Journal of Machine Learning Research 10 (2009) 2667-2714 Submitted 2/07; Revised 3/09; Published 11/09

Structure Spaces

Brijnesh J. Jain JBJ@CS.TU-BERLIN.DE
Klaus Obermayer OBY@CS.TU-BERLIN.DE
Department of Electrical Engineering and Computer Science
Berlin University of Technology
Berlin, Germany

Editor: Tommi Jaakkola

Abstract

Finite structures such as point patterns, strings, trees, and graphs occur as ”natural” representations of structured data in different application areas of machine learning. We develop the theory of *structure spaces* and derive geometrical and analytical concepts such as the angle between structures and the derivative of functions on structures. In particular, we show that the gradient of a differentiable structural function is a well-defined structure pointing in the direction of steepest ascent. Exploiting the properties of structure spaces, it will turn out that a number of problems in structural pattern recognition such as central clustering or learning in structured output spaces can be formulated as optimization problems with cost functions that are locally Lipschitz. Hence, methods from nonsmooth analysis are applicable to optimize those cost functions.

Keywords: graphs, graph matching, learning in structured domains, nonsmooth optimization

1. Introduction

In pattern recognition and machine learning, it is common practice to represent data by feature vectors living in a Banach space, because this space provides powerful analytical techniques for data analysis, which are usually not available for other representations. A standard technique to solve a learning problem in a Banach space is to set up a smooth error function, which is then minimized by using local gradient information.

But often, the data we want to learn about have no natural representation as feature vectors and are more naturally represented in terms of finite combinatorial structures such as, for example, point patterns, strings, trees, lattices or graphs. Such learning problems arise in a variety of applications, which range from predicting the biological activity of a given chemical structure over finding frequent substructures of a data set of chemical compounds, and predicting the 3D-fold of a protein given its amino sequence, to natural language parsing, to name just a few.

In many applications, the set $\mathcal{X}$ of finite combinatorial structures is equipped with a distance function $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}_+$, which is often provided by external knowledge. An example of such a distance function is the edit distance on string, trees, or graphs (Levenshtein, 1966; Sanfeliu and Fu, 1983; Shapiro and Haralick, 1985; Shasha and Zhang, 1989; Zhang, 1996). The edit distance is applied to sequence alignment in bioinformatics (Gusfield, 1997), in chemoinformatics (Raymond and Willett, 2002) by means of the maximum common subgraph isomorphism, and in computer vision (Eshera and Fu, 1986; Myers, Wilson, and Hancock, 2000; Robles-Kelly and Hancock, 2005). Since distance spaces $(\mathcal{X}, d)$ of structures often have less mathematical structure than Banach spaces, several standard statistical pattern recognition techniques cannot be easily applied to $(\mathcal{X}, d)$.

There are two main approaches that apply standard statistical pattern recognition techniques to a given distance space $(\mathcal{X}, d)$. The first approach directly operates on the space $(\mathcal{X}, d)$. Examples are the *k*-nearest neighbor classifier, the linear programming machine (Graepel, Herbrich, Bollmann-Sdorra, and Obermayer, 1999), and pairwise clustering (Hofmann and Buhmann, 1997; Graepel and Obermayer, 1999). These methods can cope with sets $\mathcal{X}$ that possess an arbitrary distance function d as the sole mathematical structure on $\mathcal{X}$. The problem is that many pattern recognition methods require a space $\mathcal{X}$ with a richer mathematical structure. For example, large margin classifiers require as mathematical structure a complete vector space in which distances and angles can be measured. From an algorithmic point of view, many pattern recognition methods use local gradient information to minimize some cost function. For these methods, Banach spaces are endowed with enough structure to define derivatives and gradients.

The aim of the second approach is to overcome the lack of mathematical structure by embedding a given distance space $(\mathcal{X}, d)$ into a mathematically richer space $(\mathcal{X}', d')$. Several methods have been proposed, which mainly differ in the choice of the target space $\mathcal{X}'$ and to which extent the original distance function d is preserved. Typical examples are embeddings into Euclidean spaces (Cox and Cox, 2000; Luo, Wilson, and Hancock, 2003; Minh and Hofmann, 2004), Hilbert spaces (Gärtner, 2003; Hochreiter and Obermayer, 2004, 2006; Kashima, Tsuda, and Inokuchi, 2003; Lodhi, Saunders, Shawe-Taylor, Cristianini, and Watkins, 2002), Banach spaces (Hein, Bousquet, and Schölkopf, 2005; von Luxburg and Bousquet, 2004), and Pseudo-Euclidean spaces (Herbrich, Graepel, Bollmann-Sdorra, and Obermayer, 1998; Goldfarb, 1985; Pekalska, Paclik, and Duin, 2001).

During this transformation, one has to ensure that the relevant information of the original problem is preserved. Under the assumption that d is a reasonable distance function on $\mathcal{X}$ provided by some external knowledge, we can preserve the relevant information by isometrically embedding the original space $(\mathcal{X}, d)$ into some target space $(\mathcal{X}', d')$. Depending on the choice of the target space this is only possible if the distance function d satisfies certain properties. Suppose that $\mathcal{S} = \{x_1, \ldots, x_k\} \subseteq \mathcal{X}$ is a finite set and $D = (d_{ij})$ is a distance matrix with elements $d_{ij} = d(x_i, x_j)$. If d is symmetric and homogeneous, we can isometrically embed $\mathcal{D}$ into a Pseudo-Euclidean space (Goldfarb, 1985). In the case that d is a metric, the elements of $\mathcal{D}$ can be isometrically embedded into a Banach space. An isometric embedding of $\mathcal{S}$ into a Hilbert or Euclidean space is possible only if the matrix D^2 is of negative type (Schoenberg, 1937).[1]

Most standard learning methods have been developed in a Hilbert space or in a Euclidean space equipped with a Euclidean distance. But distance matrices of a finite set of combinatorial structures are often not of negative type and therefore an isometric embedding into a Hilbert space or Euclidean space is not possible. Another common problem of most isometric embeddings is that they only preserve distance relations and disregard knowledge about the inherent nature of the elements from the original space. For example the inherent nature of graphs is that they consist of a finite set of vertices together with a binary relation on that set. These information is lost, once we have settled in the target space for solving a pattern recognition problem. But for some methods in pattern recognition it is necessary to either directly access the original data or to recover the effects of the operations performed in the target space. One example is the sample mean of a set of combinatorial

1. A symmetric matrix M is of negative type if $x^T M x <= 0$ for all x with $x^T 1 = 0$.

structures (Jain and Obermayer, 2008; Jiang, Münger, and Bunke, 2001), which is a fundamental concept for several methods in pattern recognition such as principal component analysis and central clustering (Gold, Rangarajan, and Mjolsness, 1996; Günter and Bunke, 2002; Lozano and Escolano, 2003; Jain and Wysotzki, 2004; Bonev, Escolano, Lozano, Suau, Cazorla, and Aguilar, 2007; Jain and Obermayer, 2008). The sample mean of a set of vectors is the vector of sample means of each component of those vectors. Similarly, a sample mean of a set of combinatorial structures is a combinatorial structure composed of the sample means of the constituents parts the structure is composed of. Another example is finding frequent substructures in a given set of combinatorial structures (Dehaspe, Toivonen, and King, 1998; Yan and Han, 2002). For such problems a principled framework is missing.

In this contribution, we present a theoretical framework that isometrically and isostructurally embeds certain metric spaces $(\mathcal{X}, d)$ of combinatorial structures into a quotient space $(\mathcal{X}', d')$ of a Euclidean vector space. Instead of discarding information about the inherent nature of the original data, we can weaken the requirement that the embedding of $(\mathcal{X}, d)$ into $(\mathcal{X}', d')$ should be isometric for all metrics. Here, we focus on metrics d that are related to the pointwise maximum of a set of Euclidean distances. This restriction is acceptable from an application point of view, because we can show that such metrics on combinatorial structures and their related similarity functions are a common choice of proximity measure in a number of different applications (Gold, Rangarajan, and Mjolsness, 1996; Holm and Sander, 1993; Caprara, Carr, Istrail, Lancia, and Walenz, 2004).

The quotient space $(\mathcal{X}', d')$ preserves the distance relations and the nature of the original data. The related Euclidean space provides the mathematical structure that gives rise to a rich arsenal of learning methods. The goal of the proposed approach is to adopt standard learning methods based on local gradient information to learning on structures in the quotient space $\mathcal{X}'$. In order to do so, we need an approach that allows us to formally adopt geometrical and analytical concepts for finite combinatorial structures. The proposed approach maps combinatorial structures to equivalence classes of vectors, where the elements of the same equivalence class are different vector representations of the same structure. Mapping a combinatorial structure to an equivalence class of vectors rather than to a single vector provides a link to the geometry of Euclidean spaces and at the same time preserves the nature of the original data. The resulting quotient set (the set of equivalence classes) leads to the more abstract notion of $\mathcal{T}$-space. Formally, a $\mathcal{T}$-space $\mathcal{X}_{\mathcal{T}}$ over a vector space $\mathcal{X}$ is a quotient set of $\mathcal{X}$, where the equivalence classes are the orbits of the group action of a transformation group $\mathcal{T}$ on $\mathcal{X}$. We show that $\mathcal{T}$-spaces encompass a variety of different classes of combinatorial structures, which also includes vectors. Thus, the theory of $\mathcal{T}$-spaces generalizes the vector space concept to cope with combinatorial structures and aims at retaining the geometrical and algebraic properties of a vector space to a certain extent.

We present case studies to illustrate that the theoretical framework can be applied to machine learning applications.

This paper is organized as follows: Section 2 provides an overview about the basic idea of the proposed approach. In Section 3, we study $\mathcal{T}$-spaces $\mathcal{X}_{\mathcal{T}}$ over metric, normed, and inner product vector spaces $\mathcal{X}$. We show that the gradient of a smooth function on structures satisfies the necessary condition of optimality and is a well-defined structure pointing in direction of steepest ascent. In Section 4, we use the theory of $\mathcal{T}$-spaces to formulate selected problems in structural pattern recognition as continuous optimization problems. We show that the proposed cost functions are locally Lipschitz and therefore nonsmooth on a set of Lebesgue measure zero. For this class of functions, we can apply methods from nonsmooth optimization. As a case study, we discuss in Section 5 the

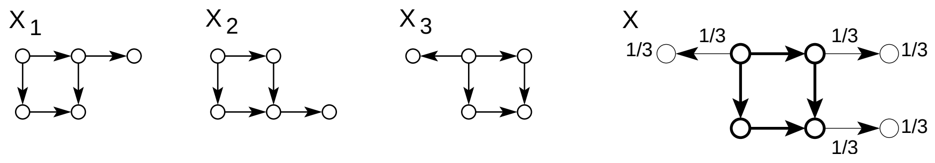

Figure 1: Illustration of a sample mean $\overline{X}$ of the three graphs $\mathcal{D} = \{X_1, X_2, X_3\}$. Vertices and edges of $\overline{X}$ that occur in all of the three example graphs from $\mathcal{D}$ are highlighted with bold lines. All other vertices and edges of $\overline{X}$ are annotated with the relative frequency of their occurrence in $\mathcal{D}$. By annotating the highlighted vertices and edges of $\overline{X}$ with 1, we obtain a weighted graph.

problem of determining a sample mean of a set of structures including its application to central clustering. As structures we consider point patterns and attributed graphs. Section 6 concludes. Technical parts and proofs have been delegated to the appendix.

2. An Example

The purpose of this section is to provide an overview about the basic idea of the proposed approach. To this end, we consider the (open) problem of determining the sample mean of graphs as a simple introductory example. The concept of a sample mean is the theoretical foundation for central clustering algorithms (see Section 4.3 and references therein).

A *directed graph* is a pair $X = (V, E)$ consisting of a finite set V of *vertices* and a set $E = \{(i, j) \in V \times V \, : \, i \neq j\}$ of *edges*.

By $\mathcal{G}$ we denote the set of all directed graphs. Suppose that

$$\mathcal{D} = (X_1, \ldots, X_k)$$

is a collection of k not necessarily distinct graphs from $\mathcal{G}$. Our goal is to determine a sample mean of $\mathcal{X}$. Intuitively, a sample mean averages the occurrences of vertices and edges within their structural context as illustrated in Figure 1.

As the sample mean of integers is not necessarily an integer, the sample mean of $\mathcal{D}$ is not necessarily a directed graph from $\mathcal{G}$ (see Figure 1). Therefore, we extend the set $\mathcal{G}$ of directed graphs to the set $\mathcal{G}[\mathbb{R}]$ of weighted directed graphs. A *weighted directed graph* is a triple $X = (V, E, \alpha)$ consisting of a directed graph (V, E) and a *weight function* $\alpha : V \cup V \to \mathbb{R}$ such that each edge has nonzero weight. A weighted directed graph X of order $|V| = n$ is completely specified by its *weight matrix* $X = (x_{ij})$ with elements $x_{ij} = \alpha(i, j)$ for all $i, j \in \{1, \ldots, n\}$.

The standard method

$$\overline{X} = \frac{1}{k} \sum_{i=1}^{k} X_i$$

to determine the sample mean $\overline{X}$ of $\mathcal{D}$ fails, because a well-defined addition of directed graphs is unknown as indicated by Figure 2. Therefore, we consider an equivalent characterization of the standard notion of sample mean. Following Jiang, Münger, and Bunke (2001), we adopt the optimization formulation of the standard sample mean. For vectors, the sample mean minimizes the sum of squared Euclidean distances from the data points. In line with this formulation, we define a sample mean of $\mathcal{D}$ as a global minimum of the cost function

$$F(X) = \sum_{i=1}^{k} D(X, X_i)^2, \tag{1}$$

where D is some appropriate distance function on $\mathcal{G}[\mathbb{R}]$ that measures structural consistent and inconsistent parts of the graphs under consideration.

In principle, we could use any ”well-behaved” distance function.[2] Here, we first consider distance functions on structures that generalize the Euclidean metric, because Euclidean spaces have a rich repository of analytical tools. To adapt at least parts of these tools for structure spaces, it seems to be reasonable to relate the distance function D in Equation (1) to the Euclidean metric. From an application point of view, this restriction is acceptable for the following reasons: (i) Geometric distance functions on graphs and their related similarity functions are a common choice of proximity measure in a number of different applications (Gold, Rangarajan, and Mjolsness, 1996; Holm and Sander, 1993); and (ii) it can be shown that a number of structure-based proximity measures like, for example, the maximum common subgraph (Raymond and Willett, 2002) or maximum contact map overlap problem for protein structure comparison (Goldman, Istrail, and Papadimitriou, 1999) can be related to an inner product and therefore to the Euclidean distance.

The geometric distance functions D we consider here are usually defined as the maximum of a set of Euclidean distances. This definition implies that (i) the cost function F is neither differentiable nor convex; (ii) the sample mean of graphs is not unique as shown in Figure 2; and (iii) determining a sample mean of graphs is NP-complete, because evaluation of D is NP-complete. Thus, we are faced with an intractable combinatorial optimization problem, where, at a first glance, a solution has to be found from an uncountable infinite set. In addition, multiple local minima of the cost function F complicates a characterization of a structural mean.

To deal with these difficulties, we embed graphs into a $\mathcal{T}$-space as we will show shortly. The basic idea is to view graphs as equivalence classes of vectors via their weight matrices, where the elements of the same equivalence class are different vector representations of the same structure. The resulting quotient set (the set of equivalence classes) leads to the more abstract notion of $\mathcal{T}$-space. Formally, a $\mathcal{T}$-space $\mathcal{X}_\mathcal{T}$ over a vector space $\mathcal{X}$ is a quotient set of $\mathcal{X}$, where the equivalence classes are the orbits of the group action of a transformation group $\mathcal{T}$ on $\mathcal{X}$. The theory of $\mathcal{T}$-spaces generalizes the vector space concept to cope with combinatorial structures and aims at retaining the geometrical and algebraic properties of a vector space to a certain extent. In doing so, the $\mathcal{T}$-space concept not only clears the way to approach the structural version of the sample mean in a principled way, but also generalizes standard techniques of learning in structured domains.

