. The data set is split into three time series groups; One growing, one decaying and one steady.

The volume paths vp_i of all Contoso time series $(t, d)_i$ are plotted in Figure 4.3. To get a better view on individual volume paths the volume paths of a random customers sample of size 100 is plotted in Figure 4.4. Note that every time series also has a delta entry, such that the original time series path p_i is a multidimensional curve.

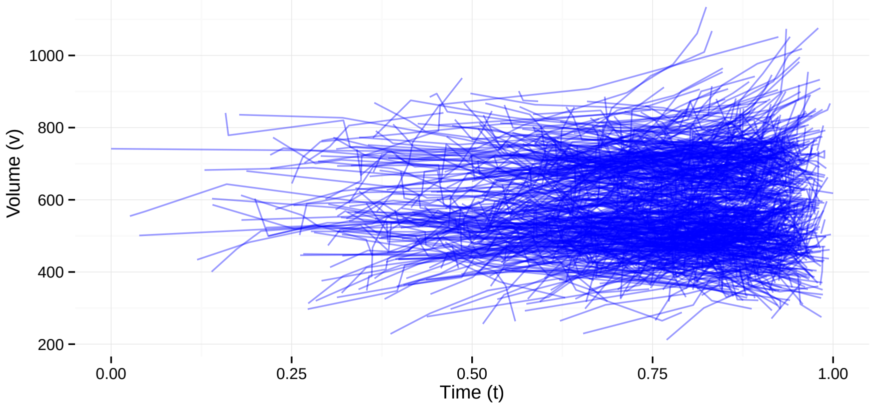


Figure 4.3: Volume paths for the Contoso time series.

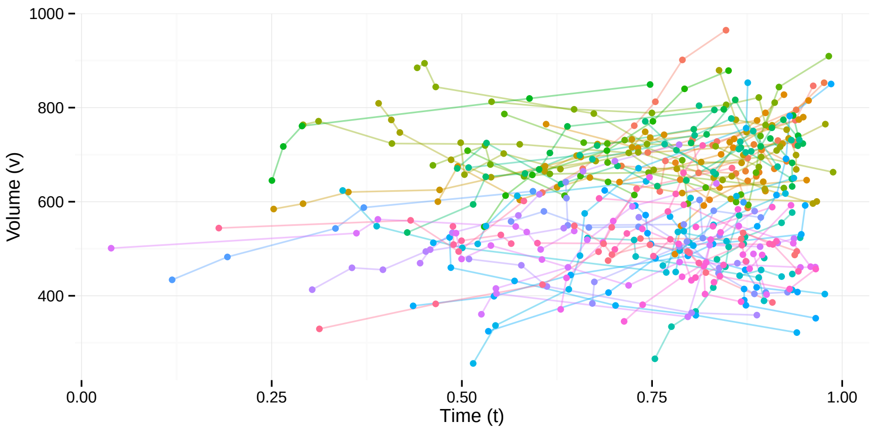


Figure 4.4: Volume paths for a time series sample of size 100 from the Contoso time series.

Example 18 (ACME)
The ACME data set is slightly modified to fit the mathematical time series model, see Example 16. But even after fitting the data still has a lot of regularity issues.

In Figure 4.5 the volume path of each time series in the ACME data set is plotted. To create a better view on individual volume paths the volume path of a random customer sample of size 30 is plotted in Figure 4.6. Note that a lot of irregularities exist in the volume paths of ACME customers. An in depth view on irregularities can not be obtained by looking at Figure 4.6, therefore we summarize some of the regularity issues that are found in ACME time series.

Example 19 (Cancelling correction)
Faulty transaction volumes in the data set are corrected by implementing a new transaction. This new transaction then has the same volume but different sign, such that the volumes

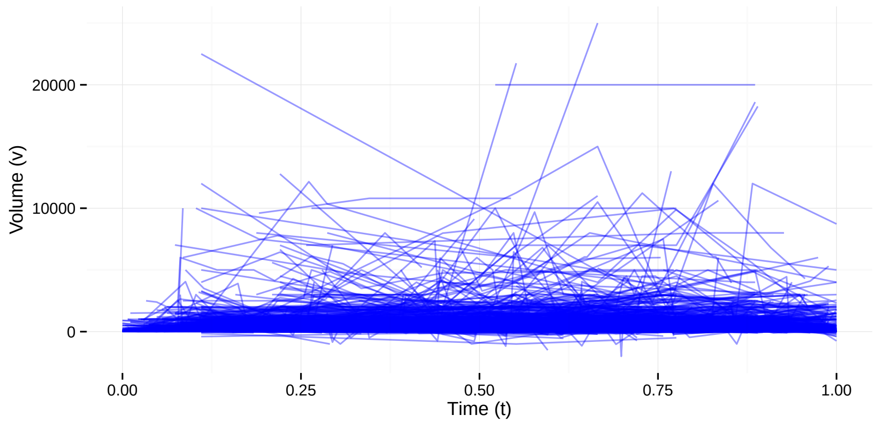


Figure 4.5: Volume paths for the ACME time series.

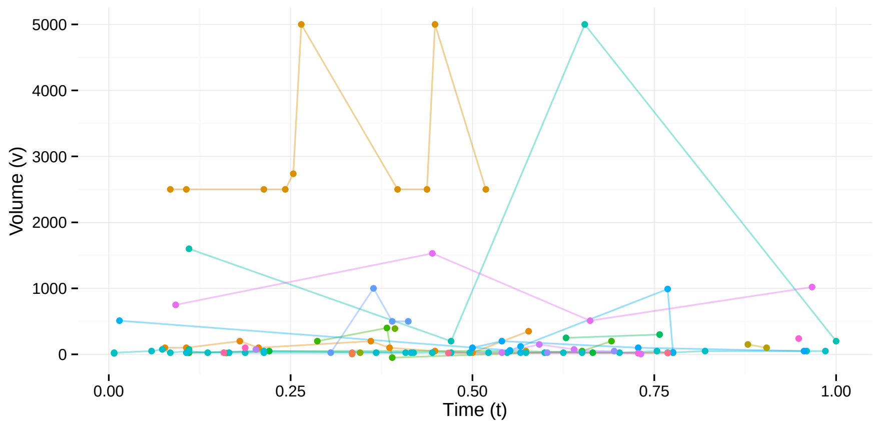


Figure 4.6: Volume paths for a time series sample of size 30 from the ACME time series.

cancel each other out. This creates for example time series $(t, v)_i$ with volumes

$$v_i = (1000, -1000), \tag{4.4}$$

where the second transaction cancels the first. Unfortunately these corrections ensure relatively large differences in transactional volumes.

In (4.4) the correcting transaction is subsequent in time to the original, this however is not standard. This creates for example time series $(t, v)_i$ with volumes

$$v_i = (1000, 50, -1000),$$

also occur, making it hard to detect what transaction was originally cancelled.

Example 20 ("Smart" cancelling corrections)
Faulty transaction volumes are also corrected in a "smarter" way, where the difference with the correct volume is implemented as a new transaction. This creates for example time series $(t, v)_i$ with volumes

$$v_i = (1000, -860),$$

where the correct volume is 140. These "smarter" corrections do not necessarily occur subsequent in time, as cancellations in Example 19. Altogether these corrections are hard to detect and correct.

Example 21 (Correction times)
Some of the wrong volumes in the transaction data set are noticed quickly, such that they are corrected within a short amount of time. Meaning that the time period $t_i^{k+1} - t_i^k$ then becomes small with respect to $[t_i^1, t_i^{P_i}]$. Also normal transactions may be entered shortly after eachother, for instance via small back order, resulting in relatively small time periods.

Definition 4.2.2 (Irregular time series)
We define *irregular time series* as time series $(t, v)_i$ that contain either

- Small number of transactions P_i, for instance 1 or 2
- Small interval $[t_i^1, t_i^{P_i}]$ inside the time window $[0, 1]$
- large differences between intervals

$$\frac{\max_k(t_i^{k+1} - t_i^k)}{\min_k(t_i^{k+1} - t_i^k)} >> 1$$

- $t_i^{k+1} = t_i^k$ (should not occur but does)
- $v_i^k < 0$ (should also not occur but does)

An *irregular data set* is a data set containing irregular time series.

CHAPTER 5

Distance Measures

Most clustering methods such as those presented in Chapter 3 depend on distances $d(x,y)$ in $\mathbb{R}$ between cdata points x, y in $\mathbb{R}^d$. The employed distance measure determines the "value" of the obtained clusters because it determines whether (time-dependent) behavior is similar. From the business perspective it seems desirable that similar customer behavior leads to about zero distance. For instance customers that increasingly purchase more volume at every new order should be considered (more) similar and should have distance (close to) zero. Thus, we need to construct a distance measure which takes this business wish into account.

For the similarity check we need to construct a distance measure of the form

$$d\left((t,d)_i,(t,d)_j\right) \mapsto \mathbb{R}.$$

In Liao (2005) a list of existing distance measures on time series is given. The distance measures in this list can be divided into two categories

1. Methods which require
$$t_i^k = t_j^k \qquad \forall i \neq j = 1,\dots,n, \qquad \forall k = 1,\dots,\min(P_i,P_j), \tag{5.1}$$
which permits these methods to ignore the time (equidistance $t_i^{k+1} - t_i^k = \Delta t$ is not required).
2. Methods which do not require (5.1), where we take Dynamic Time Warping (DTW) as an example.

During two multi houred CASA-question meetings with members from the CASA group it was concluded that (from the distance measures which take transient behavior into account) there was no obvious distance measure because for the categories specific:

1. Before using these methods a lot of regularization on the original data is needed. As seen in the Figures 4.6 and 4.4 the data does not at all fit the requirements for this category (5.1). To fit the data first needs to be inter- or extrapolated. The amount of data that becomes oversimulated or neglected through these operations is unclear.
2. Computing distances with DTW is computationally expensive and it is not clear beforehand how this method relates to distance measures from (1).

The uncertainty and unclarity of distances computed with the measurements in Liao (2005) on our data sets together with the possible costly implementations, made us decide to analyze a simpler set of distance measures

$$\left((t,d)_i,(t,d)_j\right) \mapsto d\left(g((t,d)_i), g((t,d)_j)\right),$$

where g is an *indexing function*, i.e. a function with as input a time-series and as output a real number. This group of distance measures is further investigated in the next chapter.

In this chapter a short summary of some of the distance measures in Liao (2005) is given. This to argue the transition to simpler distance measures that use indexing functions $g((t,d)_i)$.

5.1 Euclidean distance measures

The most commonly used distance measure in multiple dimensions is the *Euclidean distance measure*. This is also the distance measure used in the clustering methods presented in Chapter 3. In Liao (2005) several versions and generalizations of the Euclidean distance measure are given. The versions are defined for time series $(t,d)_i$ with one dimensional d_i^k of fixed length P that belong to the first category (5.1).

The ACME data set does not meet the requirements (5.1) stated for the first category, as seen in Figure 4.6. Therefore the original time series have to be regularized and or simulated before Euclidean based distance measures can be used. The method to be used for these regularization and simulation steps bring up enough problems and unanswered questions to argue whether or not this function is useful at all.

Below we argue whether the Euclidean based distance measures would fit the business model if there exists a correct way to fit the ACME data set to the first category of distance measures.

Definition 5.1.1 (Euclidean distance)
Let $(t,d)_i$ and $(t,d)_j$ each be a time series of length P. The Euclidean distance is computed as

$$d_E\left((t,d)_i,(t,d)_j\right) = \sqrt{\sum_{k=1}^{P}\left(d_i^k - d_j^k\right)^2}.$$

The Euclidean distance measure computes the sum of the squared differences between the two time series at all time stamps, without using the amount of times P_i or the actual times t_i^k (or differences $t_i^{k+1} - t_i$) themselves. Therefore different behavior in time can lead to the equal distances as shown in Figure 5.1.

The extension of the Euclidean distance measures to fit time series of different lengths is not trivial. One method is to exclude non existing time series elements in the distance measure. However this would result in an unbalanced measure when comparing long with short time series.

To cope with time series of different length the distance could be averaged, resulting into the root mean square distance.