2.1 The Basic Approach

To construct $\mathcal{T}$-spaces, we demand that all graphs are of bounded order n, where the bound n can be chosen arbitrarily large. For a pattern recognition application this is not a serious restriction, because we can assume that the data graphs of interest are of bounded order. In the second step, we align each weighted directed graph X of order $m < n$ to a graph X' of order n by adding $p = n - m$

2. As we will see later, a distance function is well-behaved if it is locally Lipschitz.

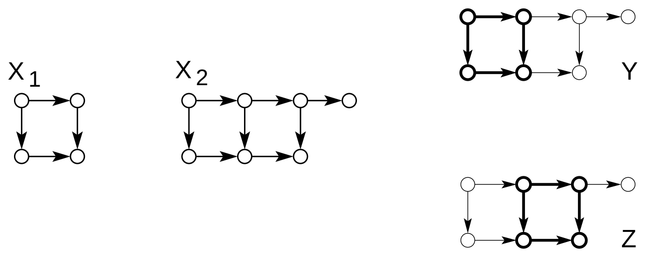

Figure 2: Illustration of one key problem in the domain of graphs: the lack of a well-defined addition. The graph X_1 can be added to X_2 with respect to $D(X_1, X_2)$ in two different ways as indicated by the highlighted subgraphs of Y and Z. As a consequence, Y and Z can be regarded as two distinct sample means of X_1 and X_2.

isolated vertices. The weighted adjacency matrix of the aligned graph X' is then of the form

$$X' = \begin{pmatrix} X & 0_{m,p} \\ 0_{p,m} & 0_{p,p} \end{pmatrix},$$

where X is the weighted adjacency matrix of X, and $0_{m,p}$, $0_{p,m}$, $0_{p,p}$ are padding zero matrices. By $\mathcal{G}[\mathbb{R}, n]$ we denote the set of weighted directed graphs of bounded order n.

For practical issues, it is important to note that restricting to structures of bounded order n and alignment of structures are purely technical assumptions to simplify mathematics. For machine learning problems, these limitations should have no practical impact, because neither the bound n needs to be specified nor an alignment of all graphs to an identical order needs to be performed. In a practical setting, we cancel out both technical assumptions by considering structure preserving mappings between the vertices of X and Y. Thus, when applying the theory, all we actually require is that the graphs are finite. We will return to this issue later, when we have provided the necessary technicalities.

The positions of the diagonal elements of X determine an ordering of the vertices. Conversely, different orderings of the vertices may result in different matrices. Since we are interested in the structure of a graph, the ordering of its vertices does not really matter. Therefore, we consider two matrices X and X' as being equivalent, denoted by $X \sim X'$, if they can be obtained from one another by reordering the vertices. Mathematically, the equivalence relation can be written as

$$X \sim X' \Leftrightarrow \exists P \in \mathcal{T} : P^{\mathsf{T}} X P = X',$$

where $\mathcal{T}$ denotes the set of all $(n \times n)$-permutation matrices.[3] The set $\mathcal{T}$ together with the function composition $T \circ T'$ for all $T, T' \in \mathcal{T}$ forms an algebraic group. By $[X]$ we denote the equivalence class of all matrices equivalent to X. Occasionally, we also refer to $[X]$ as the equivalence class of the graph X.

There are $n!$ different orderings of the vertices for an arbitrary graph X with n vertices. Each of the $n!$ orderings determines a weighted adjacency matrix. The equivalence class of X consist of all its different matrix representation. Note that different orderings of the vertices may result in the same matrix representation of the graph.

3. The letter $\mathcal{T}$ stands for transformation.

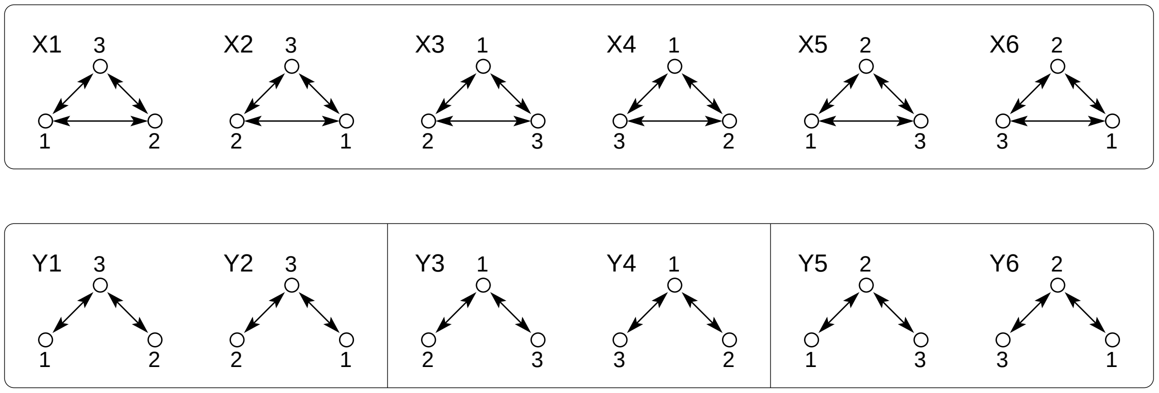

Figure 3: Illustration of two graphs with all possible orderings of their vertices. The number attached to the vertices represents their order (and are *not* attributes). The ordered graphs are grouped together according to their matrix representations. All ordered graphs $X1$-$X6$ yield the same weighted adjacency matrix. In the second row, the pairs $(Y1,Y2)$, $(Y3,Y4)$, and $(Y5,Y6)$ result in identical matrices.

Example 1 *Consider the graphs $X = X1$ and $Y = Y1$ depicted in the first column of Figure 3. The numbers annotated to the vertices represent an arbitrarily chosen ordering. Suppose that all vertices and edges have attribute 1. Then the weighted adjacency matrices of X and Y given the chosen ordering of their vertices are of the form*

$$X = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \qquad \textit{and} \qquad Y = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

The first and second row of Figure 3 show the $3! = 6$ different orderings of X and Y, respectively.

The matrix representation of X is independent of its ordering, that is, each reordering of the vertices of X results in the same matrix representation. Hence, the equivalence class of X consists of the singleton X.

The 6 different orderings of graph Y result in three different matrix representations. The equivalence class of Y is of the form

$$[Y] = \left\{ \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix} \right\},$$

where the first matrix refers to the ordering of $Y = Y1$ and $Y2$, the second to $Y3$ and $Y4$, and the third to $Y5$ and $Y6$.

Since we may regard matrices as vectors, we can embed X into the vector space $\mathcal{X} = \mathbb{R}^{n \times n}$ as the set $[X]$ of all vector representations of X. We call the quotient set

$$\mathcal{X}_{\mathcal{T}} = \mathcal{X}/\sim = \bigcup_{X \in \mathcal{X}} [X]$$

consisting of all equivalence classes $\mathcal{T}$*-space* over the *representation space* $\mathcal{X}$. Figure 4 depicts an embedding of a graph into a vector space.

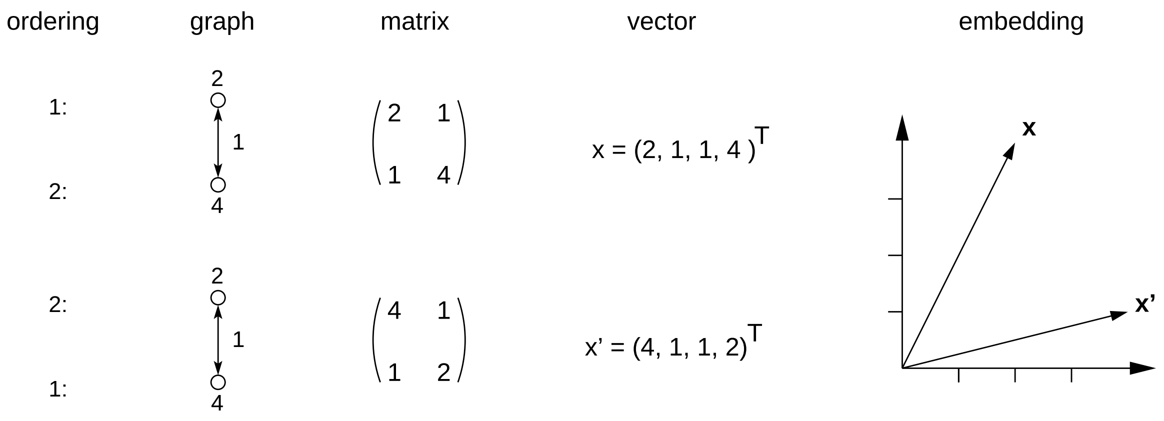

Figure 4: Illustration of an embedding of a graph of order 2. The attributes of the vertices are 2 and 4, and the attribute of the bidirectional edge is 1. Depending on the ordering of the vertices, we obtain two different matrix representations. Stacking the columns of the matrix to a 4-dimensional vector yields the vector representations x and x'. The plot in the last column depicts both representation vectors by considering their first and fourth dimension. Thus a graph is represented as a set of vectors in some vector space.

3. $\mathcal{T}$-spaces

In this section, we formalize the ideas on $\mathcal{T}$-spaces of the previous section. We consider a more general setting in the sense that we include classes of finite structures other than directed graphs with attributes from arbitrary vector spaces rather than weights from $\mathbb{R}$. The chosen approach that allows to formally adopt geometrical and analytical concepts makes use of the notion of *r-structures*. We introduce r-structures in Section 3.1. Based on the notion of r-structures, we develop the theory of $\mathcal{T}$-spaces in Sections 3.2 and 3.3. For a detailed technical treatment of $\mathcal{T}$-spaces we refer to Appendix A and B. Finally, Section 3.4 considers optimization of locally Lipschitz functions on $\mathcal{T}$spaces.

3.1 Attributed r-Structures

A *r-structure* is a pair $X = (\mathcal{P}, \mathcal{R})$ consisting of a finite set $\mathcal{P} \neq \emptyset$, and a subset $\mathcal{R} \subseteq \mathcal{P}^r$. The elements of $\mathcal{P}$ are the *points* of the r-structure X, the elements of $\mathcal{R}$ are its r-ary *relations*. A r-structure with points $\mathcal{P}$ is said to be a r-structure *on* $\mathcal{P}$. For convenience, we occasionally identify the structure X on $\mathcal{P}$ with its relation $\mathcal{R}$.

The following examples serve to indicate that several types of combinatorial structures can be regarded as r-structures. We first show that graphs are 2-structures. For this, we use the following notation: Given a finite set $\mathcal{P}$ of points, let

$$\mathcal{P}^{[2]} = \{(p,q) : p,q \in \mathcal{P}, p \neq q\}$$

be the set of tuples from $\mathcal{P}^2$ without diagonal elements (p,p).

Example 2 (Graphs) *Let $\mathcal{P}$ be a finite set of points, and let $X = (\mathcal{P}, \mathcal{R})$ be a* 2*-structure with $\mathcal{R} \subseteq \mathcal{P}^2$.*

1. *X is a* directed graph *if $\mathcal{R} \subseteq \mathcal{P}^{[2]}$.*
2. *X is a* simple graph *if $\mathcal{R} \subseteq \mathcal{P}^{[2]}$ such that $(p,q) \in \mathcal{R}$ implies $(q,p) \in \mathcal{R}$.*
3. *X is a* simple graph with loops *if $\mathcal{R} \subseteq \mathcal{P}^2$ such that $(p,q) \in \mathcal{R}$ implies $(q,p) \in \mathcal{R}$. Loops are edges (p,p) with the same endpoints.*

In a similar way, we can define further types of graphs such as, for example, trees, directed acyclic graphs, complete graphs, and regular graphs as 2-structures by specifying the corresponding properties on $\mathcal{R}$.

The next example shows that elements of a set are 1-structures.

Example 3 (Set of Elements) *Let $\mathcal{P}$ be a finite set of points. The* elements *$E(\mathcal{P}) = (\mathcal{P}, \mathcal{R})$ is a* 1-*structure with $\mathcal{R} = \mathcal{P}$, that is its relations are the elements of $\mathcal{P}$.*

To introduce analytical concepts to functions on r-structures, we shift from discrete to continuous spaces by introducing attributes. Let $\mathcal{A} = \mathcal{R}^d$ denote the set of attributes. An *$\mathcal{A}$-attributed r-structure* is a triple $X_\alpha = (\mathcal{P}, \mathcal{R}, \alpha)$ consisting of a r-structure $X = (\mathcal{P}, \mathcal{R})$ and an *attribution* $\alpha : \mathcal{P}^r \to \mathcal{A}$ with $\alpha(p) \neq 0$ if, and only if, $p \in \mathcal{R}$. Besides the technical argument, attributions also have a practical relevance, because they are often used to enhance descriptions of structured objects. The next example collects some attributed structures.

Example 4 *Let $\mathcal{P}$ be a finite set of order n.*

- Attributed graphs*: Let $\mathcal{A} = \mathbb{R}^d$. An* attributed graph *is an $\mathcal{A}$-attributed* 2-*structure $G_\alpha = (\mathcal{P}, \mathcal{R}, \alpha)$, where $G = (\mathcal{P}, \mathcal{R})$ is a simple graph with loops and $\alpha : \mathcal{R} \to \mathcal{A}$ is an attribution that assigns each vertex (loop) and each edge a non-zero feature vector.*
- Point patterns*: Let $\mathcal{A} = \mathbb{R}^2$. A* point pattern *is an $\mathcal{A}$-attributed* 1-*structure $P_\alpha = (\mathcal{P}, \mathcal{R}, \alpha)$, where $E(\mathcal{P}) = (\mathcal{P}, \mathcal{R})$ are the elements of $\mathcal{P}$ and $\alpha : \mathcal{R} \to \mathcal{A}$ is an attribution that assigns each element $p \in \mathcal{P}$ its coordinates $\alpha(p)$.*

The next example shows that vectors are attributed 1-structures. Hence, all results on r-structures are also valid in vector spaces.

Example 5 *Let $\mathcal{A}$ be a vector space. Suppose that $\mathcal{P}$ is of order $n = 1$. A* vector *is an $\mathcal{A}$-attributed* 1-*structure $x_\alpha = (\mathcal{P}, \mathcal{R}, \alpha)$, where $E(\mathcal{P}) = (\mathcal{P}, \mathcal{R})$ is the single element of $\mathcal{P}$ and $\alpha : \mathcal{R} \to \mathcal{A}$ is an attribution that maps a singleton to a vector. Hence, the set of all possible structures x_α on $\mathcal{P}$ reproduces the vector space $\mathcal{A}$.*

Note that we may assume without loss of generality that the attributes of r-relations from $\mathcal{R}$ are nonzero, that is $\alpha(\mathcal{R}) \subseteq \mathcal{A} \setminus \{0\}$. If the zero vector 0 is required as a valid attribute of a r-relation, we can always change, for example, to the vector space $\mathcal{A}' = \mathcal{A} \times \mathbb{R}$ and redefine α as

$$\alpha' : \mathcal{P}^r \to \mathcal{A} \times \mathbb{R}, \quad p \mapsto \begin{cases} (0,0) & : \; p \in \mathcal{P}^r \setminus \mathcal{R} \\ (\alpha(p), 1) & : \; p \in \mathcal{R} \end{cases}.$$

An $\mathcal{A}$-attributed r-structure $X = (\mathcal{P}, \mathcal{R}, \alpha)$ is completely specified by its *matrix representation* $X = (x_{p_1 \dots p_r})$ with elements

$$x_{p_1 \dots p_r} = \alpha(p_1, \dots, p_r)$$

for all $p = (p_1, \dots, p_r) \in \mathcal{P}^r$. For example, the matrix representation of a simple graph is its ordinary adjacency matrix.

Neither the "nature" of the points $\mathcal{P}$ nor the particular form of the r-relations $\mathcal{R}$ of a given r-structure $X = (\mathcal{P}, \mathcal{R}, \alpha)$ do really matter. What matters is the structure described by $\mathcal{R}$. Suppose that X is of order $|\mathcal{P}| = n$. To abstract from the "nature" of points, we choose $\mathbb{Z}_n = \{1, \dots, n\}$ as our standard set of points. The particular form of $\mathcal{R}$ depends on the numbering of the points from $\mathbb{Z}_n$. To abstract from the particular form of $\mathcal{R}$, we identify sets of r-relations that can be obtained from one another by renumbering the points. Mathematically, we can express these sets by means of *isomorphism classes*. Two $\mathcal{A}$-attributed r-structures $X = (\mathcal{P}, \mathcal{R}, \alpha)$ and $X' = (\mathcal{P}', \mathcal{R}', \alpha')$ are *isomorphic*, written as $X \simeq X'$ if there is a bijective mapping $\phi : \mathcal{P} \to \mathcal{P}'$ satisfying

1. $p = (p_1, \dots, p_r) \in \mathcal{R} \Leftrightarrow \phi(p) = (\phi(p_1), \dots, \phi(p_r)) \in \mathcal{R}'$
2. $\alpha(p) = \alpha'(\phi(p))$ for all $p \in \mathcal{R}$.