Definition 5.1.2 (Root mean square distance)
Let $(t,d)_i$ and $(t,d)_j$ each be a time series of length P. The *root mean square distance*, also known as the *average geometric distance*, is computed as

$$d_{\text{rms}}\left((t,d)_i,(t,d)_j\right) = \frac{1}{P}\sqrt{\sum_{k=1}^{P}\left(d_i^k - d_j^k\right)^2} = \frac{d_E\left((t,d)_i,(t,d)_j\right)}{P}.$$

Note that the root mean square distance measure is a discretization of the well known L_2 norm.

Because of the averaging over the amount of transactions P perhaps this distance measure applied to an actual (ACME) data set will lead to clusters which provide more insight. However the actual times t_i^k (or differences $t_i^{k+1} - t_i$) themselves are not considered in this distance measure, such that it still has shortcomings.

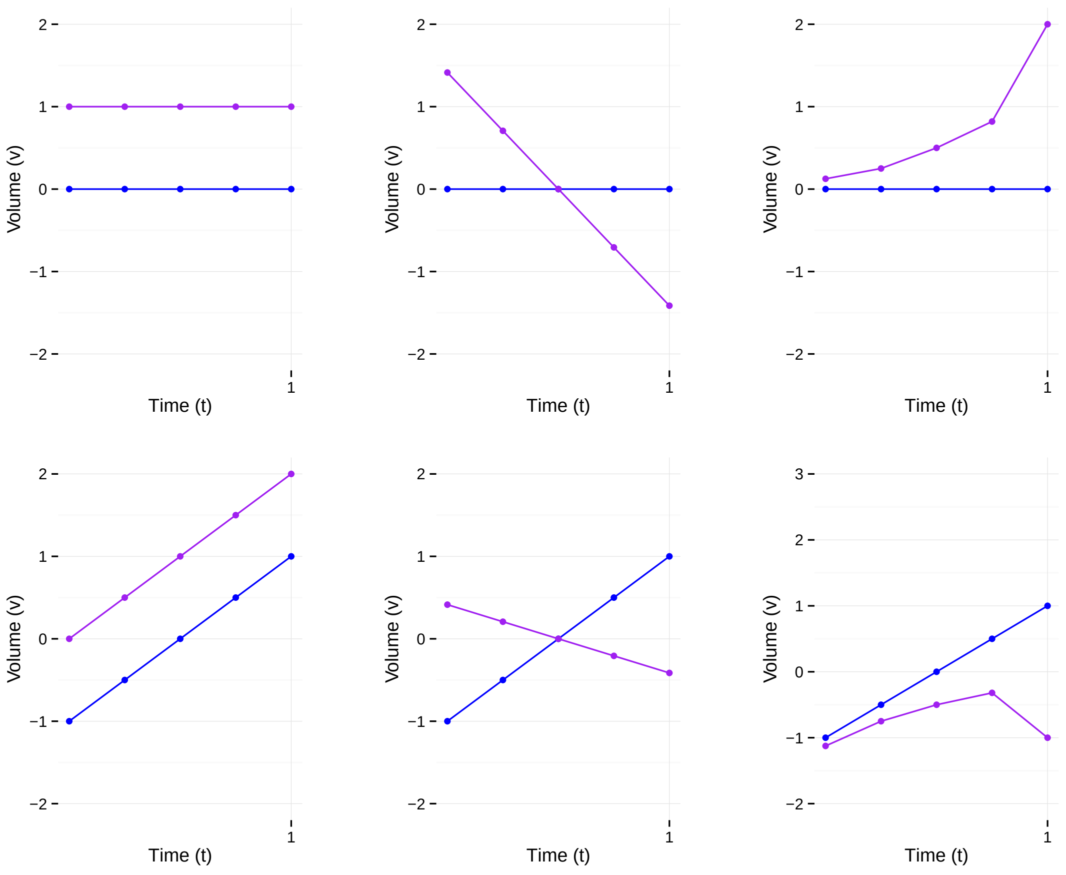


Figure 5.1: Different pairs of time series that all have a distance equal to $\sqrt{5}$.

Definition 5.1.3 (Minkowski distance)
Let $(t,d)_i$ and $(t,d)_j$ each be a time series of length P. The *Minkowski distance*, a generalization of the Euclidean distance, is computed as

$$d_{E(q)}\left((t,d)_i,(t,d)_j\right) = \left(\sum_{k=1}^{P} \left|d_i^k - d_j^k\right|^q\right)^{1/q},$$

where $q > 0$ is a positive real number.

The Minkowski distance measure is more general than the Euclidean distance measure. With the value q the impact of outliers and or fluctuations on the distance can be managed. To show this concept the unit circles in two dimensions for various values of q are given in Figure 5.2. Here each dimension represents a timestamp and it is seen that for smaller q the differences at these timestamps have less impact on the distance.

The Minkowski distance measure, as the Euclidean distance measure, can be averaged in order to cope with time series length differences. Note that the averaged Minkowski distance measure is a discretization of the L_Q norm.

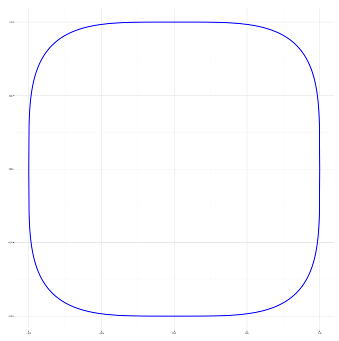
The actual times t_i^k (or differences $t_i^{k+1} - t_i$) themselves are not considered in all Euclidean based distance measures, as illustrated in Figure 5.1. They also do not cope very well with small time shiftings.

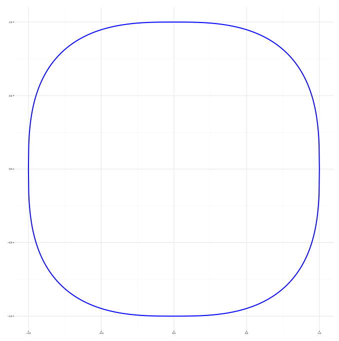
We consider the two time series $(t,d)_1$ and $(t,d)_2$ of length P, where

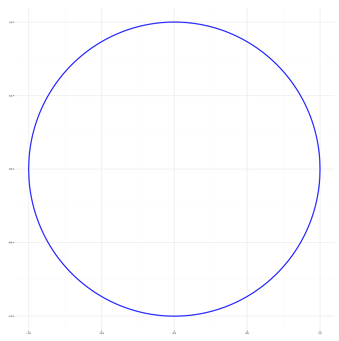
$$t_2^k = t_1^k + c \qquad \forall k = 1,\ldots,P,$$

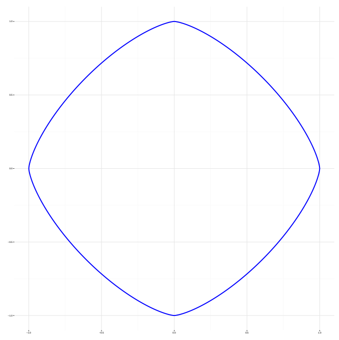
Here $(t,d)_2$ is delayed from the time series $(t,d)_1$ by a constant c. The value of the constant c will have impact on the distance, such that even small c may result in a lot of distance between the two time series. This makes the Euclidean based distance measure unstable with respect to time shiftings.

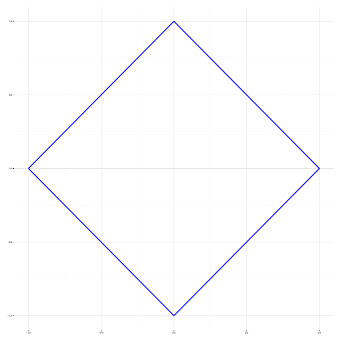
From a business point of view, the growth, decay and or shiftings are of importance on time series similarities. Since none of these elements are considered correctly through the Euclidean based distance measures, we conclude that they are not sufficient to accurately indicate similarities between customer time series.

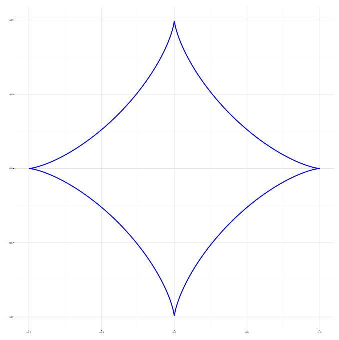

(a) $q = 2^2 = 4$

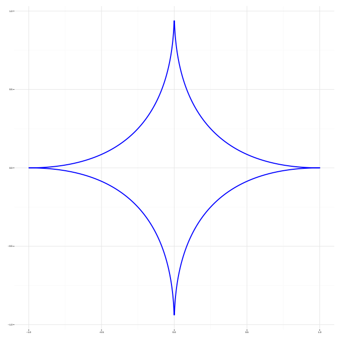

(b) $q = 2^{1.5} = 2.828$

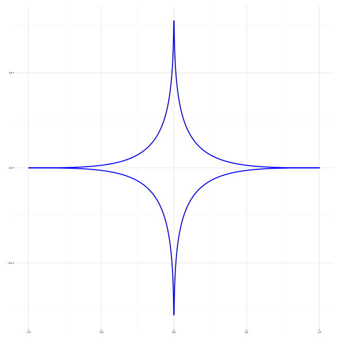

(c) $q = 2^1 = 2$


(d) $q = 2^{0.5} = 1.414$


(e) $q = 2^0 = 1$


(f) $q = 2^{-.5} = .707$


(g) $q = 2^{-1} = .5$


(h) $q = 2^{-1.5} = 0.354$

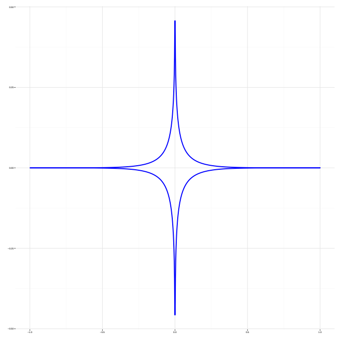

(i) $q = 2^{-1} = 0.25$

Figure 5.2: Unit circles of the Minkowski distance in a 2 dimensional space for different values of q

5.2 Dynamic Time Warping

Dynamic Time Warping (*DTW*) is a distance measure of the second category, it therefore can be directly applied to our transaction time series. In DTW two time series are stretched in time in order to find a best fit. The stretching is done via warping paths, describing links between elements from each of the time series. The best fit is the warp path that indicates the stretched time series with minimal distance.

DTW is applied on time series such that patterns rather than positions are compared. For instance, walking patterns could be compared, even if one person is walking faster than the other. Another well known application is speech recognition, where it copes with different speaking speeds. DTW is also used in shape and signature recognition.

The DTW method, as described below, is based on time series $(t,d)_i$ with one dimensional quality data d_i^k. The length and the moment of measurement can vary per time series, such that time series are of the form

$$(t,d)_i = (t_i^1, d_i^1), (t_i^2, d_i^2), \dots, (t_i^{P_i}, d_i^{P_i}),$$

for $i = 1, \dots, n$. Which is equal to the form of our transaction time series.

Definition 5.2.1 (Warping path)
Time warping is a method used in Dynamic time warping. This method creates a warping path describing links between points of two time series.

Let $(t,d)_i$ and $(t,d)_j$ be two time series of lengths P_i and P_j respectively, time warping is defined by a constructed *warp path* w

$$w = w_1, w_2, \dots, w_H \qquad \max(P_i, P_j) \leq H < P_i + P_j,$$

where H is the length of the warp path. Each element $w_h \in w$ describes a link between two points, each from one of the two time series $(t,d)_1$ and $(t,d)_2$. When w_h connects (t_i^k, d_i^k) to (t_j^l, d_j^l) it is simply written as a tuple

$$w_h = (k, l) \quad \in \mathbb{N}^2.$$

The warp path is complete when it satisfies the following three constraints:

I Boundary conditions: The warp path has to start by connecting the first time series points and end by connecting the last of the two time series,

$$w_1 = (1,1),$$
$$w_K = (P_i, P_j).$$

II Continuity: Two adjacent tuples, w_h and w_{h+1} for $h = 1, \dots, H-1$, in the warp path can only have a maximum norm (2.6) of 1,

$$\|w_{h+1} - w_h\|_{\max} \leq 1.$$

III Monotonically increasing: Under the maximum norm two adjacent tuples, w_h and w_{h+1} for $h = 1, \dots, H-1$, from the warp path w have a minimal distance of 1,

$$\|w_{h+1} - w_h\|_{\max} \geq 1.$$

Concluding that two adjacent tuples have to have a maximum norm of 1. The three conditions ensure that every point of each time series is connected to a point on the other via a warp tuple and no duplicates or crossings exist in the warp path.

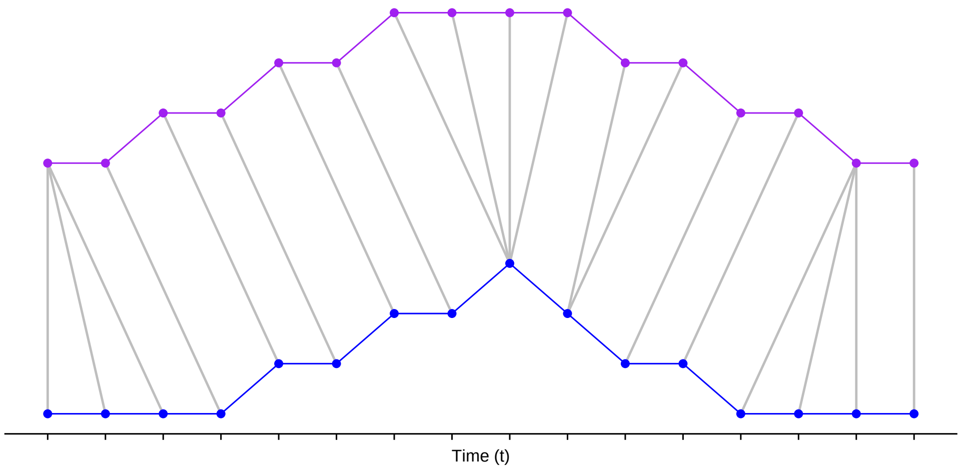


Figure 5.3: A warping between two time series

Note that from the definition it follows that the number of possible warping paths will be very large for reasonably sized time series. The distance of a warp path is based on distances between time series points. Each time series $(t,d)_i \in \tilde{V}$ has one dimensional real valued quality data ($\hat{D} = \mathbb{R}$). The norm (2.5) over the difference between quality data then indicates an distance between time series points. The *distance of a warp path* w between the time series vectors $(t,d)_i$ and $(t,d)_j$ is the sum of such distances, defined by

$$Dist(w) = \sum_{h=1}^{H} \left\| d_i^{w_h[1]} - d_j^{w_h[2]} \right\|,$$

where $w_h[1]$ and $w_h[2]$ are the first and second element of the h-th tuple of the warp path w respectively. The *optimal warp path* w^* is defined by the warp path with minimal distance.

Definition 5.2.2 (Strict Subsequence Inclusion)
Define the operation $\sqsubseteq$ as the *strict subsequence* on time series $(t,d)_i$ on $(t,d)_j \in \tilde{X}$ of length $P_i \leq P_j$ respectively by,

$$(t,d)_i \sqsubseteq (t,d)_j$$
$$\Longleftrightarrow$$
$$\left(t_i^k, d_i^k\right) = \left(t_j^{k+n}, d_j^{k+n}\right) \qquad \text{for some fixed } n \in 0,\dots,P_j - P_i \quad \forall k \in 1,\dots,P_i.$$

Define the operation $|\sqsubseteq$ as the *headed strict subsequence* on two time series $(t,d)_i$ and $(t,d)_j \in \tilde{X}$ of length P_i and P_j respectively by,

$$(t,d)_i \,|\!\sqsubseteq (t,d)_j$$
$$\Longleftrightarrow$$
$$\left(t_i^k, d_i^k\right) = \left(t_j^k, d_j^k\right) \qquad \forall k \in 1,\dots,P_i.$$

The headed strict subsequence is a strict subsequence of a time series that is equal to the head of that time series. Define $(t,d)_i^r$ as the headed subsequence of time series $(t,d)_i$ of length r,

$$(t,d)_i^r = (t_i^1, d_i^1), (t_i^2, d_i^2), \dots, (t_i^r, d_i^r) \qquad r \leq P_i,$$

such that $(t,d)_i^{P_i} = (t,d)_i$.

Dynamic Time Warping is a dynamic programming approach to find the optimal warp path w^* and its distance. The program has a recursive method which iteratively computes the warping path with minimal distance for smaller headed strict subsequences of the time series $(t,d)_i$ and $(t,d)_j$. Instead of solving the entire problem at once, sub-problems are solved and used to find solutions to larger problem.

In the dynamic programming approach, DTW constructs a 2 dimensional *cost matrix* D in a "bottom-up" way. Its elements $D_{r,s}$ indicate the distance of the optimal warp path for the time series $(t,d)_i^r$ and $(t,d)_j^s$. The "bottom-up" way ensures that the value of D_{P_i,P_j} is the distance of the optimal warp path between the time series $(t,d)_i$ and $(t,d)_j$.

The DTW method starts with the initial setting

$$\begin{aligned} D_{i,0} &= \infty & \forall i &= 1,\dots,P_i, \\ D_{0,j} &= \infty & \forall j &= 1,\dots,P_j, \\ D_{0,0} &= 0. \end{aligned}$$

This initial setting ensures that time warping paths w starts with $w_1 = (1,1)$.

The cost matrix is filled with the iterative function

$$D_{r,s} = \left\| d_i^r - d_j^s \right\| + \min\left[D_{r-1,s}, D_{r,s-1}, D_{r-1,s-1}\right]. \tag{5.2}$$

The values $D_{r-1,s}, D_{r,s-1}$ & $D_{r-1,s-1}$ represent the distance of an optimal warping path to prior headed subsequences, prior with respect to the second and third property of a warping path. By choosing the minimum of those elements the optimal warping path between the time series $(t,d)_i^r$ and $(t,d)_j^s$ is defined. The iteration stops at the final entry D_{P_i,P_j}, giving the distance of the optimal warp path between $(t,d)_i$ and $(t,d)_j$.

The optimal warp path itself can then be computed using the full cost matrix D. The warping path w^* is computed in reverse order starting at D_{P_i,P_j}. A greedy search is performed that evaluates cells near the current cell. If the current cell is $D_{r,s}$ the cells $D_{r-1,s}, D_{r,s-1}$ and $D_{r-1,s-1}$ will be evaluated. The warp tuple connected to the matrix entry with minimal value is added to the warp path, creating the optimal warppath recursively.

In this section the one dimensional application of the DTW distance measure is given. However, other than the Minkowski and Euclidean distance measures, this distance measure can be easily extended to multiple dimensions. The norm used in (5.2) can be used as a multidimensional norm, giving a DTW path for time series with multidimensional quality data.

5.2.1 Application

DTW distance measure is mostly applied to time series that include a lot of measurements, for instance in the stock market time series. These time series are created using measurements at a fixed time interval, such that they also fit the first category of distance measures. However, due to small shiftings in the time axis the Euclidean distance measure gives an unbalanced results. With the use of DTW these time shifting influences are managed.

There is a difference between DTW's standard field of application and our situation. The transactional data is not measured at a fixed time interval and does not occur at a high and steady rate as for instance stock market data. The DTW method also has no time shift limit such that we question how relevant, from a business point of view, large time shiftings will be for the transactional data comparison. Even the results of the existing modifications that manage the time shiftings are questionable.

Next to the business application questions there exist other issues. The distances are to be used for clustering, meaning that all distances between each of the time series in the data set $\tilde{V}$ have to be computed. Each DTW distance measure is computationally expensive meaning that applying the distance measure to large data sets will eventually result in unmanageable computation times.

The existing methods that improve speed of the DTW distance measure rely on improving the measure for long time series. This however does not result in computation time improvements with respect to the large data sets with short time series. Meaning that these speed improvements will have no effect on the ACME data set.

The question remains whether the DTW method solves the initial problem. We search for a distance measure that evaluates customers that show the same dynamic behavior as similar. Therefore growing customers should have smaller distance to other growing customers and vice versa. However the distances computed with DTW depend mainly on the average volume value. Managing the impact of dynamics on the similarity seems impossible. Therefore, the urge to create a more tangible distance measure is high.

CHAPTER 6

Dynamic cdata

A dichotomy of the problems with dynamics in the transaction data exists between the business and mathematical applications. From the business point of view it is uncertain what dynamics are of interest and how they are shown in time series. On the other hand, from the mathematical application side it is not certain what mathematical formulas indicate what dynamics.

The distance measures in Chapter 5 do not surely compare dynamic properties and it is not certain what dynamics are of influence on the distance. To get a better grasp on the influence of dynamics on mathematical formulas we try to create simple functions that should indicate the value of a specific dynamic property.

To this end a single value (i.e. index) is to be computed, indicating the value of a dynamic property in a time series. This is in essence the same as creating a new cdata entry function, $g_3((t,d)_i) : X \mapsto \mathbb{R}$, that indexes a dynamic property. In contrast to the static cdata entry functions in Section 2.3, for the cdata entry function g_3 indexing a dynamic property time dependent information is used, i.e., information connected to the length P_i of a time series $(t,d)_i$ and/or the time elements $t_i^1, t_i^2, \ldots, t_i^{P_i}$ themselves. We call cdata entries that use time dependent information *dynamic cdata entries*. If the cdata contains dynamic cdata entries it is called *dynamic cdata*.

The dynamic cdata entries have several applications. For instance they can be used as a separate distance measure, or as a dimension extension to the static cdata (2.12). The latter creates dynamic cdata, which for instance can be used in clustering. However, for the function $g_3((t,d)_i)$ to fit the clustering methods in Chapter 3 it has to be independent from the cdata entry functions g_1 and g_2. Note that for the functions g_1 and g_2 independence is already assumed.

The remainder of this chapter focuses on the creation of a dynamic cdata entry function that indexes the volume growth of a time series, called *volume growth indexing function*. Note that the notion of growth from a business point of view is uncertain, creating a field of play where all indexing functions that index some sort of volume growth dynamic may be of use. What we do know is that growth is related to differential formulas and that we need to apply these on the volume time series to get an indication of volume growth.

For now, assume that such a dynamic cdata entry function g_3 correctly indexing the volume growth of a volume time series exists. The independence between the values of g_2 and g_3 is trivial, since the volume data entry is independent of the delta data entry. The independence with g_1 however is not trivial, because both cdata entry functions use the original data entry volume as input. When g_3 and g_1 are dependent, for instance a large total volume is associated with a big or small growth, the distances used in clustering will be out of balance. Meaning that one of the original data entries will be over represented, resulting in clusters that are not balanced and therefore less correct.

It is arguable whether or not it is possible to create a volume growth indexing function g_3 that is independent to total volume g_1. A logical reasoning would be that total volume should be independent to growth because two time series that result in the same total volume do not necessarily mean to both be growing the same, see Figure 4.1. Concluding that, at least to some extend, we should be able to find an average volume growth cdata entry function g_3 that indexes volume growth independently to the total volume.

6.1 Volume Growth indexing functions

In this section indexing functions $g_{3,k} : X \mapsto \mathbb{R}$ for $k = 1, \ldots, 4$ are given that index volume growth of time series. The question is whether these indexing functions fit the mathematical model and the ACME data set. The indexing functions are applied to the volume time series of length P

$$(t, v) = (t^1, v^1), (t^2, v^2), \ldots, (t^P, v^P),$$

where the multidimensional data d^k are replaced by one dimensional data v^k indicating the volume at time t^k. Each element of the time series (t, v) indicates a transactional volume and time, totaling into P transactions.

6.1.1 Average local growth over total volume

A function that indexes the average local growth between neighboring elements of the time series (t, v), is given by

$$\begin{aligned} g_{3,1}\left((t,v)\right) &= \sum_{k=1}^{P-1} \frac{v^{k+1} - v^k}{t^{k+1} - t^k} \Big/ \sum_{k=1}^{P} v^k \\ &= \sum_{k=1}^{P-1} \frac{v^{k+1} - v^k}{t^{k+1} - t^k} \Big/ g_1((t,v)). \end{aligned} \tag{6.1}$$

This function $g_{3,1}$ gives an indication of the average volume growth. Since the growth indexing function should be independent on g_1 we normalized the v_i^k in $g_{3,1}$ through the division by g_1.

To find whether the dynamic cdata entry function $g_{3,1}$ fits the mathematical model *correlations* between the cdata entries g_1 and $g_{3,1}$ are tested. In Figure 6.1 the ACME customers are plotted as single dots of cdata $(g_1, g_{3,1})$. This figure shows that large $g_1((t,d)_i)$ often correspond with small $|g_{3,1}((t,d)_i)|$, indicating a correlation between the two cdata entry functions. One of the problems of the correlation between $g_{3,1}$ and g_1 is that growing and decaying customers with large total volume can not be told apart.