The *isomorphism* class $[X]$ of X consists of all $\mathcal{A}$-attributed r-structures on $\mathcal{P} = \mathbb{Z}_n$ that are isomorphic to X. By $\mathcal{S}_{\mathcal{A}}^{n,r}$ we denote the set of all $\mathcal{A}$-attributed r-structures on $\mathcal{P} = \mathbb{Z}_n$ and by $\left[\mathcal{S}_{\mathcal{A}}^{n,r}\right]$ the set of all isomorphism classes of structures from $\mathcal{S}_{\mathcal{A}}^{n,r}$.

We can identify any r-structure $X = (\mathbb{Z}_m, \mathcal{R}, \alpha)$ of order $m < n$ with a structure of order n by adding $q = n - m$ isolated points. The aligned structure is then of the form $X' = (\mathbb{Z}_n, \mathcal{R}, \alpha')$, where

$$\alpha'(p) = \begin{cases} \alpha(p) & : \quad p \in \mathcal{R} \\ 0 & : \quad otherwise \end{cases} .$$

Using alignment, we can regard $\mathcal{S}_{\mathcal{A}}^{n,r}$ as the set of $\mathcal{A}$-attributed r-structures of bounded order n. Similarly, we may think of $\left[\mathcal{S}_{\mathcal{A}}^{n,r}\right]$ as the set of abstract $\mathcal{A}$-attributed r-structures of bounded order n. Again recall that specifying a bound n and aligning smaller structures to structures of order n are purely technical assumptions to simplify mathematics.

3.2 $\mathcal{T}$-Spaces

Let $\mathcal{X} = \mathbb{R}^n$ be the n-dimensional Euclidean vector space, and let $\mathcal{T}$ be a subgroup of the group of all $n \times n$ permutation matrices. Then the binary operation

$$\cdot : \mathcal{T} \times \mathcal{X} \to \mathcal{X}, \quad (T, x) \mapsto Tx$$

is a group action of $\mathcal{T}$ on $\mathcal{X}$. For $x \in \mathcal{X}$, the *orbit* of x is denoted by

$$[x]_{\mathcal{T}} = \{Tx : T \in \mathcal{T}\}.$$

If no misunderstanding can occur, we write $[x]$ instead of $[x]_{\mathcal{T}}$.

A $\mathcal{T}$-*space* over $\mathcal{X}$ is the orbit space $\mathcal{X}_{\mathcal{T}} = \mathcal{X}/\mathcal{T}$ of all orbits of $x \in \mathcal{X}$ under the action of $\mathcal{T}$. We call $\mathcal{X}$ the *representation space* of $\mathcal{X}_{\mathcal{T}}$. By

$$\mu : \mathcal{X} \to \mathcal{X}_{\mathcal{T}}$$

we denote the *membership function* that sends vector representations to the structure they describe.

$\mathcal{T}$-spaces are a convenient abstraction of r-structures in order to adopt geometrical and analytical concepts. To see this, let $\mathcal{A} = \mathbb{R}^d$ and let $\mathcal{X} = \mathcal{A}^N$, where $N = n^r$. Via the matrix representations, we can identify r-structures from $\mathcal{S}_{\mathcal{A}}^{n,r}$ as vectors from $\mathcal{X}$. Obviously, we have a relaxation in the sense that $\mathcal{S}_{\mathcal{A}}^{n,r} \subseteq \mathcal{X}$ and $\left[\mathcal{S}_{\mathcal{A}}^{n,r}\right] \subseteq \mathcal{X}_{\mathcal{T}}$ such that μ restricted on $\mathcal{S}_{\mathcal{A}}^{n,r}$ sends vector representations to the structures they represent. Note that there are structures in $\mathcal{X}_{\mathcal{T}}$ that are not well-defined r-structures from $\left[\mathcal{S}_{\mathcal{A}}^{n,r}\right]$. Hence, care must be taken when applying $\mathcal{T}$-spaces.

The following notations, definitions, and results are useful to simplify technicalities. We use capital letters $X, Y, Z, \ldots$ to denote the elements of $\mathcal{X}_{\mathcal{T}}$. Suppose that $X = \mu(x)$ for some $x \in \mathcal{X}$. Then we identify X with $[x]$ and make use of sloppy notations like, for example, $x \in X$ to denote $x \in [x]$.[4]

Let $f : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a symmetric function satisfying $f(x,y) = f(y,x)$ for all $x, y \in \mathcal{X}$. Then f induces symmetric functions

$$\begin{aligned} F^* : \mathcal{X}_{\mathcal{T}} \times \mathcal{X}_{\mathcal{T}} \to \mathbb{R}, &\quad (X,Y) \mapsto \max\{f(x,y) : x \in X, y \in Y\}, \\ F_* : \mathcal{X}_{\mathcal{T}} \times \mathcal{X}_{\mathcal{T}} \to \mathbb{R}, &\quad (X,Y) \mapsto \min\{f(x,y) : x \in X, y \in Y\}. \end{aligned}$$

Since $\mathcal{T}$ is finite, the orbits $[x]$ of x are finite. Hence, F^* and F_* assume an extremal value. We call F^* *maximizer* and F_* *minimizer* of f on $\mathcal{X}_{\mathcal{T}} \times \mathcal{X}_{\mathcal{T}}$.

An inner product $\langle \cdot, \cdot \rangle$ on $\mathcal{X}$ gives rise to a maximizer of the form

$$\langle \cdot, \cdot \rangle^* : \mathcal{X}_{\mathcal{T}} \times \mathcal{X}_{\mathcal{T}} \to \mathbb{R}, \quad (X,Y) \mapsto \max\{\langle x, y \rangle : x \in X, y \in Y\}.$$

We call $\langle \cdot, \cdot \rangle^*$ *inner $\mathcal{T}$-product* induced by $\langle \cdot, \cdot \rangle$. The inner $\mathcal{T}$-product is *not* an inner product, because the maximum-operator in the definition of $\langle \cdot, \cdot \rangle^*$ does not preserve the bilinearity property of an inner product. But we can show that an inner $\mathcal{T}$-product satisfies some weaker properties.

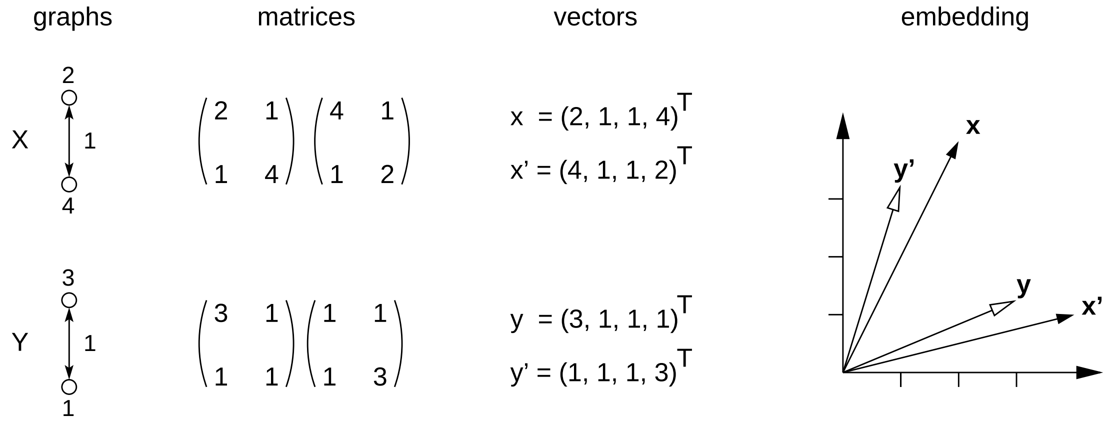

Figure 5: Illustration of two example graphs and their embeddings in a vector space (see Figure 4 for a detailed description).

Example 6 *Consider the graphs X and Y from Figure 5. We have*

$$\langle X, Y \rangle^* = \langle x, y' \rangle = \langle x', y \rangle = 16.$$

4. The notation is sloppy, because X is an element in $\mathcal{X}_{\mathcal{T}}$ and not a set, whereas $[x]$ is a set of equivalent elements from $\mathcal{X}$.

Thus, to determine $\langle X, Y\rangle^*$, we select vector representations $\tilde{x}$ of X and $\tilde{y}$ of Y that have closest angle and then evaluate $\langle \tilde{x}, \tilde{y}\rangle$.

Any inner product space $\mathcal{X}$ is a normed space with norm $\|x\| = \sqrt{\langle x, x\rangle}$ and a metric space with metric $d(x,y) = \|x-y\|$. The norm $\|\cdot\|$ and the metric d on $\mathcal{X}$ give rise to minimizers $\|\cdot\|_*$ of $\|\cdot\|$ and D_* of d on $\mathcal{X}_\mathcal{T}$.

Since elements from $\mathcal{T}$ preserve lengths and angles, we have

$$\|Tx\| = \|Tx - 0\| = \|Tx - T0\| = \|x - 0\| = \|x\|$$

for all $T \in \mathcal{T}$. Hence, $\|X\|_*$ is independent from the choice of vector representation. We call the minimizer $\|\cdot\|_*$ the *$\mathcal{T}$-norm* induced by the norm $\|\cdot\|$. A $\mathcal{T}$-norm is related to an inner $\mathcal{T}$-product in the same way as a norm to an inner product. We have

1. $\langle X, X\rangle^* = \langle x, x\rangle$ for all $x \in X$.
2. $\|X\|_* = \sqrt{\langle X, X\rangle^*}$.

Note that a $\mathcal{T}$-norm is *not* a norm, because a $\mathcal{T}$-space has no well defined addition. But we can show that a $\mathcal{T}$-norm has norm-like properties.

Example 7 *Consider the graphs X and Y from Figure 5. To determine their $\mathcal{T}$-norm, it is sufficient to compute the standard norm of an arbitrarily chosen vector representation. Hence, we have*

$$\|X\|_* = \|x\| = \|x'\| = \sqrt{22},$$
$$\|Y\|_* = \|y\| = \|y'\| = \sqrt{12}.$$

The minimizer D_* of the Euclidean metric $d(x,y) = \|x-y\|$ is also a metric. To distinguish from ordinary metrics, we call the minimizer D_* of a Euclidean metric d on $\mathcal{X}$ the $\mathcal{T}$-metric induced by d. We can express the metric D_* in terms of $\langle \cdot, \cdot \rangle^*$ as follows:

$$D_*(X,Y)^2 = \|X\|_*^2 - 2\langle X, Y\rangle^* + \|Y\|_*^2.$$

Example 8 *Consider the graphs X and Y depicted in Figure 5. To determine $D_*(X,Y)$, we select vector representations $\tilde{x}$ of X and $\tilde{y}$ of Y that have minimal distance $d(\tilde{x}, \tilde{y})$. Then we find that*

$$D_*(X,Y) = d(x, y') = d(x', y) = \sqrt{2}.$$

3.3 Functions on $\mathcal{T}$-Spaces

A *$\mathcal{T}$-function* is a function of the form

$$F : \mathcal{X}_\mathcal{T} \to \mathbb{R},$$

where $\mathcal{X}_\mathcal{T}$ is a T-space over $\mathcal{X}$. Instead of considering the $\mathcal{T}$-function F, it is often more convenient to consider its *representation function*

$$f : \mathcal{X} \to \mathbb{R}, \quad x \mapsto F \circ \mu(x),$$

which is invariant under transformations from elements of $\mathcal{T}$.

Here, the focus is on $\mathcal{T}$-functions that are locally Lipschitz. A $\mathcal{T}$-function is *locally Lipschitz* if, and only if, its representation function is locally Lipschitz. We refer to Appendix C for basic definitions and properties from nonsmooth analysis of locally Lipschitz functions.

Suppose that F is a locally Lipschitz function with representation function f. By Rademacher's Theorem 23, f is differentiable almost everywhere. In addition, at non-differentiable points, f admits the concept of generalized gradient. The concepts differentiability and gradient can be transfered to $\mathcal{T}$-spaces in a well-defined way. Assume that f is differentiable at some point $x \in \mathcal{X}$ with gradient $\nabla f(x)$. Then f is differentiable at all points $Tx \in X$ with $T \in \mathcal{T}$ and the gradient of f at Tx is of the form

$$\nabla f(Tx) = T\nabla f(x).$$

We say F is $\mathcal{T}$-differentiable at X, if its representation function f is differentiable at an arbitrary vector representation $x \in X$. The well-defined structure

$$\nabla F(X) = \mu(f(x))$$

is the *$\mathcal{T}$-gradient* of F at X pointing in direction of steepest ascent.

3.4 Optimization of Locally Lipschitz $\mathcal{T}$-Functions

A standard technique in machine learning and pattern recognition is to pose a learning problem as an optimization problem. Here, we consider the problem of solving optimization problems of the form

$$(P1) \qquad \begin{array}{ll} \text{minimize} & F : \mathcal{X}_{\mathcal{T}} \to \mathbb{R}, \qquad X \mapsto \sum_{i=1}^{k} F_i(X) \\ \text{subject to} & X \in \mathcal{U}_{\mathcal{T}} \end{array}$$

where the component functions F_i are locally Lipschitz $\mathcal{T}$-functions and $\mathcal{U}_{\mathcal{T}} \subset \mathcal{X}_{\mathcal{T}}$ is the feasible set of admissible solutions. Then according to Prop. 21, the cost function F is also locally Lipschitz, and we can rewrite (P1) to an equivalent optimization problem

$$(P2) \qquad \begin{array}{ll} \text{minimize} & f : \mathcal{X} \to \mathbb{R}, \qquad x \mapsto \sum_{i=1}^{k} f_i(x) \\ \text{subject to} & x \in \mathcal{U} \end{array}$$

where the component functions f_i are the representation functions of F_i and $\mathcal{U} \subseteq \mathcal{X}$ is the feasible set with $\mu(\mathcal{U}) = \mathcal{U}_{\mathcal{T}}$. Hence, f is the representation function of the locally Lipschitz $\mathcal{T}$-function F and therefore also locally Lipschitz.

To minimize locally Lipschitz functions, the field of nonsmooth optimization offers a number of techniques. A survey of classical methods can be found in Mäkelä and Neittaanmäki (1992); Shor (1985). As an example, we describe subgradient methods, which are easy to implement and well-suited to identify the difficulties arising in nonsmooth optimization. Algorithm 1 outlines the basic procedure:

Algorithm 1 (Basic Incremental Algorithm)

```
choose starting point $x_1 \in \mathcal{U}$ and set $t := 0$
repeat
    set $\tilde{x}_{t,1} := x_1$
    for $i = 1, \ldots, k$ do
        direction finding:
            determine $d_{t,i} \in \mathcal{X}$ and $\eta > 0$ s.t. $\tilde{x}_{t,i} + \eta d_{t,i} \in \mathcal{U}$ and
                $f_i(\tilde{x}_{t,i} + \eta d_{t,i}) < f_i(\tilde{x}_{t,i})$
        line search:
            find step size $\eta_{t,i} > 0$ such that $\tilde{x}_{t,i} + \eta_{t,i} d_{t,i} \in \mathcal{U}$ and
                $\eta_{t,i} \approx \arg\min_{\eta > 0} f_i(\tilde{x}_{t,i} + \eta d_{t,i})$
        updating:
            set $\tilde{x}_{t,i+1} := \tilde{x}_{t,i} + \eta_{t,i} d_{t,i}$
    Set $x_{t+1} := \tilde{x}_{t,k+1}$
    Set $t := t + 1$
until some termination criterion is satisfied
```

To explain the algorithm, we first consider the case that f is smooth and $\mathcal{U} = \mathcal{X}$. In the step *direction finding*, we generate a descent direction by exploiting the fact that the direction opposite to the gradient is locally the steepest descent direction. Line search usually employs some efficient univariate smooth optimization method or polynomial interpolation. The necessary condition for a local minimum yields a termination criterion. Now suppose that f is locally Lipschitz. Then f admits a generalized gradient at each point. The generalized gradient coincides with the gradient at differentiable points and is a convex set of subgradients at non-differentiable points. For more details, we refer to Appendix C.