To indicate the correlation between total volume g_1 and the indexing function $g_{3,1}$ a set of time series S_1 is created

$$S_1 = \left\{(t,v)_i : (t_i^k, v_i^k) = (\Delta t \cdot k, \Delta v \cdot k + \mu_i), k = 1, \ldots, P\right\}$$

The set S_1 contains time series of equal length P that have volume paths that are parallel to each other (Δt and Δv are constant).

When applied to the set of time series S_1 the indexing function $g_{3,1}|_{S_1} : S_1 \mapsto \mathbb{R}$ can be simplified to

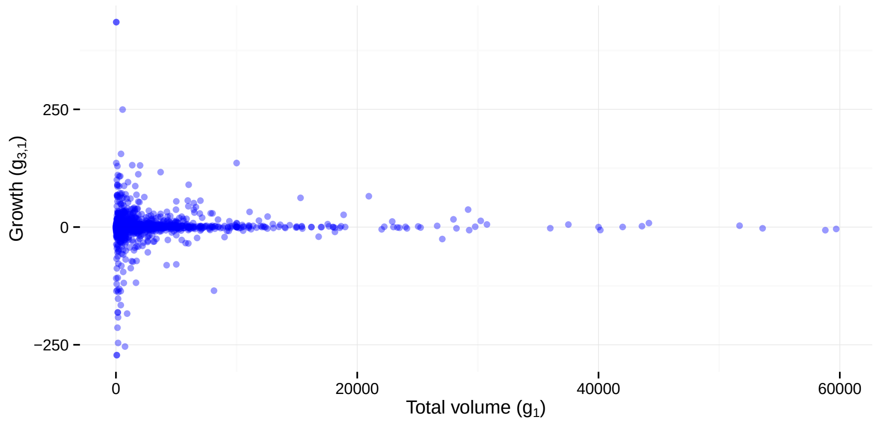
$$\begin{aligned} g_{3,1}|_{S_1}\left((t,v)_i\right) &= P\frac{\Delta v}{\Delta t} \Big/ \frac{1}{2}P(P+1)\Delta v + P \cdot \mu_i \\ &= \frac{\Delta v}{\Delta t \cdot \left(\frac{1}{2}(P+1)\Delta v + \mu_i\right)}. \end{aligned}$$

In the function $g_{3,1}|_{S_1}$ all variables are constant except for the value of μ_i, which therefore is the only value which influences the value of the index. However, since the volume paths are parallel, we would expect that the value μ_i (indicating the parallel shift) would have no influence on the growth index. The total volume g_1 of time series $(t,v)_i \in S_1$ is given by

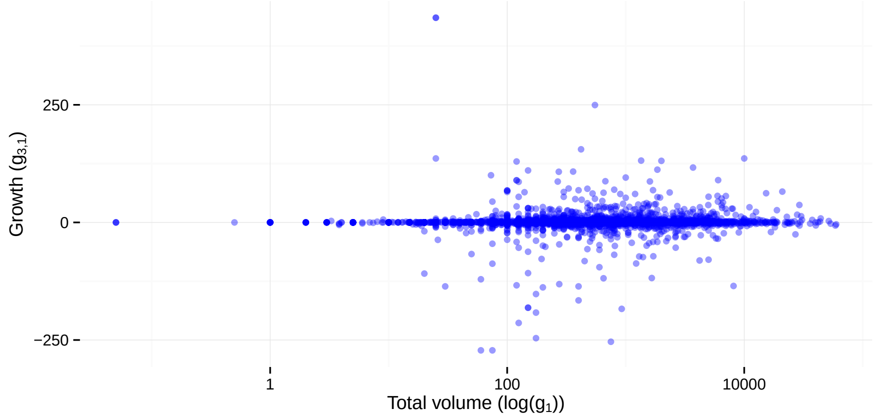
$$g_1|_{S_1}((t,v)_i) = P \cdot \mu_i + \frac{1}{2} P \cdot (P \cdot \Delta v + \Delta v).$$

Note that g_1 depends linearly on μ_i and therefore correlates with $g_{3,1}$.

The volume paths of the 50 times series below the top 50 with respect to function $g_{3,1}$ are plotted in Figure 6.2. The top 50 is omitted because they show even more extreme behavior (oscillating) than in Figure 6.2. Each customer drawn in Figure 6.2 is a customer with relative large positive values for $g_{3,1}$ and therefore should be growing. Figure 6.2 however does not emphatically show that these customers are growing.



(a) linear



(b) logarithmic

Figure 6.1: Linear and logarithmic representation of the distribution of volume growth $g_{3,1}$ versus total volume g_1 for the ACME cdata $(g_1, g_{3,1})$.

6.1.2 Sum of the local growths

In an attempt to lesser the correlation between g_1 and $g_{3,1}$ we removed the division by g_1 in (6.1). This results in a new function indexing the volume growth,

$$g_{3,2}((t,v)) = \sum_{k=1}^{P-1} \frac{v^{k+1} - v^k}{t^{k+1} - t^k}. \tag{6.2}$$

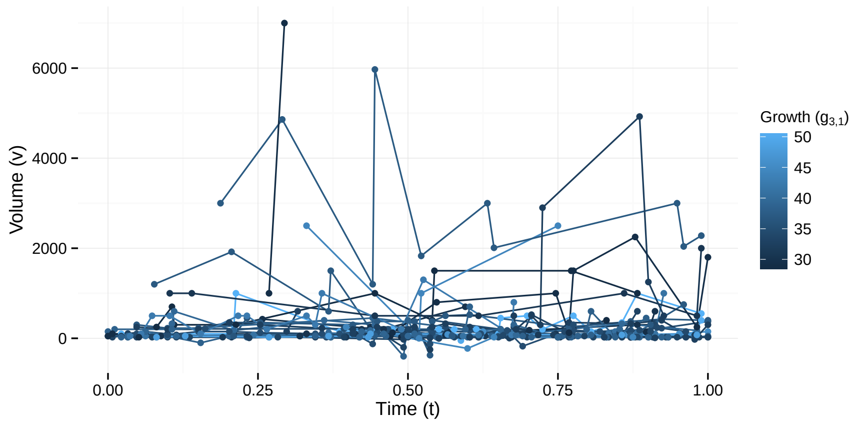


Figure 6.2: Volume paths of the 50 customers below the top 50 customers with respect to the value of $g_{3,1}$ for the ACME cdata $(g_1, g_{3,1})$.

The function $g_{3,2}$ computes the sum of the relative growths. In Figure 6.3 the ACME customers are plotted as single dots of cdata $(g_1, g_{3,2})$. Contrary to the distribution of the dynamic cdata $(g_1, g_{3,1})$ in Figure 6.1, the dynamic cdata $(g_1, g_{3,2})$ in Figure 6.3 are more evenly distributed.

To indicate how $g_{3,2}$ (6.2) behaves with respect to dynamics a new set of time series S_2 is created. S_2 contains time series with constant and equal local growth rate, but the amounts of transactions varies. The new time series set S_2 is indicated by

$$S_2 = \left\{ (t,v)_i : (t_i^k, v_i^k) = \left(\frac{\tau}{P_i - 1} \cdot (k-1), \frac{\nu}{P_i - 1} \cdot (k-1) + c \right), k = 1, \dots, P_i \right\}.$$

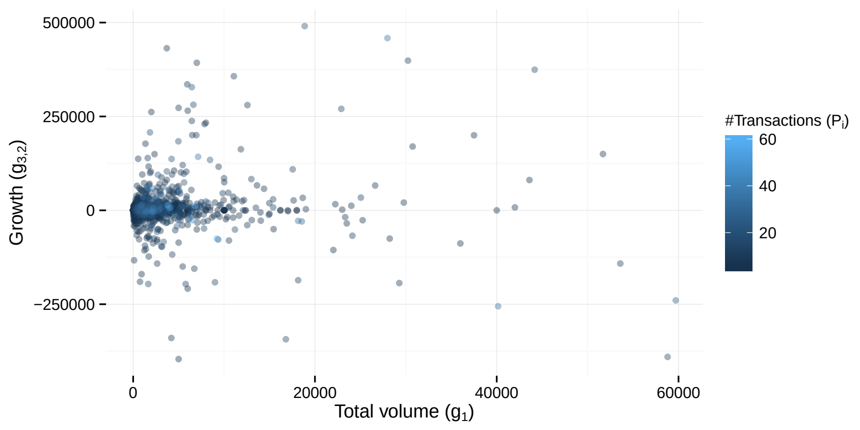
Here τ indicates the full time period, c the start volume and ν the difference in volume between the first and last transaction. Note that the time series $(t,v)_i$ all have the same volume path. The cdata entry function $g_{3,2}$ applied to time series $(t,v)_i \in S_2$ can be simplified to the indexing function $g_{3,2}|_{S_2} : S_2 \mapsto \mathbb{R}$ by

$$\begin{aligned} g_{3,2}|_{S_2}((t,v)_i) &= \sum_{k=1}^{P_i - 1} \frac{\nu}{P_i - 1} \Big/ \frac{\tau}{P_i - 1} \\ &= \sum_{k=1}^{P_i - 1} \frac{\nu}{\tau} \\ &= (P_i - 1)\frac{\nu}{\tau}. \end{aligned}$$

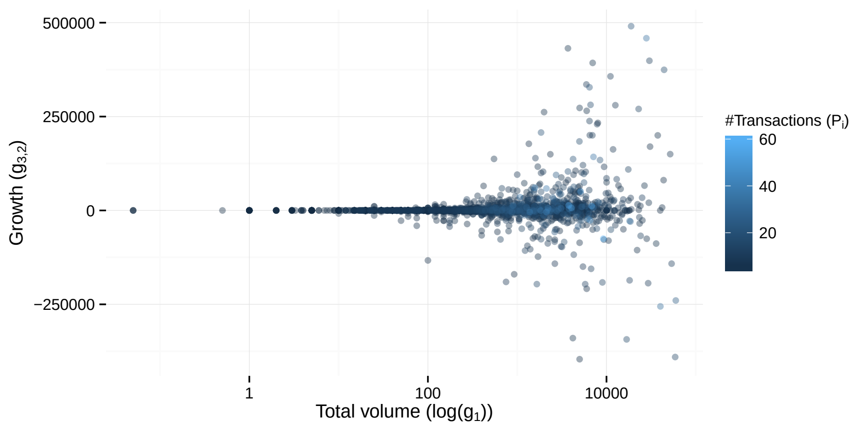
The value of $g_{3,2}|_{S_2}((t,v)_i)$ is linearly dependent to the value P_i, which therefore has a direct influence on the growth indexing function $g_{3,2}$. To visualize this situation two buying patterns are indicated in Figure 4.2. The volume paths of the red and blue time series in Figure 4.2 have the same volume path. However the growth indexed by $g_{3,2}$ is different for the two time series, namely 4 times higher for the red time series (note that $P_{\text{red}} = 5$ and $P_{\text{blue}} = 2$). The cumulative volume paths in the right figure however do show dissimilar behavior for the two customers, such that the growth value of $g_{3,2}$ can be related to the growth of the cumulative volume paths.

The cdata entry function g_1 applied to time series $(t,v)_i \in S_2$ can also be simplified,

$$\begin{aligned} g_1|_{S_2}((t,v)_i) &= \sum_{k=1}^{P_i} \frac{\nu}{P_i - 1} \cdot (k-1) + c \\ &= P_i \cdot \left(c + \frac{\nu}{2} \right). \end{aligned}$$



(a) linear



(b) logarithmic

Figure 6.3: Linear and logarithmic representation of the distribution of volume growth $g_{3,2}$ versus total volume g_1 for the ACME cdata $(g_1, g_{3,2})$.

In Figure 6.4 the position $(g_1, g_{3,2})$ is plotted as a dot for time series $(t, v)_i \in S_2$ created from all combinations of the variables

$$\nu = \{100, 200, \dots, 500\},$$
$$\tau = \{0.2, 0.4, \dots, 1\},$$
$$P_i = \{1, 2, \dots, 10\}.$$

The lines in this figure indicate a group of time series with equal ν and τ, such that within a line only the variable P_i varies. The outer left and right dots on these lines indicate position $(g_1, g_{3,2})$ of time series with $P_i = 1$ and $P_i = 10$ respectively. The value P_i increases by one for each dot when following the line from left to right. In Figure 6.4 we see the following relations between variables ν, τ & P_i and the position $(g_1, g_{3,2})$ of time series $(t, v)_i \in S_2$:

- When increasing the value ν both growth $g_{3,2}$ and volume g_1 increase
- When decreasing the value τ growth $g_{3,2}$ will increase and volume g_1 will remain equal
- When increasing the value P_i both growth $g_{3,2}$ and volume g_1 increase.

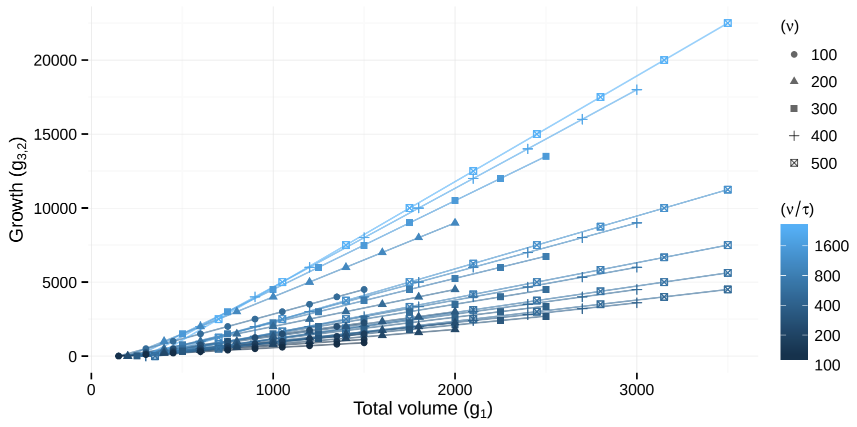


Figure 6.4: The cdata $(g_1, g_{3,2})$ for time series in S_2 grouped via a line for equal ν and τ.

The volume paths of the 50 times series below the top 50 with respect to growth $g_{3,2}$ are plotted in Figure 6.5. The top 50 is again omitted to exclude even more extremes. In contrast to Figure 6.2, Figure 6.5 shows volume paths with a much larger volume g_1. However the volume paths in Figure 6.5 do not emphatically show that these customers are growing.

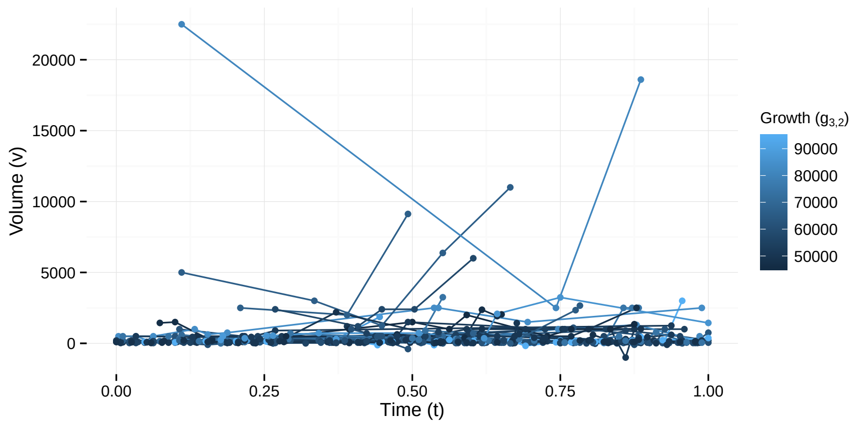


Figure 6.5: Volume paths of the 50 customers below the top 50 customers with respect to the value of $g_{3,2}$ for the ACME cdata $(g_1, g_{3,2})$.

6.1.3 Local volume growth per transaction

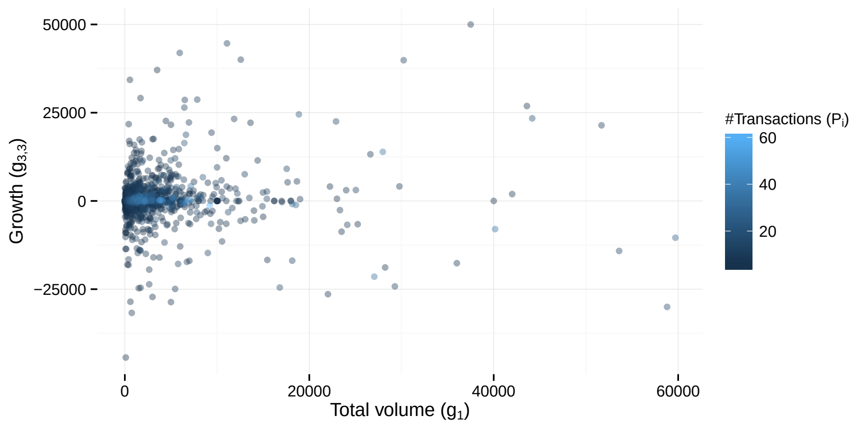
Another growth index is the average local growth per transaction, the formula is given by

$$g_{3,3}\left((t,v)\right) = \sum_{k=1}^{P-1} \frac{v^{k+1} - v^k}{t^{k+1} - t^k} \bigg/ P. \tag{6.3}$$

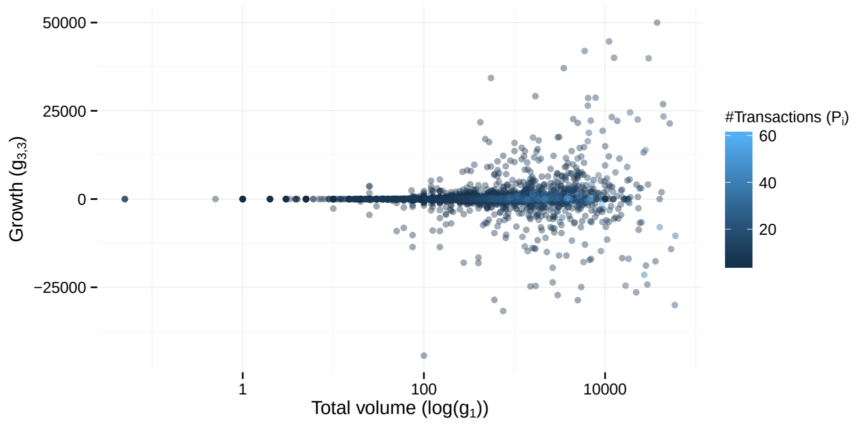
$g_{3,3}$ computes the average growth step between subsequent transactions, which gives an indication of partial growths. In Figure 6.6 the ACME customers are plotted as single dots of cdata $(g_1, g_{3,3})$. Contrary to the distribution of the dynamic cdata $(g_1, g_{3,1})$ in Figure 6.1, the dynamic cdata $(g_1, g_{3,3})$ in Figure 6.6 are more evenly distributed.

The color scale in Figure 6.6 indicates the number of transactions P. For a relative high P the value of $g_{3,3}$ tends to zero. For a better comparison the ACME customers are plotted as single dots of cdata $(P_i, g_{3,3})$ in Figure 6.7. Here the distribution of dynamic cdata is similar to the distribution in Figure 6.1, indicating a dependency between the transaction count P_i and the growth indexing function $g_{3.3}$.

To see whether or not customers that purchase products more often are stable with respect to growth, the volume paths of the customers with transaction count $P_i > 100$ are drawn in Figure 6.8. This figure shows oscillating volume paths and therefore does not support nor contradicts the earlier claim of dependency between P_i and $g_{3,3}$.



(a) linear



(b) logarithmic

Figure 6.6: Linear and logarithmic representation of the distribution of volume growth $g_{3,3}$ versus total volume g_1 for the ACME cdata $(g_1, g_{3,3})$.

The volume paths of the 50 times series below the top 50 with respect to volume growth $g_{3,3}$ are plotted in Figure 6.9. Again the top 50 is omitted because it usually contains extremes that do not give a relevant insight in results of $g_{3,3}$. In contrast to Figure 6.2, Figure 6.9 as 6.5 show volume paths with a higher volume g_1. However (again) the volume paths shown do not emphatically show that these customers are growing.

Figures 6.2, 6.5 and 6.9 all do not show emphatically growing customers. The volume paths shown in these figures are instead all highly irregular. The irregularities may be the cause of extreme growth indexes since all of the indexing functions $g_{3,k}$, $k = 1, 2, 3$, contain the factors $\frac{1}{t^{k+1} - t^k}$.

The irregularities in time and volume somehow seem to have a dominant effect on the previous

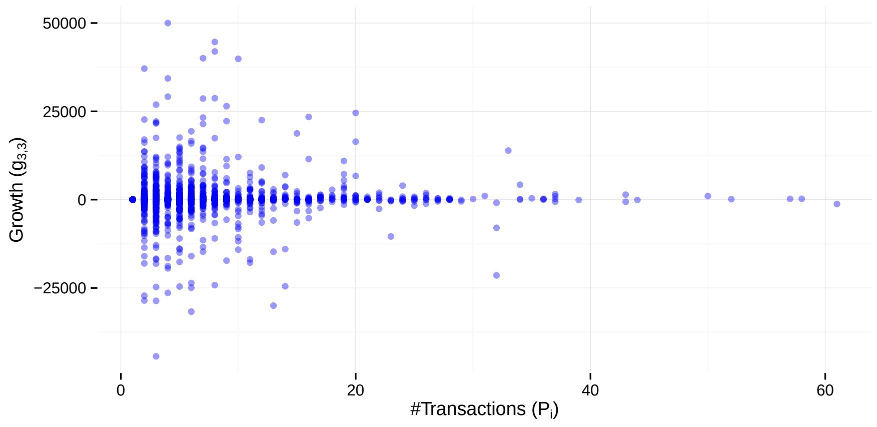


Figure 6.7: Growth computed via the indexing function $g_{3,3}$ over transaction count for the ACME data.

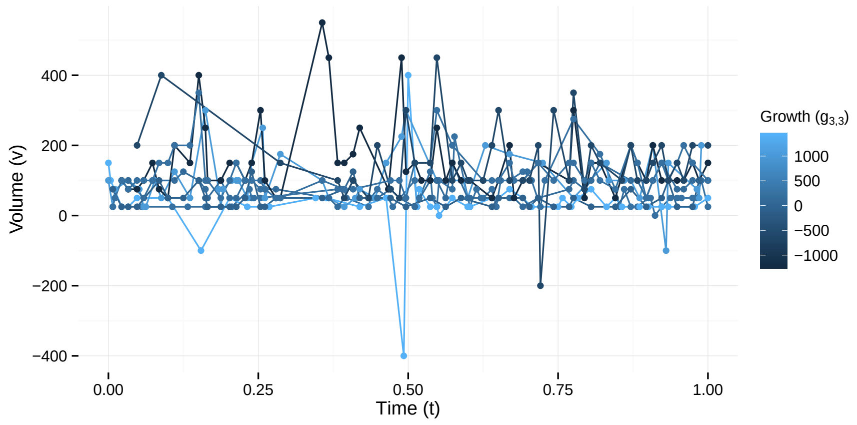


Figure 6.8: Volume paths for the time series from the ACME data set with more than 40 transactions, with as color the growth index computed with function $g_{3,3}$.

indexing functions $g_{3,1}, g_{3,2}$ and $g_{3,3}$. To better comprehend the impact of time irregularities on these indexing functions a new set of time series S_3 is created. Let time series $(t, v)_i \in S_3$ be of same odd length P and have equal internal volumes $v^1, \ldots, v^P$. Let the start t^1 and end t^P of these time series be constant such that S_3 is defined by

$$S_3 = \Big\{(t, v)_i : \qquad (t_i^{2k-1}, v^{2k+1}) = (k \cdot \tau, \nu_1), \qquad k = 1, \ldots, \frac{P+1}{2},$$
$$(t_i^{2k}, v^{2k}) = ((k + \alpha) \cdot \tau, \nu_2), \qquad k = 1, \ldots, \frac{P-1}{2}\Big\},$$

where τ is the time period in which exactly two transactions occur, ν_1 and ν_2 are two different volumes with $\nu_1 << \nu_2$ and $0 < \alpha < 1$ is a shifting factor. The volume and cumulative volume paths of two example time series from S_3 are plotted in Figure 6.11.