Subgradients, the elements of a generalized gradient, play a very important role in algorithms for non-differentiable optimization. The basic idea of subgradient methods is to generalize the methods for smooth problems by replacing the gradient by an arbitrary subgradient. In the direction finding step, Algorithm 1 computes an arbitrary subgradient $d \in \partial f(x)$ at the current point x. If f is differentiable at x, then the subgradient d coincides with the gradient $\nabla f(x)$. If in addition $d \neq 0$, then the opposite direction $-d$ is the direction of steepest descent. On the other hand, if f is not differentiable at x, then $-d$ is not necessarily a direction of descent of f at x. But since f is differentiable almost everywhere by Rademacher's Theorem 23, the set of non-differentiable points is a set of Lebesgue measure zero.

Line search uses predetermined step sizes $\eta_{t,i}$, instead of an exact or approximate line search as in the gradient method. One reason for this is that the direction $-d$ computed in the direction finding step is not necessarily a direction of descent. Thus, the viability of subgradient methods depend critically on the sequence of step sizes. One common choice are step sizes that satisfy

$$\sum_{t=0}^{\infty} \eta_t = \infty \quad \text{and} \quad \sum_{t=0}^{\infty} \eta_t^2 < \infty,$$

where $\eta_t = \eta_{t,1}$.

To formulate a termination criterion, we could—in principle—make use of the following necessary condition of optimality.

Theorem 1 *Let $f : \mathcal{X} \to \mathbb{R}$ be locally Lipschitz at its minimum (maximum) $x \in \mathcal{R}$. Then*

$$0 \in \partial f(x).$$

At non-differentiable points, however, an arbitrary subgradient provides no information about the existence of the zero in the generalized gradient $\partial f(x)$. Therefore, when assuming an instantly decreasing step size, one reliable termination criterion stops the algorithm as soon as the step size falls below a predefined threshold.

Since the subgradient method is not a descent method, it is common to keep track of the best point found so far, which is the one with smallest function value. For further advanced and more sophisticated techniques to minimize locally Lipschitz functions, we refer to Mäkelä and Neittaanmäki (1992); Shor (1985).

We conclude this section with a remark on determining intractable subgradients in a practical setting.

3.4.1 APPROXIMATING SUBGRADIENTS

Nonsmooth optimization as discussed in Mäkelä and Neittaanmäki (1992); Shor (1985) assumes that at each point x we can evaluate at least one subgradient $y \in \partial f(x)$ and the function value $f(x)$. In principle, this should be no obstacle for the class of problems we are interested in. In a practical setting, however, evaluating a subgradient as descent direction can be computationally intractable. For example, the pattern recognition problems described later in Section 4.2-4.5 are all computationally efficient for structures like point patterns, but NP-hard for structures like graphs. A solution to this problem is to approximate a subgradient by using polynomial time algorithms. An approximated subgradient corresponds to a direction that is no longer a subgradient of the cost function. In particular, at smooth points, an approximated (sub)gradient (hopefully) corresponds to a descent direction close to the direction of steepest descent. We call Algorithm 1 an *approximate incremental subgradient methods* if the direction finding step produces directions that are not necessarily subgradients of the corresponding component function f_i.

We replace the subgradient by a computationally cheaper approximation as a direction of descent. In a computer simulation, we show that determining a sample mean of weighted graph is indeed possible when using approximate subgradient methods.

Suppose that $\mathcal{X}_\mathcal{T}$ is the $\mathcal{T}$-space of simple weighted graphs over $\mathcal{X} = \mathbb{R}^{n \times n}$, and let $\mathcal{U}_\mathcal{T} \subseteq \mathcal{X}_\mathcal{T}$ be the subset of weighted graphs with attributes from the interval $[0,1]$. Our goal is to determine a sample mean of a collection of simple weighted graphs $\mathcal{D} = \{\mathcal{X}_1, \ldots, \mathcal{X}_k\} \subseteq \mathcal{U}_\mathcal{T}$.

Given a representation x of X, the computationally intractable task is to find a representation x_i of X_i such that $(x, x_i) \in \operatorname{supp}\left(d^2_{X_i} | x\right)$. This problem is closely related to the problem of computing the distance $D_*(X, X_i)$, which is known to be an NP-complete *graph matching problem*. Hence, in a practical setting, exact algorithms that guarantee to find a subgradient as descent direction are useless for all but the smallest graphs. A solution to this problem is to approximate a subgradient by using polynomial time algorithms. An approximated subgradient corresponds to a direction that is no longer a subgradient of the cost function. In particular, at smooth points, an approximated (sub)gradient (hopefully) corresponds to a descent direction close to the direction of steepest descent. We call Algorithm 1 an *approximate incremental subgradient methods* if the direction finding step produces directions that are not necessarily subgradients of the corresponding component function f_i.

4. Pattern Recognition in $\mathcal{T}$-Spaces

This section shows how the framework of $\mathcal{T}$-spaces can be applied to solve problems in structural pattern recognition. We first propose a generic scheme for learning in distance spaces. Based on this generic scheme, we derive cost functions for determining a sample mean, central clustering, learning large margin classifiers, supervised learning in structured input and/or output spaces, and finding frequent substructures. Apart from the last problem, all other cost functions presented in this section extend standard cost functions from the vector space formalism to $\mathcal{T}$-spaces in the sense that we recover the standard formulations when regarding vectors as r-structures.

4.1 A Generic Approach: Learning in Distance Spaces

Without loss of generality, we may assume that $(\mathcal{X}_\mathcal{T}, D)$ is a distance space, where D is either the metric D_* induced by the Euclidean metric on $\mathcal{X}$ or another (not necessarily metric) distance function that is more appropriate for the problem to hand. A generic approach to solve a learning problem (P) in $\mathcal{X}_\mathcal{T}$ is as follows:

1. Transform (P) to an optimization problem, where the cost function F is a function defined on $\mathcal{X}_\mathcal{T}$.

2. Show that F is locally Lipschitz.

3. Optimize F using methods from nonsmooth optimization.

Since $\mathcal{X}_\mathcal{T}$ is a metric space over an Euclidean vector space, we can apply subgradient methods or other techniques from nonsmooth optimization to minimize locally Lipschitz $\mathcal{T}$-functions on $\mathcal{X}_\mathcal{T}$. If the cost function F depends on a distance measure D, we demand that D is locally Lipschitz to ensure the locally Lipschitz property for F.

4.2 The Sample Mean of k-Structures

The sample mean of structures is a basic concept for a number of methods in statistical pattern recognition. Examples include visualizing or comparing two populations of chemical graphs, and central clustering of structures (Section 4.3).

We define a sample mean of the k elements $X_1, \ldots, X_k \in \mathcal{X}_\mathcal{T}$ as a minimizer of

$$\begin{array}{ll} \text{minimize} & F(X) = \sum_{i=1}^{k} D(X, X_i)^2 \\ \text{subject to} & X \in \mathcal{X}_\mathcal{T} \end{array},$$

where D is a distance function on $\mathcal{X}_\mathcal{T}$. If D is locally Lipschitz, then F is also locally Lipschitz by Prop. 21.

First approaches to study averages of graphs have been pursued by Jiang, Münger, and Bunke (2001). They considered the set median and generalized median of a sample of graphs as a discrete optimization problem with a similar cost function as for the sample mean. To minimize the cost function, they applied a genetic algorithm to graphs with a small number of discrete attributes. In this contribution, we shift the problem of determining a sample mean from discrete to continuous optimization.

4.3 Central Clustering of *k*-Structures

Suppose that we are given a training sample $\mathcal{X} = \{X_1, \ldots, X_m\}$ consisting of m structures X_i drawn from the structure space $\mathcal{X}_{\mathcal{T}}$. The aim of central clustering is to find k cluster centers $\mathcal{Y} = \{Y_1, \ldots, Y_k\} \subseteq \mathcal{X}_{\mathcal{T}}$ such that the following cost function

$$F(M, \mathcal{Y}, \mathcal{X}) = \frac{1}{m} \sum_{j=1}^{k} \sum_{i=1}^{m} m_{ij} D(X_i, Y_j),$$

is minimized with respect to a given distortion measure D. The matrix $M = (m_{ij})$ is a $(m \times k)$-membership matrix with elements $m_{ij} \in [0,1]$ such that $\sum_j m_{ij} = 1$ for all $i = 1, \ldots, m$.

If the distortion measure is locally Lipschitz, then F as a function of the cluster centers Y_j is locally Lipschitz by Prop. 21.

A number of central clustering algorithms for graphs have been devised recently (Gold, Rangarajan, and Mjolsness, 1996; Günter and Bunke, 2002; Lozano and Escolano, 2003; Jain and Wysotzki, 2004; Bonev, Escolano, Lozano, Suau, Cazorla, and Aguilar, 2007). In experiments it has been shown that the proposed methods converge to satisfactory solutions, although neither the notion of cluster center nor the update rule of the cluster centers is well-defined. Because of these issues one might expect that central clustering algorithm could be prone to oscillations halfway between different cluster centers of the same cluster. An explanation why this rarely occurs can now be given. As long as the cost function is locally Lipschitz, almost all points are differentiable. For these points the update rule is well-defined. Hence, it is very unlikely that the aforementioned oscillations occur over a longer period of time, when using an optimization algorithm that successively decreases the step size.

4.4 Large Margin Classifiers

Consider the function

$$h_{W,b} : \mathcal{X}_{\mathcal{T}} \to \mathbb{R}, \quad X \mapsto \langle W, X \rangle^* + b,$$

where $W \in \mathcal{X}_{\mathcal{T}}$ is the *weight structure* and $b \in \mathbb{R}$ the *bias*. The discriminant $h_{W,b}$ implements a two-category classifier in the obvious way: Assign an input structure X to the class labeled $+1$ if $h_{W,b}(X) \geq 0$ and to -1 if $h_{W,b}(X) < 0$.

Suppose that $\mathcal{Z} = \{(X_1, y_1), \ldots, (X_k, y_k)\}$ is a training sample consisting of k training structures $X_i \in \mathcal{X}_{\mathcal{T}}$ together with corresponding labels $y_i \in \{\pm 1\}$. We say, $\mathcal{Z}$ is *$\mathcal{T}$-separable* if there exists a $W_0 \in \mathcal{X}_{\mathcal{T}}$ and $b_0 \in \mathbb{R}$ with

$$h_*(X) = \langle W_0, X \rangle^* + b_0 = y$$

for all $(X, y) \in \mathcal{Z}$.

To find an "optimal" discriminant that correctly classifies the training examples, we construct the cost function

$$F(W, b, \alpha) = \frac{1}{2} \|W\|_*^2 - \sum_{i=1}^{k} \alpha_i \left(y_i \left(\langle W, X_i \rangle^* + b \right) - 1 \right),$$

where the $\alpha_i \geq 0$ are the Lagrangian multipliers. The representation function of F is of the form

$$f(w,b,\alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{k} \alpha_i y_i \left(s_i(w,b) - 1\right),$$

where $s_i(w,b) = \max_{T \in \mathcal{T}} \langle w, Tx_i \rangle + b$. The elements $w \in W$ and $x_i \in X_i$ are arbitrary. The first term of f is smooth and convex (and therefore locally Lipschitz). The locally Lipschitz property and convexity of the second term follows from the rules of calculus for locally Lipschitz functions (see Section C) and Prop. 20. Hence, f is locally Lipschitz and convex.

The structurally linear discriminants sets the stage to (i) explore large margin classifiers in structure spaces and (ii) construct neural learning machines for adaptive processing of finite structures. Subgradient methods for maximum margin learning has been applied in Ratliff, Bagnell, and Zinkevich (2006) for predicting structures rather than classes. Finally note that the inner $\mathcal{T}$-product as a maximizer of a set of similarities is not a kernel (Gärtner, 2005).

4.5 Supervised Learning

The next application example generalizes the problem of learning large margin classifiers for k-structures by allowing $\mathcal{T}$-spaces as input and as output space. Note that the in- and output spaces may consist of different classes of k-structures, for example, the input patterns can be feature vectors and the output space can be the domain of graphs.

Assume that we are given a a training sample $\mathcal{Z} = \{(X_1,Y_1),\ldots,(X_k,Y_k)\}$ consisting of k training structures X_i drawn from some $\mathcal{T}$-space $\mathcal{X}_\mathcal{T}$ over $\mathcal{X}$ together with corresponding output structures Y_i from a $\mathcal{T}'$-space $\mathcal{Y}_{\mathcal{T}'}$ over $\mathcal{Y}$. Given the training data $\mathcal{Z}$, our goal is to find an unknown functional relationship (hypothesis)

$$H : \mathcal{X}_\mathcal{T} \rightarrow \mathcal{Y}_\mathcal{T}$$

from a hypothesis space $\mathcal{H}$ that best predicts the output structures of unseen examples $(X,Y) \in \mathcal{X}_\mathcal{T} \times \mathcal{Y}_\mathcal{T}$ according to some cost function

$$F(H,\mathcal{Z}) = \frac{1}{k}\sum_{i=1}^{k} L(H(X_i),Y_i),$$

where $L : \mathcal{Y}_\mathcal{T} \times \mathcal{Y}_\mathcal{T} \rightarrow \mathbb{R}$ denotes the loss function.

The representation function of F is of the form

$$f(h,\mathcal{Z}) = \frac{1}{k}\sum_{i=1}^{k} \ell(h(x_i),y_i),$$

where $\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$ is the representation function of L, $h : \mathcal{X} \rightarrow \mathcal{Y}$ the representation function of H, and $x_i \in X_i$, $y_i \in Y_i$. We assume that the functions h have a parametric form and are therefore uniquely determined by the value of their parameter vector θ_h. We make this dependence of h on θ_h explicit by writing $f(\theta_h,\mathcal{Z})$ instead of $f(h,\mathcal{Z})$.

The function f is locally Lipschitz if ℓ and h (as a function of θ_h) are locally Lipschitz. As an example for a locally Lipschitz function f, we extend supervised neural learning machines to k-structures:

- *Loss function*: The loss
$$\ell(x,y) = D_*\left(\mu(x), \mu(y)\right)^2$$
is locally Lipschitz as a function of x.

- *Hypothesis space*: Consider the set $\mathcal{H}_{NN}$ of all functions $g : \mathcal{X} \to \mathcal{Y}$ that can be implemented by a neural network. Suppose that all functions from $\mathcal{H}_{NN}$ are smooth. If $\dim(\mathcal{V}_{\mathcal{Y}}) = M$, then g is of the form $g = (g_1, \ldots, g_M)$, where the g_i are the component functions of g. For each component g_i, the pointwise maximizer
$$h_i(x) = \max_{T \in \mathcal{T}} g_i(Tx)$$
is locally Lipschitz. Hence, $h = (h_1, \ldots, h_M)$ is locally Lipschitz.

Compared to common models in predicting structures as applied by Taskar (2004); Tsochantaridis, Hofmann, Joachims, and Altun (2004), the proposed approach differs in two ways: First, the proposed cost function requires no indirection via a score function $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ to select the prediction from $\mathcal{Y}$ by maximizing f for a given input from $\mathcal{X}$. Second, the proposed approach suggests a formulation that can be exploited to approximately solve discrete and continuous prediction problems.

4.6 Frequent Substructures

Our aim is to find the most frequent substructure occurring in a finite data set $\mathcal{D}$ of k-structures. To show how to apply the theory of $\mathcal{T}$-spaces to this problem, we consider a simplified setting.

First we define what we mean by a substructure. A k-structure $X' = (\mathbb{Z}_m, \mathcal{R}', \alpha')$ is said to be a *substructure* of a k-structure $X = (\mathbb{Z}_n, \mathcal{R}, \alpha)$, if there is an isomorphic embedding $\phi : \mathbb{Z}_m \to \mathbb{Z}_n$.