The indexing function $g_{3,3}$ applied to time series $(t, v)_i \in S_3$ can be simplified to the indexing

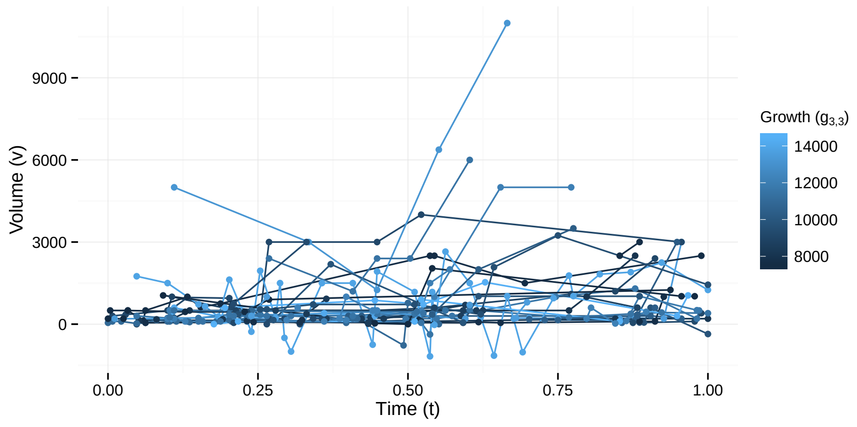


Figure 6.9: Volume paths of the 50 customers below the top 50 customers with respect to the value of $g_{3,3}$ for the ACME cdata $(g_1, g_{3,3})$.

function $g_{3,3}|_{S_3} : S_3 \mapsto \mathbb{R}$ defined by

$$\begin{aligned} g_{3,3}|_{S_3}\left((t,v)_i\right) &= \left(\sum_{k=1}^{(P-1)/2} \frac{\nu_2 - \nu_1}{\alpha\tau} + \sum_{k=1}^{(P-1)/2} \frac{\nu_1 - \nu_2}{(1-\alpha)\tau}\right) \Big/ P \\ &= \frac{(\nu_2 - \nu_1)(P-1)}{2P\tau} \underbrace{\left(\frac{1}{\alpha} - \frac{1}{(1-\alpha)}\right)}_{h(\alpha)}, \end{aligned}$$

where the function

$$h(\alpha) = \frac{1-2\alpha}{\alpha(1-\alpha)}, \tag{6.4}$$

assuming $\tau > 0$ and $0 < v_1 << v_2$ are fixed. The only free variable is α such that

$$\begin{aligned} g_{3,3}|_{S_3}\left((t,v)_i\right) &= 0 && \text{for } \alpha = 0.5, \\ g_{3,3}|_{S_3}\left((t,v)_i\right) &< 0 && \text{for } \alpha > 0.5, \\ g_{3,3}|_{S_3}\left((t,v)_i\right) &> 0 && \text{for } \alpha < 0.5. \end{aligned}$$

The function h describes the influence of α on the index $g_{3,3}((t,v)_i)$ for time series $(t,v)_i \in S_3$. The graph h with $\tau = 1$ is plotted in Figure 6.10. This figure shows that the function h (6.4) is not suited if the data set contains time series (t,v) with small $t^{k+1} - t^k$.

The two time series in Figure 6.11 show similar paths, especially when looking at the cumulative volume path. However, the index computed with either $g_{3,1}, g_{3,2}$ or $g_{3,3}$ differs a lot for the two time series.

The Figures 6.2, 6.5 and 6.9 show a lot of oscillating volume paths. Showing that, one, the ACME data set contains highly oscillating volume paths and, two, that the growth indexing functions $g_{3,k}$, $k = 1, 2, 3$ are sensitive with respect to irregularities in data sets.

6.1.4 Average volume difference between transactions

An indexing function that does not consider the time intervals $t^{k+1} - t^k$ is given by

$$g_{3,4}\left((t,v)\right) = \frac{\sum_{k=1}^{P-1} v^{k+1} - v^k}{P} = \frac{v^P - v^1}{P}. \tag{6.5}$$

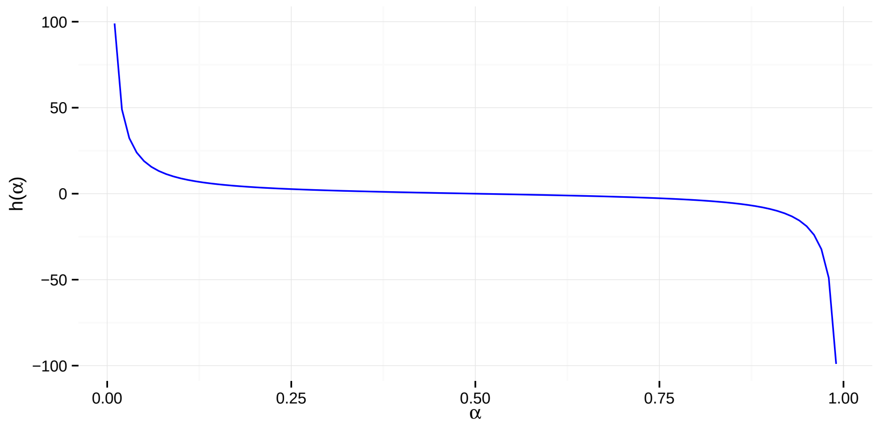
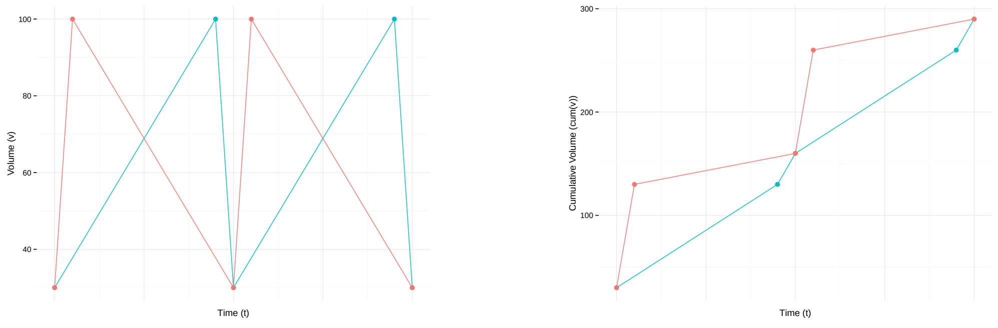


Figure 6.10: α impact on the growth index, indicated via the trajectory of h (6.4).



(a) Volume

(b) Cumulative Volume

Figure 6.11: Volume and Cumulative Volume over time for time series that give large differences in growth computed with either (6.1), (6.2) and (6.3).

This however is a quick and dirty indexing function that ignores all internal transactions and only depends on the first and last volume v^P and v^1. Therefore it does not provide reliable information about internal dynamics.

6.2 Volume Growth indexing functions using smoothing

The inclusion of individual time intervals $t^{k+1} - t^k$ makes indexing functions $g_{3,1}$, $g_{3,2}$ and $g_{3,3}$ unusable with highly irregular data sets. The exclusion on the other hand, as in the indexing function $g_{3,4}$, results in a function which does not provide reliable dynamics information. In this section we construct an indexing function that provides dynamics information but does not use the time intervals $t^{k+1} - t^k$ directly.

The indexing function analyzed divides the time frame $[t^1, t^P]$ of time series (t, v) into two periods of equal length. The periods are defined by the divider

$$t_{\text{mid}} = \frac{t^1 + t^P}{2},$$

which is the middle of the time frame. Let r be defined such that

$$t^r < t_{\text{mid}} \leq t^{r+1}.$$

We define a *smooth representation* of the time series (t, v) by a new time series containing two volumes v^{left} and v^{right} as the left and right average volumes of the original time series respectively

$$\begin{aligned} v^{\text{left}} &= \frac{\sum_{k=1}^{r} v^k}{r} \Big/ \frac{\sum_{k=1}^{P} v^k}{P}, \\ v^{\text{right}} &= \frac{\sum_{k=r+1}^{P} v^k}{P-r} \Big/ \frac{\sum_{k=1}^{P} v^k}{P}. \end{aligned} \tag{6.6}$$

To create a time series $(t^{\text{left}}, v^{\text{left}}), (t^{\text{right}}, v^{\text{right}})$ these volumes are paired with timestamps

$$\begin{aligned} t^{\text{left}} &= \frac{t^1 + t_{\text{mid}}}{2} = \frac{3t^1 + t^P}{4} \\ t^{\text{right}} &= \frac{t_{\text{mid}} + t^P}{2} = \frac{t^1 + 3t^P}{4} \end{aligned} \tag{6.7}$$

Define $\hat{g}_3$ applied to the smooth representation of time series (t, v) as

$$\hat{g}_3 \left((t^{\text{left}}, v^{\text{left}}), (t^{\text{right}}, v^{\text{right}}) \right) = v^{\text{right}} - v^{\text{left}}. \tag{6.8}$$

An indexing function $g_{3,5}$ that is defined on original time series (t, v) is thus given by

$$g_{3,5}((t,v)) = \hat{g}_3 \left((t^{\text{left}}, v^{\text{left}}), (t^{\text{right}}, v^{\text{right}}) \right) = \left(\frac{1}{P-r} \sum_{k=r+1}^{P} v^k - \frac{1}{r} \sum_{k=1}^{r} v^k \right) \Big/ \frac{1}{P} \sum_{k=1}^{P} v^k. \tag{6.9}$$

This function computes the relative growth of the average volume over the two time periods defined by t_{mid}.

In Figure 6.12 the ACME customers are plotted as single dots of cdata $(g_1, g_{3,5})$. Contrary to the distribution of the cdata $(g_1, g_{3,1})$ in Figure 6.1, the cdata $(g_1, g_{3,5})$ in Figure 6.12 is more evenly distributed.

In Figure 6.13 the volume paths of the 50 times series below the top 50 with respect to growth $g_{3,5}$ are plotted. The top 50 is omitted because it usually contains extremes that do not give a relevant insight in results of $g_{3,5}$. Figure 6.13 in contrast to Figures 6.2, 6.5 and 6.9, shows that $g_{3,5}$ attains similar positive values for growing v^k. For decreasing v^k also similar negative values are attended.

Although the better result as indicated by Figure 6.13 there are obvious issues involved with the calculation of $g_{3,5}$ using the proposed smoothing.

- What to do with time series (t, v) of the form

$$(t, v) = (t^1, v^1), (t^2, v^2), (t^3, v^3),$$

  with $t^2 = \frac{1}{2}(t^1 + t^3)$ and $v^2 >> v^1 + v^3$.

- The normalization caused by the division by the total volume g_1 in the smoothed representation volumes (6.6) could result in a correlation between the values $g_{3,5}$ and g_1.

Indexing function $g_{3,5}$ is one of the simplest cases in a family of indexing functions that use smoothing. Modifications applied to the function $g_{3,5}$ to obtain other indexing functions are

- Using the sum of the volumes v^k rather than the average volumes for the smoothed representation volumes

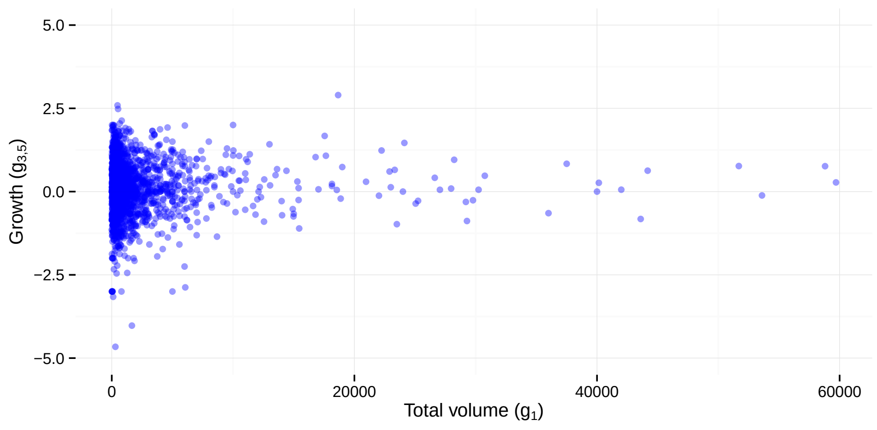
$$\begin{aligned} \tilde{v}^{\text{left}} &= \frac{\sum_{k=1}^{r} v^k}{\sum_{k=1}^{P} v^k}, \\ \tilde{v}^{\text{right}} &= \frac{\sum_{k=r+1}^{P} v^k}{\sum_{k=1}^{P} v^k}. \end{aligned}$$

In combination with timestamps t^{left} and t^{right} as in (6.7) these two volumes create a new time series. Using $\hat{g}_3$ (6.8) it also creates a new indexing function $g_{3,6}$ that is defined on the original time series (t, v) as

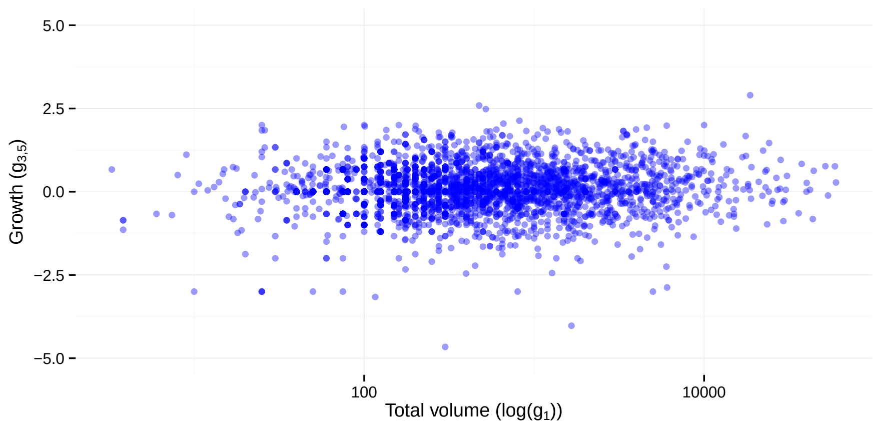
$$g_{3,6}((t,v)) = \hat{g}_3\left((t^{\text{left}}, \tilde{v}^{\text{left}}), (t^{\text{right}}, \tilde{v}^{\text{right}})\right) = \left(\sum_{k=r+1}^{P} v^k - \sum_{k=1}^{r} v^k\right) \Big/ \sum_{k=1}^{P} v^k. \qquad (6.10)$$

- Averaging by the total time intervals $t^P - t^1$ such that the value is relative to the time period in which the growths occur.
- More intervals (not only "left" and "right") such that more dynamics may be considered.
- Using the difference in number of transactions over the periods as a frequency indication.
- Using the values $t^{k+1} - t^k$ as an irregularity indication.

The exact effects of these modifications to the growth index for time series $(t, d)_i$ in the ACME data set are unknown. Given the amount of time allotted to the writing of this thesis these other indexing functions have not been examined in detail.



(a) linear



(b) logarithmic

Figure 6.12: Linear and logarithmic representation of the distribution of volume growth $g_{3,5}$ versus total volume g_1 for the ACME cdata $(g_1, g_{3,5})$.

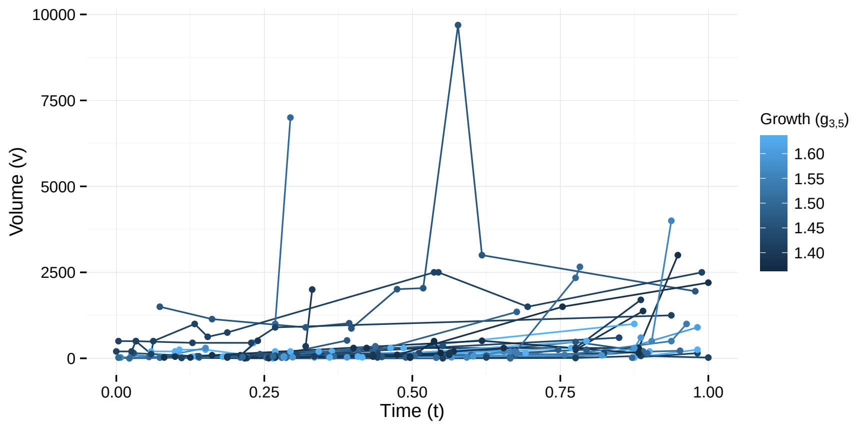


Figure 6.13: Volume paths of the 50 customers below the top 50 customers with respect to the value of $g_{3,5}$ for the ACME cdata $(g_1, g_{3,5})$.

6.2.1 Smoothing data

The main difference between indexing function $g_{3,5}$ and the prior indexing functions is that $g_{3,5}$ smoothed the time domain for the calculation of the growth index. The latter show that using the separate time intervals $t_i^{k+1} - t_i^k$ results in an indexing function that is sensitive to irregularities and unusable on highly irregular data sets. On the other hand via the creation a cdata set where high volume growth index corresponds to larger growth (in general) through smoothing, a lot of dynamics (mostly all dynamics but one) are ignored. It is therefore useful to investigate the smoothing effects on dynamics and function values.

The smoothing of time series can be done by dividing the points inside time series into time periods, called *time smoothing periods*. Over these time smoothing periods cdata is computed that is afterwards used to create the smoothed representation of the time series. There exist a lot of choices in creating the time smoothing periods and the cdata, where different combinations may lead to completely different representations. Note that the original static cdata as given in Section 2.3 is a rough form of smoothing, so rough that that the research in this thesis was motivated by its shortcomings.

In Examples 22 and 23 two families of time smoothing periods are given that both create time periods that are of equal length. Next to these time smoothing periods a lot more periods are possible. The best or most relevant of these time smoothing periods however remains unknown, especially when considering that we do not know the effect of these time smoothing periods on dynamics.

Example 22 (Total time smoothing periods)
The total time period indicated by delimiters dat_1, dat_2 as in Chapter 2 can be divided into T time periods of equal length $\Delta t = \dfrac{dat_2 - dat_1}{T}$. This creates time period delimiters for period $s = 1, \ldots, T$ as

$$\begin{aligned} dat_1^s &= dat_1 + (s-1)\Delta t, \\ dat_2^s &= dat_1 + (s)\Delta t. \end{aligned}$$

The points in time series $(t, d)_i$ that reside in the same time period s

$$(t_i^k, d_i^k) \text{ such that } dat_1^s < t_i^k < dat_2^s$$

are then to be accumulated using cdata functions to create a smoothed representation of the original time series.

Example 23 (Time series smoothing periods)
Time periods can also be created with respect to the values t_i^1 and $t_i^{P_i}$ of the time series $(t, d)_i$, as for indexing function $g_{3,5}$. This makes the time smoothing periods time series specific. The time period $[t_i^1, t_i^{P_i}]$ could for instance be divided into T time periods of equal length $\Delta t = \dfrac{t_i^{P_i} - t_i^1}{T}$. This creates time period delimiters for period $s = 1, \ldots, T$ as

$$\begin{aligned} dat_1^s &= t_i^1 + (s-1)\Delta t, \\ dat_2^s &= t_i^1 + (s)\Delta t. \end{aligned}$$

The points in time series $(t, d)_i$ are then to be accumulated using cdata functions in the same manner as in Example 22.

Another question is what cdata is relevant to represent the data in a smoothing period. The chosen cdata has to be relevant for the analyses of dynamics. Whether sums or averages of original data entries should be used in the smoothed representation is unclear.

Finally note that whether or not a smoothing works is closely related to the application (the considered data set and business target) that it is used for.

CHAPTER 7

Clustering using the dynamic cdata entry function $g_{3,5}$

The volume growth indexed with indexing function $g_{3,5}$ gives an impression of global volume growths through time. The next question is whether it is usable with the clustering methods given in Chapter 3. For the application of the clustering methods we use the dynamic cdata points $\bar{x}_i$ representing time series $(t,d)_i$ as

$$\bar{x}_i = \begin{pmatrix} g_1((t,d)_i) \\ g_{3,5}((t,d)_i) \end{pmatrix}$$

We define $\bar{V} = \{\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n\}$ as the dynamic cdata set, containing points $\bar{x}_i$ each related to the unique $c_i \in C$.

Example 24 (Contoso)
For the Contoso dynamic cdata set we cluster using the Mean Shift and LPC clustering methods, see Chapter 3. We apply these clustering methods on the two dimensional space $\bar{V}$ defined by the dynamic cdata entry functions volume growth $g_{3,5}$ and total volume g_1. This to see the effects of divisions between cdata entries total volume and volume growth on the clustering results. The points from the dynamic cdata set are plotted in Figure 7.1.

We omitted on purpose the examination of dynamic cdata

$$\begin{pmatrix} g_1((t,d)_i) \\ g_2((t,d)_i) \\ g_{3,5}((t,d)_i) \end{pmatrix}.$$

This because we are interested in the effects of combining total volume g_1 and volume growth $g_{3,5}$ in clustering methods. Since considering g_2 could blur these effects, we are omitting this dimension that the cdata entry function g_2 creates.

In Figure 7.2 the cluster centers resulting from a Mean Shift method application on $\bar{V}$ are given. The clusters seem to be mainly caused by total volume division because the volume growth indexes are centered around zero, causing most modes to lie on values where $g_{3,5} \approx 0$. One cluster is created in the top left of Figure 7.2 containing only one customer because its position is isolated from the other dots.

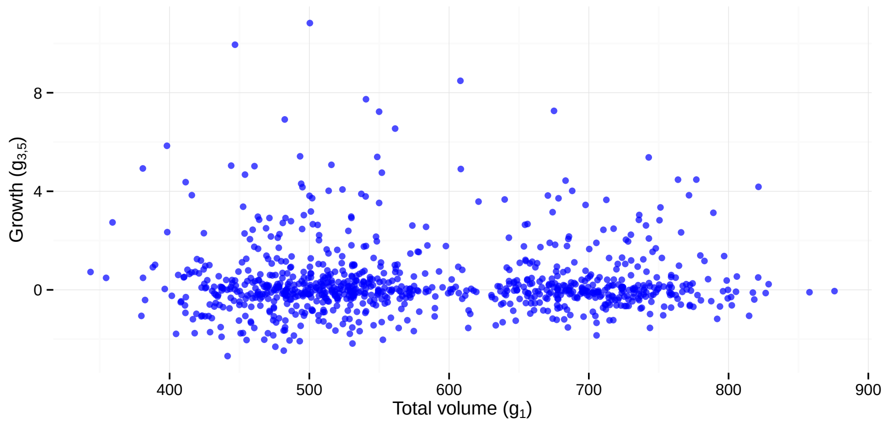


Figure 7.1: Distribution of volume growth $g_{3,5}$ versus total volume g_1 for the Contoso dynamic cdata $(g_1, g_{3,5})$.

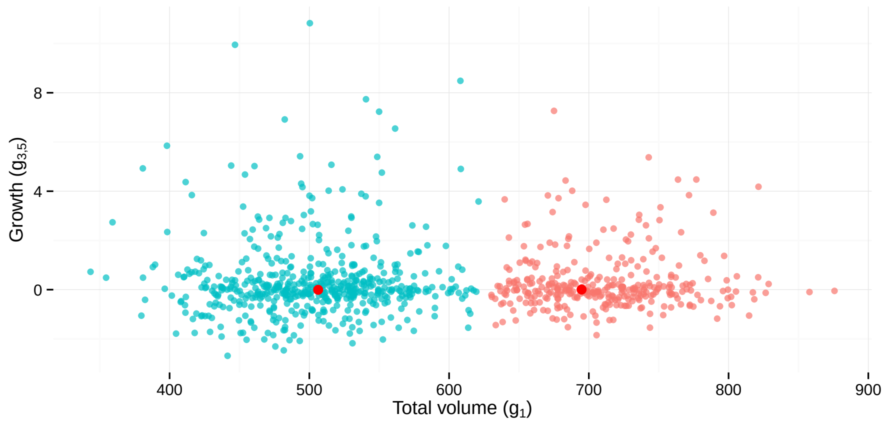


Figure 7.2: Mean Shift method clusters based on the Contoso dynamic cdata set $\bar{V}$ (Each cluster differently colored, centers indicated).

In Figure 7.3 the Local Principal Curve through the Contoso dynamic cdata set is plotted. It is seen that the local principal curve is almost perpendicular to the volume growth direction such that the curve does not describe the division of total volume with respect to volume growth but simply the division of total volume.

Figure 7.4 gives the division of Contoso points into clusters, where the number of clusters k is such that the coverage increase of the k-spline approaches the maximum coverage increase. Note that the clusters are mainly motivated by the value of g_1.