Next, we restrict ourselves to k-structures with attributes from $[0,1] \subseteq \mathbb{R}$ for the sake of simplicity. Let
$$\mathcal{B}_{\mathcal{T}} = \left\{X = (\mathbb{Z}_n, \mathcal{R}_i, \alpha) \in \mathcal{X}_{\mathcal{T}} \,:\, \alpha(X) \subseteq \{0,1\}\right\},$$
$$\mathcal{U}_{\mathcal{T}} = \left\{X = (\mathbb{Z}_n, \mathcal{R}_i, \alpha) \in \mathcal{X}_{\mathcal{T}} \,:\, \alpha(X) \subseteq [0,1]\,\right\}$$
be the set of all $\mathcal{A}$-attributed k-structures $X \in \mathcal{X}_{\mathcal{T}}$ with attributes from $\{0,1\}$ and $[0,1]$, respectively. Suppose that $\mathcal{D} = \{X_1, \ldots, X_k\} \subseteq \mathcal{B}_{\mathcal{T}}$ is a set of k-structures.

The characteristic function of the i-th structure $X_i \in \mathcal{D}$
$$\chi_i(X) = \begin{cases} 1 & : \ X \text{ is a substructure of } X_i \\ 0 & : \ \text{otherwise} \end{cases}.$$
indicates whether the k-structure X is a substructure of X_i. We say X^* is a *maximal frequent substructure* of order m if it solves the following discrete problem
$$\begin{array}{ll} \text{maximize} & F(X) = \sum_{i=1}^k \chi_i(X) \\ \text{subject to} & |X| = m \\ & X \in \mathcal{B}_{\mathcal{T}}. \end{array}$$
We cast the discrete to a continuous problem. For this, we define
$$F_i(X) = \frac{\langle X, X_i \rangle^*}{\|X\|_*^2}.$$

for all $X \in \mathcal{U}_{\mathcal{T}}$. We have $F_i(X) \in [0,1]$ with $F_i(X) = 1$ if, and only if, X is a substructure of X_i. Consider for a moment the problem to maximize the criterion function

$$G(X) = \sum_{i=1}^{k} F_i(X).$$

The problem with this criterion function is that a maximizer $X \in \mathcal{U}_{\mathcal{T}}$ of G could be a k-structure not occurring as a substructure in any of the k-structures from $\mathcal{D}$. To fix this problem, we use the soft-max function $\exp(\beta(F_i(X) - 1))$ with control parameter β. In the limit $\beta \to \infty$, the i-th soft-max function reduces to the characteristic function χ_i. Given a fixed $\beta > 0$, the soft-max formulation of the frequent subgraph problem is of the form

$$\begin{array}{ll} \text{maximize} & F_\beta(X) = \sum_{i=1}^{k} \exp\{\beta(F_i(X) - 1)\} \\ \text{subject to} & |X| = m \\ & X \in \mathcal{U}_{\mathcal{T}}. \end{array}$$

The representation function of F_β is of the form

$$f_\beta(x) = \sum_{i=1}^{k} \exp\left\{\beta\left(\frac{s_i(x)}{\|x\|} - 1\right)\right\},$$

where $s_i(x) = \max_{T \in \mathcal{T}} \langle x, Tx_i \rangle$. Applying the rules of calculus yields that f_β is locally Lipschitz.

The common approach casts the frequent subgraph mining problem to a search problem in a state space, which is then solved by a search algorithm (Dehaspe, Toivonen, and King, 1998; Han, Pei, and Yin, 2000; Inokuchi, Washio, and Motoda, 2000; Kuramochi and Karypis, 2001; Yan and Han, 2002). Here, we suggest a continuous cost function for the frequent subgraph mining problem that can be solved using optimization based methods (see Section 3.4).

5. Experimental Results

To demonstrate the effectiveness and versatility of the proposed framework, we applied it to the problem of determining a sample mean of randomly generated point patterns and weighted graphs as well as to central clustering of letters and protein structures represented by graphs.

5.1 Sample Mean

To assess the performance and to investigate the behavior of the subgradient and approximated subgradient method for determining a sample mean, we conducted an experiments on random graphs, letter graphs, and chemical graphs. For computing approximate subgradients we applied the graduated assignment (`GA`) algorithm (see Appendix D). For data sets consisting of small graphs, we also applied a depth first search (`DF`) algorithm that guarantees to return an exact subgradient.

5.1.1 DATA

Random Graphs. The first data set consists of randomly generated graphs. We sampled k graphs by distorting a given initial graph according to the following scheme: First, we randomly generated an initial graph M_0 with 6 vertices and edge density 0.5. Next, we assigned a feature vector to

	depth-first search	graduated assignment	set mean
Random Graphs	29.6 (± 5.3)	34.5 (± 6.6)	43.0 (± 7.5)
Letter Graphs	42.3 (± 10.1)	43.9 (± 11.1)	60.5 (± 16.6)
Molecules		262.2 (± 113.6)	338.0 (± 115.0)

Table 1: Average SSD of sample mean approximated by depth-first search and graduated assignment. As reference value the average SSD of the set mean is shown in the last column. Standard deviations are given in parentheses.

each vertex and edge of M_0 drawn from a uniform distribution over $[0,1]^d$ ($d = 3$). Given M_0, we randomly generated k distorted graphs as follows: Each vertex and edge was deleted with 20% probability. A new vertex was inserted with 10% probability and randomly connected to other vertices with 50% probability. Uniform noise from $[0,1]^d$ with standard deviation $\sigma \in [0,1]$ was imposed to all feature vectors. Finally, the vertices of the distorted graphs were randomly permuted.

We generated 500 samples each consisting of $k = 10$ graphs. For each sample the noise level $\sigma \in [0,1]$ was randomly prespecified.

Letter Graphs. The letter graphs were taken from the IAM Graph Database Repository. [5] The graphs represent distorted letter drawings from the Roman alphabet that consist of straight lines only. Lines of a letter are represented by edges and ending points of lines by vertices. Each vertex is labeled with a two-dimensional vector giving the position of its end point relative to a reference coordinate system. Edges are labeled with weight 1. We considered the 150 letter graphs representing the capital letter *A* at a medium distortion level.

We generated 100 samples each consisting or $k = 10$ letter graphs drawn from a uniform distribution over the data set of 150 graph letters representing letter *A* at a medium distortion level.

Chemical Graphs. The chemical compound database was taken from the gSpan site[6]. The data set contains 340 chemical compounds, 66 atom types, and 4 types of bonds. On average a chemical compound consists of 27 vertices and 28 edges. Atoms are represented by vertices and bonds between atoms by edges. As attributes for atom types and type of bonds, we used a 1-to-k binary encoding, where $k = 66$ for encoding atom types and $k = 4$ for encoding types of bonds.

We generated 100 samples each consisting of $k = 10$ chemical graphs drawn from a uniform distribution over the data set of 340 chemical graphs.

5.1.2 EVALUATION PROCEDURE

As performance measure, we used the average sum-of-squared distances (SSD) of the sample mean described in Section 4.2 averaged over all samples. The average SSD of the set mean graph serves as our reference value. The set mean is an element from the set $\mathcal{D}$ itself that minimizes the SSD over all structures from $\mathcal{D}$.

Algorithm 2 (K-Means for Structures)

```
initialize number $k$ of clusters
initialize cluster centers $Y_1, \dots, Y_k$
repeat
    classify structures $X_i$ according to nearest $Y_j$
    recompute $Y_j$
until some termination criterion is satisfied
```

5.1.3 RESULTS

Table 1 shows the average SSD and its standard deviation. The results show that using exact subgradients gives better approximations of the sample mean than using approximated subgradients. Compared with the set median, the results indicate that the subgradient and approximated subgradient method have found reasonable solutions in the sense that the resulting average SSD is lower.

5.2 Central Clustering

Based on the concept of sample mean for structures, we applied the structural versions of k-means and simple competitive learning on four data sets in order to assess and compare the performance of subgradient methods.

5.2.1 CENTRAL CLUSTERING ALGORITHMS FOR GRAPHS

We consider k-means and simple competitive learning in order to minimize the cluster objective (see Section 4.2)[7]

$$F(M,\mathcal{Y},\mathcal{X}) = \frac{1}{2}\sum_{j=1}^{k}\sum_{i=1}^{m} m_{ij} D(X_i,Y_j).$$

K-means for graphs. The structural version of k-means is outlined in Algorithm 2. This method operates as the EM algorithm of standard k-means, where the chosen distortion measure in the E-step is D to classify the structures X_i according to nearest cluster center Y_j . In the M-step the basic incremental subgradient method described in Algorithm 1 is applied to recompute the means.

Simple competitive learning. The structural version of simple competitive learning corresponds to the basic incremental subgradient method described in Algorithm 1 for minimizing the cluster objective $F(X)$.

5.2.2 DATA

We selected four data sets described in Riesen and Bunke (2008). The data sets are publicly available at the IAM Graph Database Repository. Each data set is divided into a training, validation, and a test set. In all four cases, we considered data from the test set only. The description of the data

5. The repository can be found at `http://www.iam.unibe.ch/fki/databases/iam-graph-database`.
6. gSpan can be found at `http://www.xifengyan.net/software/gSpan.htm`.
7. We replaced the factor $1/m$ by the factor $1/2$ for convenience of presentation of our results.

data set	#graphs)	#(classes)	avg(nodes)	max(nodes)	avg(edges)	max(edges)
letter	750	15	4.7	8	3.1	6
grec	528	22	11.5	24	11.9	29
fingerprint	900	3	8.3	26	14.1	48
molecules	100	2	24.6	40	25.2	44

Table 2: Summary of main characteristics of the data sets used for central clustering.

sets are mainly excerpts from Riesen and Bunke (2008). Table 2 provides a summary of the main characteristics of the data sets.

Letter Graphs. We consider all 750 graphs from the test data set representing distorted letter drawings from the Roman alphabet that consist of straight lines only (A, E, F, H, I, K, L, M, N, T, V, W, X, Y, Z). The graphs are uniformly distributed over the 15 classes (letters). The letter drawings are obtained by distorting prototype letters at low distortion level. Lines of a letter are represented by edges and ending points of lines by vertices. Each vertex is labeled with a two-dimensional vector giving the position of its end point relative to a reference coordinate system. Edges are labeled with weight 1. Figure 6 shows a prototype letter and distorted version at various distortion levels.

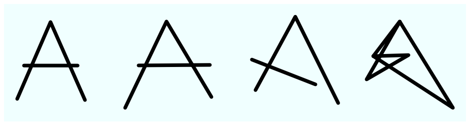

Figure 6: Example of letter drawings: Prototype of letter A and distorted copies generated by imposing low, medium, and high distortion (from left to right) on prototype A.

GREC Graphs. The GREC data set (Dosch and Valveny, 2006) consists of graphs representing symbols from architectural and electronic drawings. We use all 528 graphs from the test data set uniformly distributed over 22 classes. The images occur at five different distortion levels. In Figure 7 for each distortion level one example of a drawing is given. Depending on the distortion level, either erosion, dilation, or other morphological operations are applied. The result is thinned to obtain lines of one pixel width. Finally, graphs are extracted from the resulting denoised images by tracing the lines from end to end and detecting intersections as well as corners. Ending points, corners, intersections and circles are represented by vertices and labeled with a two-dimensional attribute giving their position. The vertices are connected by undirected edges which are labeled as line or arc. An additional attribute specifies the angle with respect to the horizontal direction or the diameter in case of arcs.

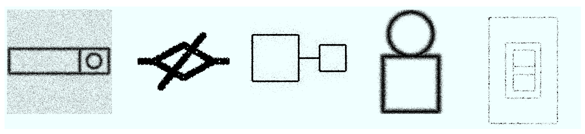

Figure 7: GREC symbols: A sample image of each distortion level

Fingerprint Graphs. We consider a subset of 900 graphs from the test data set representing fingerprint images of the NIST-4 database (Watson and Wilson, 1992). The graphs are uniformly distributed over three classes *left*, *right*, and *whorl*. A fourth class (*arch*) is excluded in order to keep the data set balanced. Fingerprint images are converted into graphs by filtering the images and extracting regions that are relevant (Neuhaus and Bunke, 2005). Relevant regions are binarized and a noise removal and thinning procedure is applied. This results in a skeletonized representation of the extracted regions. Ending points and bifurcation points of the skeletonized regions are represented by vertices. Additional vertices are inserted in regular intervals between ending points and bifurcation points. Finally, undirected edges are inserted to link vertices that are directly connected through a ridge in the skeleton. Each vertex is labeled with a two-dimensional attribute giving its position. Edges are attributed with an angle denoting the orientation of the edge with respect to the horizontal direction. Figure 8 shows fingerprints of each class.

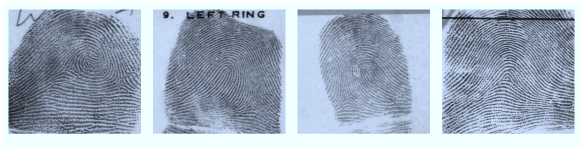

Figure 8: Fingerprints: (a) Left (b) Right (c) Arch (d) Whorl. Fingerprints of class arch are not considered.

Molecules. The mutagenicity data set consists of chemical molecules from two classes (mutagen, non-mutagen). The data set was originally compiled by Kazius, McGuire, and Bursi (2005) and reprocessed by Riesen and Bunke (2008). We consider a subset of 100 molecules from the test data set uniformly distributed over both classes. We describe molecules by graphs in the usual way: atoms are represented by vertices labeled with the atom type of the corresponding atom and bonds between atoms are represented by edges labeled with the valence of the corresponding bonds. We used a 1-to-*k* binary encoding for representing atom types and valence of bonds, respectively.

5.2.3 GENERAL EXPERIMENTAL SETUP

In all experiments, we applied k-means and simple competitive learning for graphs to the aforementioned data sets. We used the following experimental setup:

Performance measures. We used the following measures to assess the performance of an algorithm on a data set: (1) error value of the cluster objective (see Section 4.2), (2) classification accuracy, and (3) silhouette index. The silhouette index is a cluster validation index taking values from $[-1,1]$. Higher values indicate a more compact and well separated cluster structure. For more details we refer to Theodoridis and Koutroumbas (2009).

Initialization of the clustering algorithms. The number k of centroids as shown in Table 3 was chosen by compromising a satisfactory classification accuracy against the silhouette index. To initialize both clustering algorithms, we used a modified version of the ”furthest first” heuristic (Hochbaum and Shmoys, 1985). For each data set $\mathcal{S}$, the first centroid Y_1 is initialized to be a graph closest to the sample mean of $\mathcal{S}$. Subsequent centroids are initialized according to

$$Y_{i+1} = \arg\max_{X \in \mathcal{S}} \min_{Y \in \mathcal{Y}_i} D(X,Y),$$

data set	k	measure	km	cl
letter	30			
		error	11.6	11.1
		accuracy	0.86	0.90
		silhouette	0.38	0.40
grec	33			
		error	32.7	27.6
		accuracy	0.84	0.87
		silhouette	0.40	0.42
fingerprint	60			
		error	1.88	1.30
		accuracy	0.81	0.79
		silhouette	0.32	0.34
molecules	10			
		error	56.0	53.8
		accuracy	0.68	0.70
		silhouette	0.04	0.05

Table 3: Results of k-means (km) and simple competitive learning (cl) on four data sets.

where $\mathcal{Y}_i$ is the set of the first i centroids chosen so far.

Subgradient and graph distance calculations. For subgradient and graph distance calculations, we applied a depth first search algorithm on the letter data set and the graduated assignment algorithm (Gold and Rangarajan, 1996) on the grec, fingerprint, and molecule data set.

5.2.4 RESULTS

Table 3 summarizes the results. The first observation to be made is that simple competitive learning performs slightly better than k-means with respect to all three performance measures. This is in contrast to findings on standard k-means and simple competitive learning in vector spaces. The second observation is that both k-means algorithms yield satisfying classification accuracies on all data sets. This result shows that approximated subgradient methods can be applied to central clustering in the domain of graphs.

5.3 Clustering Protein Structures

In our last experiment, we compared the performance of k-means and simple competitive learning of graphs with hierarchical clustering applied on protein structures.