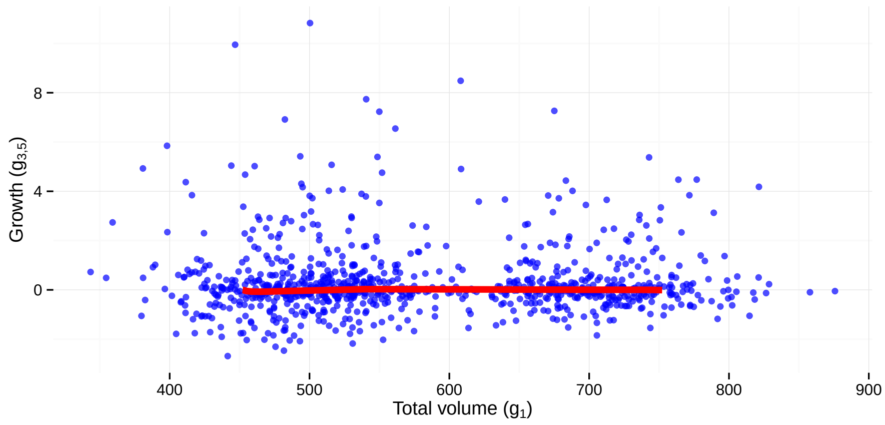


Figure 7.3: Local principal curve through Contoso dynamic cdata set $\bar{V}$.

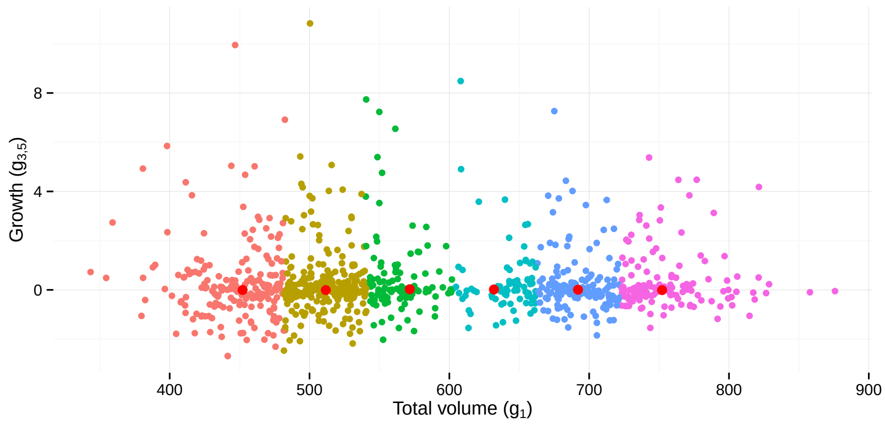


Figure 7.4: Clusters on the Contoso dynamic cdata set $\bar{V}$ computed with the LPC method.

Example 25 (ACME)
For the ACME dynamic cdata set we also cluster using the Mean Shift and LPC clustering methods, see Chapter 3. We apply these clustering methods on the two dimensional space $\bar{V}$ defined by the cdata entries volume growth $g_{3,5}$ and total volume g_1. The ACME dynamic cdata $\bar{V}$ is plotted in Figure 6.12.

In Figure 7.5 the cluster center resulting from a Mean Shift method application to $\bar{V}$ is given. There is only one cluster, because the majority of the customers have relatively low total volumes and low volume growth index. This ensures that the two dimensional density function only contains one density peak resulting in only one mode and thus one cluster.

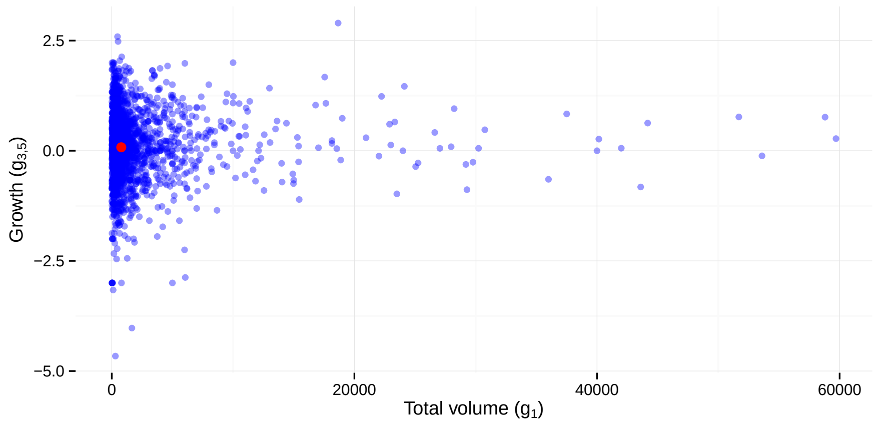


Figure 7.5: Mean Shift method clusters based on the ACME cdata set $\bar{V}$ (The cluster center is indicated).

In Figure 7.6 the Local Principal Curve through the ACME dynamic cdata set is plotted. Contrary to the line in Figure 7.3 this line is almost perpendicular to the total volume axis.

Figure 7.7 divides the points from the ACME dynamic cdata set into clusters. The number of clusters k in this figure is such that the coverage increase of the k-spline approaches the maximum coverage increase. Again we see that the resulting clusters mainly depend on one variable, in this case the value of $g_{3,5}$. The question is whether this is caused by the clustering method, the cdata set, the volume growth index or simply by the fact that we try to cluster on a two-dimensional space with total volume and volume growth as the dimensions.

The question remains whether there exists an indexing function (or another measure on time series) which allows for the clusters to be more dispersed. This however is a topic for future research.

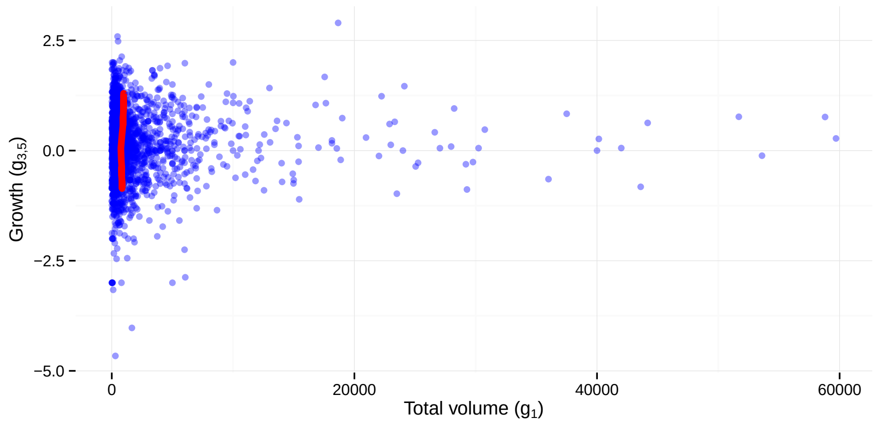


Figure 7.6: Local principal curve through ACME dynamic cdata set $\bar{V}$.

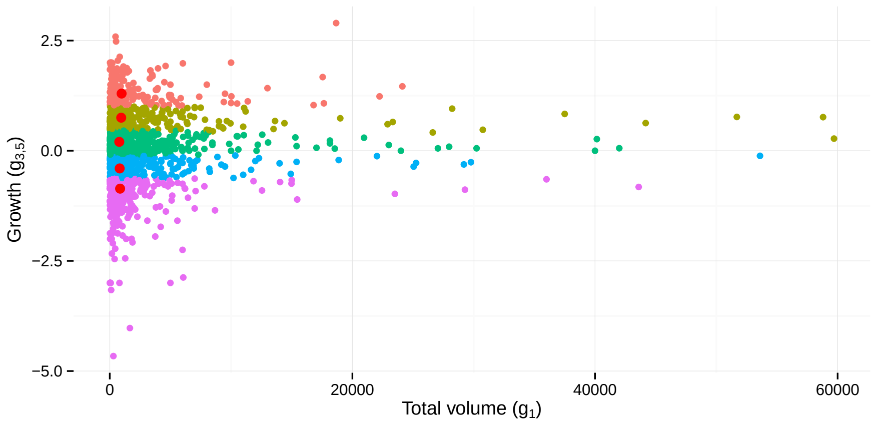


Figure 7.7: Clusters on the ACME dynamic cdata set $\bar{V}$ computed with the LPC method.

Future Research

There are many questions left to be addressed by future research:

Dynamics

- What dynamics are to be taken into account by the distance measures?
- What dynamic properties can we define that are to be measured separately from other variables?
- Is it mathematically possible to define a dynamic property as an independent variable?
- What mathematical dynamics are linked to business related behaviors?

Clustering

- How to obtain "better" clusters, improving the dynamic indexing function?
- Are the presented clustering techniques fitted for dense data clouds?
- How does clustering behave with respect to logarithmic scaling?

Distance measures

- What possible advantages and disadvantages exist for indexing dynamic properties separately?
- Are there any other distance measures that consider dynamics for which the influence of dynamics on similarity is controllable?
- With smoothing, can we correctly and efficiently use the presented distance measures?

Smoothing

- How does smoothing relate the the dynamics?
- How to use multi-period smoothing?
- Are there any other smoothing techniques that consider dynamics?
- Does there exist a method to smooth the highly irregular data sets and keep desired dynamics?

Bibliography

Comaniciu, D. and P. Meer
2002. Mean shift: A robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):603–619. 15, 17

Cuevas, A., M. Febrero, and R. Fraiman
2001. Cluster analysis: a further approach based on density estimation. *Computational statistics & data analysis*, 36(4):441–459.

Delicado, P.
2001. Another look at principal curves and surfaces. *Journal of Multivariate Analysis*, 77:84–116. 19, 21, 22

Einbeck, J. and L. Evers
2013. Local principal curve methods. `http://cran.r-project.org/web/packages/LPCM/LPCM.pdf`. 24

Einbeck, J., G. Tutz, and L. Evers
2005. Local principal curves. *Statistics and Computing*, 15(4):301–313. 18, 19, 22, 24

Fraley, C. and A. Raftery
2002. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, 97(458):611–631.

Fukunga, K. and L. Hostetler
1975. The estimation of the gradient of a density functions, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21(1):32–40.

Keogh, E. and J. Lin
2005. Clustering of time-series subsequences is meaningless: Implications for previous and future research. *Knowl. Inf. Syst.*, 8(2):154–177.

Keogh, E. J. and M. J. Pazzani
2000. Scaling up dynamic time warping for datamining applications. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, Pp. 285–289. ACM.

Keogh, E. J. and M. J. Pazzani
2001. Derivative dynamic time warping. In *SDM*, volume 1, Pp. 5–7. SIAM.

Langone, R., C. Alzate, and J. A. Suykens
2013. Kernel spectral clustering with memory effect. *Physica A: Statistical Mechanics and its Applications*, 392(10):2588–2606.

Liao, T. W.
2005. Clustering of time series data-a survey. *Pattern recognition*, 38(11):1857–1874. 35, 36

Ratanamahatana, C. A. and E. Keogh
2004. Everything you know about dynamic time warping is wrong. In *Third Workshop on Mining Temporal and Sequential Data*. Citeseer.

Silverman, B. W.
1986. *Density estimation for statistics and data analysis*, volume 26. CRC press. 15

Tran, T., R. Wehrens, and L. Buydens
2006. KNN-kernel density-based clustering for high-dimensional multivariate data. *Computational Statistics & Data Analysis*, 51(2):513–525.

Worton, B.
2008. Kernel methods for estimating the utilization distribution in home-range studies.

Implementation

The algorithms in this MSc thesis are implemented in R. For these implementations we used several R-packages. These packages contain methods that handle:

- The database connection (`RMySQL`)
- The data frame manipulation (`dplyr`, `tidyr`)
- The Mean Shift method and LPC method (`LPCM`)
- Other statistical operations (`MASS`, `SDMTools`)
- The plots (`ggplot2`, `plot3D`, `ggthemes`, `lattice`, `grid`).

The functions that compute Mean Shift clusters and local principal curves are provided within the package `LPCM`. The clustering technique given for local principal curves uses functions (coverage and spline) that are also provided in this package. An example application of the Mean Shift as well as the local principal curve method is provided as an example code chunk.

Using `dplyr` and `tidyr` we created short functions that manipulate the data in order to compute the values of $g_{3,k}$ for $k = 1, \dots, 6$. These functions are provided as separate files.

This thesis itself is created in R Sweave (using `knitr`) such that all calculations are done in situ while rendering the document. The raw version of this thesis, and thus all its calculations, is available. The ACME dataset, used by the in situ R calculations is not an available product.

Index