5.3.1 DATA: PROTEIN CONTACT MAPS

One common way to model the 3D structure of proteins are contact maps. A contact map is a graph $X = (V, E)$ with ordered vertex set. Vertices represent residues. Two vertices are connected by an edge (contact) if the spatial distance of the corresponding residues is below some prespecified threshold.

ID	domain	ID	domain	ID	domain	ID	domain
1	1b00A	11	4tmyB	21	2b3iA	31	1tri
2	1dbwA	12	1rn1A	22	2pcy	32	3ypiA
3	1nat	13	1rn1B	23	2plt	33	8timA
4	1ntr	14	1rn1C	24	1amk	34	1ydvA
5	1qmpA	15	1bawA	25	1aw2A	35	1b71A
6	1qmpB	16	1byoA	26	1b9bA	36	1bcfA
7	1qmpC	17	1byoB	27	1btmA	37	1dpsA
8	1qmpD	18	1kdi	28	1htiA	38	1fha
9	3chy	19	1nin	29	1tmhA	39	1ier
10	4tmyA	20	1pla	30	1treA	40	1rcd

Table 4: PDB domain names of the Skolnick test set and their assigned indexes (ID).

No	Style	Residues	Seq. Sim.	Proteins
1	alpha-beta	124	15-30%	1-14
2	beta	99	35-90%	15-23
3	alpha-beta	250	30-90%	24-34
4		170	7-70%	35-40

Table 5: Characteristic properties of the Skolnick test set as taken from Caprara and Lancia (2002). Shown are the fold style, mean number of residues, and the range of similarity obtained by sequence alignment of the protein domains.

We used the Skolnick test set consisting of 40 protein contact maps provided by Xie and Sahinidis (2007). Table 4 shows the PDB domain names of the test set and their assigned indexes.[8] Table 5 describes characteristic properties of the protein domains. The characteristic feature of the Skolnick data is that sequence similarity fails for correct categorization of the proteins as indicated by the fourth column (*Seq. Sim.*) of Table 5. This motivates structural alignment for solving the Skolnick clustering test.

5.3.2 ALGORITHMS

To cluster the contact maps, we minimized the cluster objective described in Section 4.3

$$F(M,\mathcal{Y},\mathcal{X}) = \frac{1}{m}\sum_{j=1}^{k}\sum_{i=1}^{m} m_{ij} D(X_i,Y_j),$$

using the extensions of k-means and simple competitive learning. The chosen distance measure D for both, the letter graphs and the contact maps, is the minimizer of the standard Euclidean metric.

8. The Protein Data Bank (PDB) is a repository for the 3D structures of proteins, nucleic acids, and other large biological molecules.

C	ID	Fold	Superfamily	Family
1	1-11	Flavodin-like	Che Y-like	Che Y-related
2	12-14	Microbial ribonucl.	Microbial ribonucl.	Fungi ribonucl.
3	15-23	Cuperdoxin-like	Cuperdoxins	Plastocyanim-like Plastoazurin-like
4	24-34	TIM-beta alpha-barrel	Triosephosphate isomerase (TIM)	Triosephosphate isomerase (TIM)
5	35-40	Ferritin-like	Ferritin-like	Ferritin

Table 6: Clusters of Skolnick proteins detected by competitive learning and k-means. Shown are the cluster memberships of the proteins via their indexes (ID) as assigned in Table 5. The clusters perfectly agree with the fold, family, and superfamily according to SCOP categories.

For letter graphs the underlying transformation set $\mathcal{T}$ is the set of all possible vertex permutations. In the case of contact maps, the set $\mathcal{T}$ is the subset of all partial vertex permutations that preserve the order of the vertices.

For subgradient and distance calculations, we used a combination of graduated assignment and dynamic programming (Jain and Lappe, 2007).

5.3.3 RESULTS

Competitive learning and k-means both correctly categorized the 40 proteins in 5 clusters according to the SCOP categories as shown in Table 6.[9] This result was also achieved by previous approaches based on hierarchical clustering using pairwise similarity matrices (Xie and Sahinidis, 2007; Caprara and Lancia, 2002; Caprara, Carr, Istrail, Lancia, and Walenz, 2004).

Competitive learning and k-means require less pairwise structural alignments than pairwise clustering. Pairwise clustering of 40 structures requires 780 structural alignments. In contrast, competitive learning required 120 and k-means 440 structural alignments. One problem of central clustering algorithms applied to contact maps is the increasing size of the cluster centers caused by the updating step. A solution to this problem is to restrict the vertices of a cluster center to those vertices that occur in at least one cluster member. In doing so, spurious vertices of former cluster members are removed.

The distinguishing feature of central clustering of structures is that we obtain prototypes for each cluster. According to Jain and Obermayer (2009), the sample mean is equivalent to the multiple alignment of proteins, which is like clustering an essential task in bioinformatics. Hence, central clustering of protein structures can solve two tasks simultaneously, categorizing the proteins and and multiple aligning cluster members, which is useful for protein structure classification, structure-based function prediction, and highlighting structurally conserved regions of functional significance.

9. The *Structural Classification of Proteins* (SCOP) database is a largely manual classification of protein structural domains based on similarities of their amino acid sequences and 3D structures.

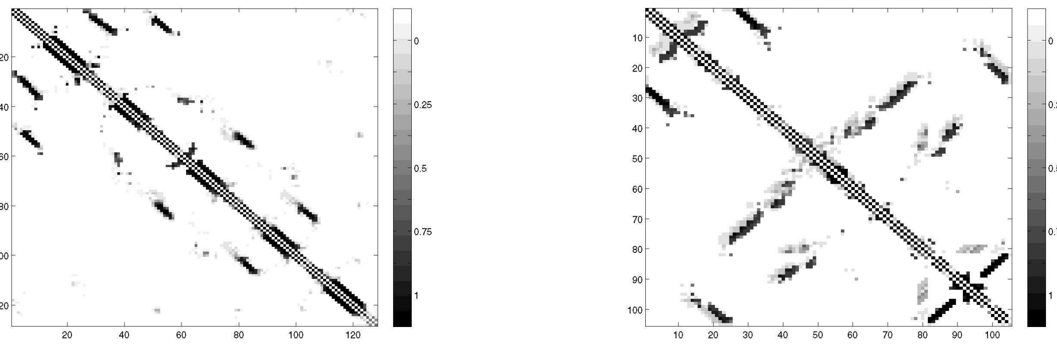

Figure 9: Shown are approximated sample means of the Che Y-like superfamily of cluster C1 (**left**) and the Cuperdoxins superfamily family of cluster C3 (**right**). Diagonal elements show the residues and off-diagonal elements the contacts. Darker shading refers to a higher relative frequency of occurrence of residues/contacts over all cluster members.

Figure 9 shows approximations of sample means of the two largest clusters computed by competitive learning.

6. Summary

In this contribution, we described a generic technique of how to generalize classical learning approaches and other problems from pattern recognition to structured domains. The proposed technique is based on the notion of $\mathcal{T}$-space. A $\mathcal{T}$-space is a quotient set of a metric vector space—the representation space—with all the vectors identified that represent the same structure. The equivalence classes of representation vectors are determined by the subgroup of homogeneous isometrics $\mathcal{T}$. This constructions turns out to be a convenient abstraction of combinatorial structures to formally adopt geometrical and analytical concepts from vector spaces.

The metric of the representation space $\mathcal{X}$ induces a metric on the $\mathcal{T}$-space. A norm on $\mathcal{X}$ induces a $\mathcal{T}$-norm on $\mathcal{X}_{\mathcal{T}}$. The $\mathcal{T}$-norm corresponds to the same geometric concept of length as the standard norm. Thus, different vector representations of the same structure have the same length. An inner product $\langle\cdot,\cdot\rangle$ on $\mathcal{X}$ induces the inner $\mathcal{T}$-product on $\mathcal{X}_{\mathcal{T}}$, which is not bilinear but has the same geometrical properties as $\langle\cdot,\cdot\rangle$. In other words, the Cauchy-Schwarz inequality is valid for structures. This result gives rise to a well-defined geometric concept of angle between structures.

The interplay of geometrical intuition, the algebraic group structure of the transformation set $\mathcal{T}$, and the link to the properties of a vector space via the membership function yields the well-defined notions of $\mathcal{T}$-differentiability and $\mathcal{T}$-gradient that generalize the standard definitions of differentiability and gradient of a smooth function at some point from a vector space. In particular, the $\mathcal{T}$-gradient of a $\mathcal{T}$-function at a $\mathcal{T}$-differentiable point is a well-defined structure pointing in direction of steepest ascent and satisfies the necessary condition of optimality. Therefore, we can apply local gradient information to minimize smooth $\mathcal{T}$-functions.

One application of the theory of $\mathcal{T}$-spaces are problems from structural pattern recognition. For selected problems, we presented continuous optimization problems, where the cost functions defined on $\mathcal{T}$-spaces are locally Lipschitz. Locally Lipschitz functions are nonsmooth on a set of Lebesgue measure zero, but admit a generalized gradient at non-differentiable points. The field of nonsmooth optimization provides techniques like the subgradient method to minimize this class of nonsmooth functions.

As case studies, we considered the problem of computing a sample mean and central clustering in the domain of graphs. The cost functions are locally Lipschitz, but computation of a subgradient is computationally intractable. To cope with the computational complexity, we suggested an approximate subgradient method that chooses the opposite of a direction close to the generalized gradient as descent direction. We illustrated that the proposed method is capable to minimize the cost function of the case study. Even so the high computational complexity of deriving a subgradient demands a reevaluation of existing nonsmooth optimization methods and asks for devising algorithms that use approximations of the generalized gradient.

Appendix A. Introduction to $\mathcal{T}$-spaces

This section formally introduces $\mathcal{T}-spaces$ and presents proofs.

A.1 $\mathcal{T}$-spaces

Let $\mathcal{X} = \mathbb{R}^n$ be the n-dimensional Euclidean vector space, and let $\mathcal{T}$ be a subgroup of the group of all $n \times n$ permutation matrices. Then the binary operation

$$\cdot : \mathcal{T} \times \mathcal{X} \to \mathcal{X}, \quad (T,x) \mapsto Tx$$

is a group action of $\mathcal{T}$ on $\mathcal{X}$. For $x \in \mathcal{X}$, the *orbit* of x is denoted by

$$[x]_{\mathcal{T}} = \{Tx : T \in \mathcal{T}\}.$$

If no misunderstanding can occur, we write $[x]$ instead of $[x]_{\mathcal{T}}$.

A *$\mathcal{T}$-space* over $\mathcal{X}$ is the orbit space $\mathcal{X}_{\mathcal{T}} = \mathcal{X}/\mathcal{T}$ of all orbits of $x \in \mathcal{X}$ under the action of $\mathcal{T}$. We call $\mathcal{X}$ the *representation space* of $\mathcal{X}_{\mathcal{T}}$. By

$$\mu : \mathcal{X} \to \mathcal{X}_{\mathcal{T}}$$

we denote the *membership function* that sends vector representations to the structure they describe.

The following notations, definitions, and results are useful to simplify technicalities. We use capital letters $X, Y, Z, \ldots$ to denote the elements of $\mathcal{X}_{\mathcal{T}}$. Suppose that $X = \mu(x)$ for some $x \in \mathcal{X}$. Then we identify X with $[x]$ and make use of sloppy notations like, for example, $x \in X$ to denote $x \in [x]$.[10]

By $0_{\mathcal{T}}$ we denote the $\mathcal{T}$-zero of $\mathcal{X}_{\mathcal{T}}$. It is easy to show that $0_{\mathcal{T}}$ has only the zero element $0 \in \mathcal{X}$ as its unique representation vector.

Proposition 2 *Let $\mathcal{X}_{\mathcal{T}}$ be a $\mathcal{T}$-space over the metric vector space $\mathcal{X}$. Then $\mu^{-1}(0_{\mathcal{T}}) = [0] = \{0\}$.*

Proof Follows directly from the fact that each $T \in \mathcal{T}$ is homogeneous and injective. ■

A $\mathcal{T}$-space $\mathcal{X}_{\mathcal{T}}$ has, in fact, a $\mathcal{T}$-zero element, but it is unclear how to define an addition $+$ on $\mathcal{X}_{\mathcal{T}}$ such that $\mathcal{X}_{\mathcal{T}}$ together with $+$ forms a group. The absence of an additive group structure is one of the

10. The notation is sloppy, because X is an element in $\mathcal{X}_{\mathcal{T}}$ and not a set, whereas $[x]$ is a set of equivalent elements from $\mathcal{X}$.

major reasons why analytical tools for structured data are extremely rare compared to the plethora of powerful tools developed for feature vectors residing in some Banach space. To mitigate this drawback of $\mathcal{T}$-spaces $\mathcal{X}_{\mathcal{T}}$, we exploit the vector space axioms of $\mathcal{X}$ via the membership function μ.

We say a membership function $\mu : \mathcal{X} \to \mathcal{X}_{\mathcal{T}}$ is *$\mathcal{T}$-linear* if

$$\text{(TL1)} \quad [x+y] \subseteq [x] \oplus [y] = \{x'+y' : x' \in [x], y' \in [y]\},$$

$$\text{(TL2)} \quad [\lambda x] = \lambda [x] = \{\lambda x' : x' \in [x]\}$$

for all $x,y \in \mathcal{X}$, and for all $\lambda \in \mathbb{R}$. Note that for (TL1) we have a subset relation and for (TL2) equality. This definition has a sound notation but appears to be independent on μ at first glance. Since we identify the orbits $[x]$ in $\mathcal{X}$ with the elements $\mu(x)$ of $\mathcal{X}_{\mathcal{T}}$, we can rewrite (TL1) and (TL2) by slight abuse of notation

$$\begin{aligned} \mu(x+y) &\subseteq \mu(x) \oplus \mu(y), \\ \mu(\lambda x) &= \lambda \mu(x). \end{aligned}$$

Membership functions that are $\mathcal{T}$-linear preserve enough structure to transfer some geometrical and analytical concepts from $\mathcal{X}$ to $\mathcal{X}_{\mathcal{T}}$.

In contrast to the standard definition of linearity, we only require a subset relation in (TL1) rather than equality. The proof of Prop. 3 explains this issue.

Proposition 3 *Let $\mathcal{X}_{\mathcal{T}}$ be a $\mathcal{T}$-space over the Euclidean space $\mathcal{X}$.Then the membership function $\mu : \mathcal{X} \to \mathcal{X}_{\mathcal{T}}$ is T-linear.*

Proof Let $z = x+y$, and let $z' \in [x+y]$. Then there is an element $T \in \mathcal{T}$ with $z' = Tz$. Since T is linear by assumption, we obtain

$$z' = Tz = T(x+y) = Tx + Ty \in [x] \oplus [y].$$

This proves (TL1). The proof of (TL2) is similar. ■

For an intuitive understanding it is sometimes more convenient to use the following notation

$$\begin{aligned} \lambda X &= \mu(\lambda x), \\ X_x + Y_y &= \mu(x+y). \end{aligned}$$

It is important to note that the '$+$' symbol in $X_x + Y_y$ does not refer to some kind of addition in $\mathcal{X}_{\mathcal{T}}$. The notation $X_x + Y_y$ is simply an alternative and for our purposes more convenient way to refer to the element $\mu(x+y) \in \mathcal{X}_{\mathcal{T}}$.

We conclude this section with some further useful technical notations and results. Let $\mathcal{X}^n$ be the n-ary cartesian product of a metric vector space $\mathcal{X}$. Any real-valued function $f : \mathcal{X}^n \to \mathbb{R}$ induces functions

$$\begin{aligned} F^* : \mathcal{X}_{\mathcal{T}}^n \to \mathbb{R}, &\quad (X_1, \ldots, X_n) \mapsto \max\{f(x_1, \ldots, x_n) : x_i \in X_i\}, \\ F_* : \mathcal{X}_{\mathcal{T}}^n \to \mathbb{R}, &\quad (X_1, \ldots, X_n) \mapsto \min\{f(x_1, \ldots, x_n) : x_i \in X_i\}. \end{aligned}$$

Since $\mathcal{T}$ is finite, the orbits $[x]$ of x are finite. Hence, F^* and F_* assume an extremal value. We call F^* *maximizer* and F_* *minimizer* of f on $\mathcal{X}_{\mathcal{T}}^n$. Let F be either a maximizer or minimizer of f. The *support*

$$\operatorname{supp}(F | X_1, \ldots, X_n)$$

of F at $(X_1, \ldots, X_n) \in \mathcal{X}_{\mathcal{T}}^n$ is the set of all elements $(x_1, \ldots, x_n) \in \prod_{i=1}^n X_i$ with $f(x_1 \ldots, x_n) = F(X_1, \ldots, X_n)$. The next results shows a useful property of the support.

Proposition 4 *Let F be the minimizer or maximizer of a function $f : \mathcal{X}^n \to \mathbb{R}$. Let $X_1, \ldots, X_n \in \mathcal{X}_{\mathcal{T}}$. Then for each $x_i \in X_i$ there are a $x_j \in X_j$ for all $j \in \{1, \ldots, n\} \setminus \{i\}$ such that $(x_1, \ldots, x_n) \in$ $\operatorname{supp}(F|X_1, \ldots, X_n)$.*

Proof Let $x_i \in X_i$. Suppose that $(x_1^*, \ldots, x_n^*) \in \operatorname{supp}(F|X_1, \ldots, X_n)$. Then there is a transformation $T \in \mathcal{T}$ with $Tx_i = x_i^*$. Since $\mathcal{T}$ is a group, the inverse T^{-1} exists. Hence, $x_j = T^{-1}x_j^*$ is an element of X_j for all $j \neq i$. We have

$$F(X_1, \ldots, X_n) = f(x_1^*, \ldots, x_n^*) = f(Tx_1, \ldots, Tx_n) = f(x_1, \ldots, x_n).$$

This shows the assertion. ■

A.2 Metric $\mathcal{T}$-Spaces

Let $\mathcal{X}_{\mathcal{T}}$ be a $\mathcal{T}$-space over the metric vector space $(\mathcal{X}, d)$. The minimizer

$$D_* : \mathcal{X}_{\mathcal{T}} \times \mathcal{X}_{\mathcal{T}} \to \mathbb{R}, \quad (X, Y) \mapsto \min\{d(x,y) : x \in X, y \in Y\}.$$

of the metric d is a distance measure on $\mathcal{X}_{\mathcal{T}}$. Theorem 5 shows that D_* is a metric.

Theorem 5 *Let $\mathcal{X}_{\mathcal{T}}$ be a $\mathcal{T}$-space over the metric space $(\mathcal{X}, d)$. Then the minimizer D_* of d is a metric on $\mathcal{X}_{\mathcal{T}}$.*

Proof Let X, Y, $Z \in \mathcal{X}_{\mathcal{T}}$.

1. We show $D_*(X,Y) = 0 \Leftrightarrow X = Y$. Let $x \in X$ be a representation vector of X. According to Prop. 4 there is a $y \in Y$ such that $(x,y) \in \operatorname{supp}(D_*|x,y)$. We have

$$\begin{aligned} D_*(X,Y) = 0 &\Leftrightarrow \forall x \in X\, \exists y \in Y\; d(x,y) = 0 \\ &\Leftrightarrow \forall x \in X\, \exists y \in Y\; x = y \\ &\Leftrightarrow X = Y. \end{aligned}$$

2. Symmetry $D_*(X,Y) = D_*(Y,X)$ follows from symmetry of d.

3. We show $D_*(X,Z) \leq D_*(X,Y) + D_*(Y,Z)$. Let $(x,y) \in \operatorname{supp}(D_*|X,Y)$. There is a $z \in Z$ such that $(y,z) \in \operatorname{supp}(D_*|Y,Z)$. Then

$$\begin{aligned} D_*(X,Y) + D_*(Y,Z) &= d(x,y) + d(y,z) \\ &\geq d(x,z) \\ &\geq \min\{d(x,z) : x \in X, z \in Z\} \\ &= D_*(X,Z). \end{aligned}$$

■

Given the assumptions of Theorem 5, we call $(\mathcal{X}_{\mathcal{T}}, D_*)$ *metric $\mathcal{T}$-space* over $(\mathcal{X}, d)$. A metric space $\mathcal{X}$ is complete if every Cauchy sequence of points in $\mathcal{X}$ converges to a point from $\mathcal{X}$. The next result states that $\mathcal{X}_{\mathcal{T}}$ is complete if $\mathcal{X}$ is complete.

Theorem 6 *Any $\mathcal{T}$-space over a complete metric vector space is a complete metric space.*

Proof Let $\mathcal{X}_{\mathcal{T}}$ be a $\mathcal{T}$-space over the complete metric space $(\mathcal{X}, d)$. According to Theorem 5, $\mathcal{X}_{\mathcal{T}}$ is a metric space with metric D_*. To show that $\mathcal{X}_{\mathcal{T}}$ is complete, consider an arbitrary Cauchy sequence $(X_i)_{i\in\mathbb{N}}$ in $\mathcal{X}_{\mathcal{T}}$. We construct a Cauchy sequence (x_k) such that $(\mu(x_k))$ is a subsequence of (X_i). For any $k > 0$ there is a n_k such that $D_*(X_i, X_j) < 1/2^k$ for all $i, j > n_k$. For each k, there are $x_k \in X_{n_k}$ and $x_{k+1} \in X_{n_{k+1}}$ with $d(x_k, x_{k+1}) \leq 1/2^k$. By the triangle inequality, we have

$$d(x_i, x_j) \leq \sum_{k=i}^{j-1} d(x_k, x_{k+1}) \leq \frac{1}{2^{i-1}}$$

for any i, j with $i < j$. Hence, (x_k) is a Cauchy sequence in $\mathcal{X}$ and $(\mu(x_k))$ a subsequence of (X_i). Since $\mathcal{X}$ is complete, (x_k) converges to a limit point $x \in \mathcal{X}$. By continuity of μ, we have $\lim_{k\to\infty} \mu(x_k) = \mu(x)$, where $\mu(x) \in \mathcal{X}_{\mathcal{T}}$. Thus, the whole sequence (X_i) converges to $\mu(x)$. This shows that $\mathcal{X}_{\mathcal{T}}$ is complete. ■

A.3 $\mathcal{T}$-Spaces over Normed Vector Spaces

Let $\mathcal{X}_{\mathcal{T}}$ be a $\mathcal{T}$-space over the normed vector space $(\mathcal{X}, \|\cdot\|)$. As a normed vector space, $\mathcal{X}$ is a metric space with metric $d(x,y) = \|x - y\|$ for all $x, y \in \mathcal{X}$. For any $T \in \mathcal{T}$, we have

$$\|Tx\| = \|Tx - 0\| = \|Tx - T0\| = \|x - 0\| = \|x\|.$$

Hence, the minimizer $\|\cdot\|_*$ and maximizer $\|\cdot\|^*$ of $\|\cdot\|$ coincide, that is

$$\|X\|_* = \|X\|^* = \|x\| \tag{2}$$

for all $X \in \mathcal{X}_{\mathcal{T}}$ and for all $x \in X$.

We call the minimizer $\|\cdot\|_*$ the *$\mathcal{T}$-norm* induced by the norm $\|\cdot\|$. Note that a $\mathcal{T}$-norm is *not* a norm, because a $\mathcal{T}$-space has no well defined addition. But we can show that a $\mathcal{T}$-norm has norm-like properties. We use the notations λX for $\mu(\lambda X)$ and $X_x + Y_y$ for $\mu(x+y)$ as introduced in Section 3.2, p. 2696.

Proposition 7 *Let $(\mathcal{X}_{\mathcal{T}}, \|\cdot\|_*)$ be a $\mathcal{T}$-space over the normed space $(\mathcal{X}, \|\cdot\|)$. For all $X, Y \in \mathcal{X}_{\mathcal{T}}$, we have*

1. $\|X\|_* = 0$ *if, and only if,* $X = 0_{\mathcal{T}}$.
2. $\|\lambda X\|_* = |\lambda|\,\|X\|_*$ *for all* $\lambda \in \mathbb{R}$.
3. $\|X_x + Y_y\|_* \leq \|X\|_* + \|Y\|_*$ *for all* $x \in X$, $y \in Y$.

Proof Follows directly from first applying Equation (2) and then using the properties of the norm $\|\cdot\|$ defined on $\mathcal{X}$. ■

A.4 $\mathcal{T}$-Spaces over Inner Product Spaces

Let $\mathcal{X}_{\mathcal{T}}$ be a $\mathcal{T}$-space over the inner product space $(\mathcal{X}, \langle\cdot,\cdot\rangle)$, and let

$$\langle\cdot,\cdot\rangle^* : \mathcal{X}_{\mathcal{T}} \times \mathcal{X}_{\mathcal{T}} \to \mathbb{R}, \quad (X,Y) \mapsto \max\{\langle x,y\rangle \,:\, x \in X, y \in Y\}$$

be the maximizer of the inner product $\langle\cdot,\cdot\rangle$.

We call $\langle\cdot,\cdot\rangle^*$ *inner $\mathcal{T}$-product* induced by the inner product $\langle\cdot,\cdot\rangle$. The inner $\mathcal{T}$-product is *not* an inner product, because the maximum-operator in the definition of $\langle\cdot,\cdot\rangle^*$ does not preserve the bilinearity property of an inner product. But we shall show later that an inner $\mathcal{T}$-product satisfies some weaker properties of an inner product.

Any inner product space $\mathcal{X}$ is a normed space with norm $\|x\| = \sqrt{\langle x,x\rangle}$. The norm $\|\cdot\|$ on $\mathcal{X}$ in turn gives rise to the $\mathcal{T}$-norm $\|\cdot\|_*$ on $\mathcal{X}_{\mathcal{T}}$. The next result shows that a $\mathcal{T}$-norm is related to an inner $\mathcal{T}$-product in the same way as a norm to an inner product.

Proposition 8 *Let $(\mathcal{X}_{\mathcal{T}}, \langle\cdot,\cdot\rangle^*)$ be a $\mathcal{T}$-space over the inner product space $(\mathcal{X}, \langle\cdot,\cdot\rangle)$, and let $X \in \mathcal{X}_{\mathcal{T}}$. Then*

1. $\langle X,X\rangle^* = \langle x,x\rangle$ *for all* $x \in X$.
2. $\|X\|_* = \sqrt{\langle X,X\rangle^*}$.

Proof

1. X is the orbit of x under the group action $\mathcal{T}$. The assertion follows from the fact that each transformation T of $\mathcal{T}$ satisfies

$$\langle Tx,Tx\rangle = \|Tx\|^2 = \|x\|^2 = \langle x,x\rangle$$

   for all $x \in \mathcal{X}$.

2. Follows from the first part by taking the square root.

■

Using Prop. 8 the following operations to construct $\|\cdot\|_*$ commute

$$\begin{tikzcd}
(\mathcal{X}, \langle\cdot,\cdot\rangle) \arrow[r, "\langle\cdot,\cdot\rangle^*"] \arrow[d, "\|\cdot\|=\sqrt{\langle\cdot,\cdot\rangle}"'] & (\mathcal{X}_{\mathcal{T}}, \langle\cdot,\cdot\rangle^*) \arrow[d, "\|\cdot\|_*=\sqrt{\langle\cdot,\cdot\rangle^*}"] \\
(\mathcal{X}, \|\cdot\|) \arrow[r, "\|\cdot\|_*"'] & (\mathcal{X}_{\mathcal{T}}, \|\cdot\|_*)\,.
\end{tikzcd}$$

Next we show that an inner $\mathcal{T}$-product satisfies some weaker properties related to an inner product.

Proposition 9 *Let $X,Y,Z \in \mathcal{X}_{\mathcal{T}}$, and let $x \in X$, $y \in Y$. Then*

1. $\langle X,X\rangle^* \geq 0$ *with* $\langle X,X\rangle^* = 0 \Leftrightarrow X = 0_{\mathcal{T}}$ (*positive definite*)
2. $\langle X,Y\rangle^* = \langle Y,X\rangle^*$ (*symmetric*)

3. $\langle \lambda X, Y \rangle^* = \lambda \langle X, Y \rangle^*$ *for* $\lambda \geq 0$ (*positive homogeneous*)

4. $\langle X_x + Y_y, Z \rangle^* \leq \langle X, Z \rangle^* + \langle Y, Z \rangle^*$ (*sublinear*)

Proof

1. Follows from Prop. 8 and the positive definiteness of $\langle \cdot, \cdot \rangle$.

2. Follows from the symmetry of $\langle \cdot, \cdot \rangle$.

3. Let $\lambda \geq 0$. Then

$$\begin{aligned} \langle \lambda X, Y \rangle^* &= \max \{ \langle \lambda x, y \rangle \, : \, x \in X, y \in Y \} \\ &= \lambda \max \{ \langle x, y \rangle \, : \, x \in X, y \in Y \} \\ &= \lambda \langle X, Y \rangle^*. \end{aligned}$$

4. Let $W = X_x + Y_y$, and let $(w,z) \in \operatorname{supp}(\langle \cdot, \cdot \rangle^* | W, Z)$. Since μ is $\mathcal{T}$-linear by Prop. 3, we have $W \subseteq X \oplus Y$. Hence, there are $x \in X$ and $y \in Y$ such that $w = x + y$. Thus,

$$\langle W, Z \rangle = \langle w, z \rangle = \langle x + y, z \rangle = \langle x, z \rangle + \langle y, z \rangle.$$

From $\langle x, z \rangle \leq \langle X, Z \rangle^*$ and $\langle y, z \rangle \leq \langle Y, Z \rangle^*$ follows the assertion.

■

From the proof of Prop. 9 follows that $\mathcal{T}$-linearity of the membership function partially preserves the structure of $\mathcal{X}$ such that the inner $\mathcal{T}$-product is a positive definite, symmetric, and sublinear in both arguments.

Any inner product space $(\mathcal{X}, \langle \cdot, \cdot \rangle)$ is a metric space with metric $d(x,y) = \|x - y\|$. The metric d induces a metric D_* on $\mathcal{X}_{\mathcal{T}}$. As for the $\mathcal{T}$-norm, we want to express D_* in terms of $\langle \cdot, \cdot \rangle^*$.

Proposition 10 *Let* $(\mathcal{X}_{\mathcal{T}}, \langle \cdot, \cdot \rangle^*)$ *be a* $\mathcal{T}$*-space over the inner product space* $(\mathcal{X}, \langle \cdot, \cdot \rangle)$*. Then for all* $X, Y \in \mathcal{X}_{\mathcal{T}}$*, we have*

$$D_*(X,Y)^2 = \|X\|_*^2 - 2 \langle X, Y \rangle^* + \|Y\|_*^2.$$

Proof We have

$$\begin{aligned} D_*(X,Y)^2 &= \min \left\{ \|x - y\|^2 \, : \, x \in X, y \in Y \right\} \\ &= \min \{ \langle x - y, x - y \rangle \, : \, x \in X, y \in Y \} \\ &= \min \left\{ \|x\|^2 - 2 \langle x, y \rangle + \|y\|^2 \, : \, x \in X, y \in Y \right\} \\ &= \|x\|^2 - 2 \max \{ \langle x, y \rangle \, : \, x \in X, y \in Y \} + \|y\|^2 \\ &= \|X\|_*^2 - 2 \langle X, Y \rangle^* + \|Y\|_*^2. \end{aligned}$$

■

Next, we extend the Cauchy-Schwarz inequality.

Theorem 11 *Let $(\mathcal{X}_\mathcal{T}, \langle \cdot, \cdot \rangle^*)$ be a $\mathcal{T}$-space over the inner product space $(\mathcal{X}, \langle \cdot, \cdot \rangle)$. Then*

$$\left| \langle X, Y \rangle^* \right| \leq \|X\|_* \|Y\|_* .$$

for all $X, Y \in \mathcal{X}_\mathcal{T}$

Proof Let $(x,y) \in \operatorname{supp}(\langle \cdot, \cdot \rangle^* | X, Y)$. Applying the conventional Cauchy-Schwarz inequality for vectors and using Eq. (2) yields

$$\left| \langle X, Y \rangle^* \right| = |\langle x, y \rangle| \leq \|x\| \, \|y\| = \|X\|_* \|Y\|_* .$$

■

Using Theorem 11, we can show that the angle of structures has a geometrical meaning. For two nonzero structures X and Y, the angle $\theta \in [0, \pi]$ between X and Y is defined (indirectly in terms of its cosine) by

$$\cos \theta = \frac{\langle X, Y \rangle^*}{\|X\|_* \|Y\|_*}.$$

Theorem 11 implies that

$$-1 \leq \frac{\langle X, Y \rangle^*}{\|X\|_* \|Y\|_*} \leq 1$$

and thus assures that this angle is well-defined. It is worthy to mention that $\langle X, Y \rangle^*$ has the same geometrical properties as an inner product, although it does not satisfy the algebraic properties of an inner product. Having the concept of an angle for structures, we can define structural orthogonality in the usual way. Two nonzero structures X and Y, written as $X \perp Y$, are *structurally orthogonal if* $\langle X, Y \rangle^* = 0$. Thus, Theorem 11 constitutes the starting point of a geometry of $\mathcal{T}$-spaces, which we do not further expand.

Appendix B. $\mathcal{T}$-Mappings

This section studies differential and the local Lipschitz property of mappings on $\mathcal{X}_\mathcal{T}$. The key result of this section is that the concept of gradient and its generalizations from nonsmooth analysis can be transferred in a well-defined manner to mappings on $\mathcal{T}$-spaces. Basic definitions and results on nonsmooth analysis of locally Lipschitz mappings are given in Appendix C.

B.1 Properties of $\mathcal{T}$-Mappings

Let $\mathcal{X}_\mathcal{T}$ be a T-space over $\mathcal{X}$ and let $\mathcal{Y}$ be a set. A *$\mathcal{T}$-mapping* is a mapping of the form $F : \mathcal{X}_\mathcal{T} \to \mathcal{Y}$. If $\mathcal{Y}$ is a subset of $\mathbb{R}$, we also call F a *$\mathcal{T}$-function*.

Instead of studying a $\mathcal{T}$-mapping F directly, it is more convenient to consider its *representation mapping* defined by

$$f : \mathcal{X} \to \mathcal{Y}, \quad x \mapsto F \circ \mu(x).$$

Thus, we have to show that analyzing $\mathcal{T}$-mappings is equivalent to analyzing their representation mappings. For this we introduce the notion of $\mathcal{T}$-invariant mapping. A *$\mathcal{T}$-invariant mapping* is a mapping $f : \mathcal{X} \to \mathcal{Y}$ that is constant on the orbits $[x]_\mathcal{T}$ for all $x \in \mathcal{X}$. Obviously, representation mappings are $\mathcal{T}$-invariant. In addition, we have the following universal properties:[11]

11. A universal property can be regarded as some abstract property which requires the existence of a unique mapping under certain conditions.

(UP1) Each mapping $F : \mathcal{X}_\mathcal{T} \to \mathcal{Y}$ has a unique representation $f : \mathcal{X} \to \mathcal{Y}$ with $f = F \circ \mu$.

(UP2) Each $\mathcal{T}$-invariant mapping $f : \mathcal{X} \to \mathcal{Y}$ can be lifted in the obvious way to a unique mapping $F : \mathcal{X}_\mathcal{T} \to \mathcal{Y}$ with $f = F \circ \mu$.

Hence, by (UP1) and (UP2) we may safely identify $\mathcal{T}$-mappings with their representation mappings.

Next, we show that $\mathcal{T}$-spaces over complete metric vector spaces are universal quotients with respect to continuity, the Lipschitz property, and the local Lipschitz property. For this, we need some additional results.

Proposition 12 *Let $(\mathcal{X}_\mathcal{T}, D_*)$ be a metric $\mathcal{T}$-space over the metric space $(\mathcal{X}, d)$. Then the membership function $\mu : \mathcal{X} \to \mathcal{X}_\mathcal{T}$ is a continuous map.*

Proof Let $(x_i)_{i \in \mathbb{N}}$ be a sequence in $\mathcal{X}$ which converges to $x \in \mathcal{X}$. Let (X_i) be the sequence in $\mathcal{X}_\mathcal{T}$ with $X_i = \mu(x_i)$ for all $i \in \mathbb{N}$, and let $X = \mu(x)$. For any $\varepsilon > 0$ there is a number $n = n(\varepsilon)$ with $D_*(X_i, X) \leq d(x_i, x) < \varepsilon$ for all $i > n$. Hence, μ is continuous. ■

A mapping $f : \mathcal{X} \to \mathcal{Y}$ between topological spaces is open if for any open set $\mathcal{U} \in \mathcal{X}$, the image $f(\mathcal{U})$ is open in $\mathcal{Y}$.

Proposition 13 *Let $(\mathcal{X}_\mathcal{T}, D_*)$ be a metric $\mathcal{T}$-space over the metric vector space $(\mathcal{X}, d)$. Then $\mu : \mathcal{X} \to \mathcal{X}_\mathcal{T}$ is an open mapping.*

Proof It is sufficient to show that for any $x \in \mathcal{X}$ and any open neighborhood $\mathcal{U}$ of x there is an open neighborhood $\mathcal{V}_\mathcal{T}$ of $X = \mu(x)$ such that $\mathcal{V}_\mathcal{T} \subseteq \mu(\mathcal{U})$.

Let $x \in \mathcal{X}$, and let $\mathcal{U} \subseteq \mathcal{X}$ be an open set with $x \in \mathcal{U}$. Then there is $\varepsilon > 0$ such that the open neighborhood $\mathcal{N}(x, \varepsilon)$ is contained in $\mathcal{U}$. Let $\mathcal{U}_\mathcal{T} = \mu(U)$ and $\mathcal{V}_\mathcal{T} = \mu(\mathcal{N}(x, \varepsilon))$. Clearly, $X = \mu(x) \in \mathcal{V}_\mathcal{T} \subseteq \mathcal{U}_\mathcal{T}$. We show that $\mathcal{V}_\mathcal{T}$ is open. From $D_*(X, \mu(y)) \leq d(x, y) < \varepsilon$ for all $y \in \mathcal{N}(x, \varepsilon)$ follows $\mathcal{V}_\mathcal{T} \subseteq \mathcal{N}_\mathcal{T}(X, \varepsilon)$. Now let $Y \in \mathcal{N}_\mathcal{T}(X, \varepsilon)$. For x, we can find a a $y \in \mathcal{X}$ with $D_*(X, Y) = d(x, y) < \varepsilon$ by Prop. 4. Hence, $y \in \mathcal{N}(x, \varepsilon)$. This proves that $\mathcal{N}_\mathcal{T}(X, \varepsilon) = \mathcal{V}_\mathcal{T} \subseteq \mathcal{U}_\mathcal{T}$. ■

The next result shows the aforementioned universal property of $\mathcal{X}_\mathcal{T}$ with respect to continuity, the Lipschitz property, and the local Lipschitz property.

Proposition 14 *Let $\mathcal{X}, \mathcal{Y}$ be complete metric vector spaces, and let $\mathcal{X}_\mathcal{T}$ be a metric $\mathcal{T}$-space over X. Suppose that $f : \mathcal{X} \to \mathcal{Y}$ is a $\mathcal{T}$-invariant mapping. If f is continuous (Lipschitz, locally Lipschitz), then f lifts to a unique continuous (Lipschitz, locally Lipschitz) mapping $F : \mathcal{X}_\mathcal{T} \to \mathcal{Y}$ with $f(x) = F(\mu(x))$ for all $x \in \mathcal{X}$.*

Proof By (UP2), the existence of such a $\mathcal{T}$-mapping F implies uniqueness. Thus, it remains to show that F preserves continuity and both Lipschitz properties. In the following let $d_\mathcal{X}$ and $d_\mathcal{Y}$ denote the metric of $\mathcal{X}$ and $\mathcal{Y}$, resp., and let D_* be the metric of $\mathcal{X}_\mathcal{T}$ induced by $d_\mathcal{X}$.

Continuity. Let $\mathcal{U} \subseteq \mathcal{Y}$ open. Then $\mathcal{V} = f^{-1}(\mathcal{U})$ is open in $\mathcal{X}$, because f is continuous. From Prop. 13 follows that $\mathcal{V}_\mathcal{T} = \mu(\mathcal{V})$ is open in $\mathcal{X}_\mathcal{T}$. The assumption follows from the fact $\mathcal{V}_\mathcal{T} = F^{-1}(\mathcal{U})$.

Lipschitz property. Suppose there is a $L \geq 0$ such that

$$d_\mathcal{Y}(f(x), f(y)) \leq L \cdot d_\mathcal{X}(x, y)$$

for all $x, y \in \mathcal{X}$. Let $X, Y \in \mathcal{X}_{\mathcal{T}}$. For $(x,y) \in \operatorname{supp}(D_*|X,Y)$ we have

$$d_{\mathcal{Y}}(F(X), F(Y)) = d_{\mathcal{Y}}(f(x), f(y)) \leq L \cdot d_{\mathcal{X}}(x,y) = L \cdot D_*(X,Y).$$

Since X and Y were chosen arbitrarily, F is Lipschitz.
Local Lipschitz property. Let $X_0 \in \mathcal{X}_{\mathcal{T}}$, and let $x_0 \in X$. Since f is locally Lipschitz at x_0, there is a $L \geq 0$ such that

$$d_{\mathcal{Y}}(f(x), f(y)) \leq L \cdot d_{\mathcal{X}}(x,y)$$

for all x, y from some neighborhood $\mathcal{U} = \mathcal{N}(x_0, \varepsilon)$ in $\mathcal{X}$. Since $\mu : \mathcal{X} \to \mathcal{X}_{\mathcal{T}}$ is an open mapping, there is an open neighborhood $\mathcal{V}_{\mathcal{T}}$ of X_0 with $\mathcal{V}_{\mathcal{T}} \subseteq \mu(\mathcal{U})$. Let $X, Y \in \mathcal{V}_{\mathcal{T}}$ arbitrary. We choose $x, y, y' \in \mu^{-1}(\mathcal{V}_{\mathcal{T}}) \subseteq \mathcal{U}$ with $\mu(x) = X$, $\mu(y) = \mu(y') = Y$, and $(x,y) \in \operatorname{supp}(D_*|X,Y)$. From $x, y' \in \mathcal{U}$ follows $d_{\mathcal{X}}(x,y') < \varepsilon$ and from $(x,y) \in \operatorname{supp}(D_*|X,Y)$ follows $D_*(X,Y) = d_{\mathcal{X}}(x,y)$. Combining both relations and using that D_* is a minimizer of $d_{\mathcal{X}}$ yields

$$D_*(X,Y) = d_{\mathcal{X}}(x,y) \leq d_{\mathcal{X}}(x,y') < \varepsilon.$$

Hence, we have

$$d_{\mathcal{Y}}(F(X), F(Y)) = d_{\mathcal{Y}}(f(x), f(y)) \leq L \cdot d_{\mathcal{X}}(x,y) = L \cdot D_*(X,Y).$$

Since $X, Y \in \mathcal{V}_{\mathcal{T}}$ were chosen arbitrarily, F is locally Lipschitz at X_0. ■

Now we want to study differential properties of $\mathcal{T}$-mappings via their representation mappings. Let $\mathcal{X}_{\mathcal{T}}$ be a $\mathcal{T}$-space over the Euclidean space $(\mathcal{X}, \|\cdot\|)$ and let $\mathcal{Y}$ be another Euclidean space. Suppose that $f : \mathcal{X} \to \mathcal{Y}$ is a $\mathcal{T}$-mapping, which is differentiable at $\bar{x} \in \mathcal{X}$. By

$$Df(\bar{x}) : \mathcal{X} \to \mathcal{Y}, \quad x \mapsto Df(\bar{x})(x)$$

we denote the derivative of f at $\bar{x}$.

Theorem 15 *Let $\mathcal{X}$ and $\mathcal{Y}$ be Euclidean spaces, and let $\mathcal{X}_{\mathcal{T}}$ be a $\mathcal{T}$-space over $\mathcal{X}$. Suppose that $f : \mathcal{X} \to \mathcal{Y}$ is a $\mathcal{T}$-mapping, which is differentiable at $\bar{x} \in \mathcal{X}$. Then f is differentiable at all points $x \in [\bar{x}]$. In addition, we have*

$$Df(T\bar{x}) = Df(\bar{x}) \circ T^{-1}$$

for all $T \in \mathcal{T}$.

Proof By $\|\cdot\|_{\mathcal{X}}$ and $\|\cdot\|_{\mathcal{Y}}$ we denote the norm defined on $\mathcal{X}$ and $\mathcal{Y}$, respectively. Let $x \in [\bar{x}]$. We show that f is differentiable in x. Let $T \in \mathcal{T}$ with $T\bar{x} = x$. For $h \neq 0$, we define the mapping

$$\begin{aligned} r(h) &= \frac{\left\| f(x+h) - f(x) - Df(\bar{x})\left(T^{-1}h\right) \right\|_{\mathcal{Y}}}{\|h\|_{\mathcal{X}}} \\ &= \frac{\left\| f(T\bar{x} + TT^{-1}h) - f(T\bar{x}) - Df(\bar{x})\left(T^{-1}h\right) \right\|_{\mathcal{Y}}}{\|TT^{-1}h\|_{\mathcal{X}}}. \end{aligned}$$

We set $h' = T^{-1}h$ and obtain

$$\begin{aligned} r(h) &= \frac{\| f(T\bar{x} + Th') - f(T\bar{x}) - Df(\bar{x})(h') \|_{\mathcal{Y}}}{\|Th'\|_{\mathcal{X}}} \\ &= \frac{\| f(T(\bar{x} + h')) - f(T\bar{x}) - Df(\bar{x})(h') \|_{\mathcal{Y}}}{\|Th'\|_{\mathcal{X}}}. \end{aligned}$$

Since f is $\mathcal{T}$-invariant, we have $f(T(\bar{x}+h')) = f(\bar{x}+h')$ and $f(T\bar{x}) = f(\bar{x})$. In addition, we have $\|Th'\| = \|h'\|$, because T is an isometry. Thus,

$$r(h) = \frac{\|f(\bar{x}+h') - f(\bar{x}) - Df(\bar{x})(h')\|_{\mathcal{Y}}}{\|h'\|_{\mathcal{X}}}.$$

Since f is differentiable in $\bar{x}$, we have

$$\lim_{h\to 0} r(h) = \lim_{h'\to 0} r(h) = 0.$$

Hence, f is differentiable at x with derivative $Df(x) = Df(T\bar{x}) = Df(\bar{x}) \circ T^{-1}$. ■

A $\mathcal{T}$-mapping $F : \mathcal{X}_{\mathcal{T}} \to \mathcal{Y}$ is said to be *$\mathcal{T}$-differentiable* at $\overline{X} \in \mathcal{X}_{\mathcal{T}}$ if its representation function is differentiable at an arbitrary vector representation of $\overline{X}$. From Theorem 15 follows that differentiability of f at an arbitrary vector representation of $\overline{X}$ implies differentiability at all vector representations of $\overline{X}$. Hence, $\mathcal{T}$-differentiability at $\overline{X} \in \mathcal{X}_{\mathcal{T}}$ is independent from the particular vector representation of $\overline{X}$ and therefore well-defined.

B.2 $\mathcal{T}$-Differentiable Functions

In this section, we study differential properties of $\mathcal{T}$-functions of the form $F : \mathcal{X}_{\mathcal{T}} \to \mathbb{R}$.

Let $f : \mathcal{X} \to \mathbb{R}$ be the representation function of F. Suppose that f is differentiable at $x \in \mathcal{X}$ with gradient $\nabla f(\bar{x})$. Then, by Theorem 15, the $\mathcal{T}$-function F is $\mathcal{T}$-differentiable at $\overline{X} = \mu(\bar{x})$. We call the structure

$$\nabla F(\overline{X}) = \mu(\nabla f(\bar{x})).$$

$\mathcal{T}$-gradient of F at $\overline{X}$. We show that the $\mathcal{T}$-gradient is well-defined, that is independent from the particular choice of vector representation $\bar{x} \in \overline{X}$. To see this let $T \in \mathcal{T}$ be an arbitrary orthogonal transformation from $\mathcal{T}$. By Theorem 15, we have

$$\nabla f(T\bar{x}) = \nabla f(\bar{x})T^{-1} = \nabla f(\bar{x})T',$$

where $T' = T^{-1}$ is the transpose of T. From $\nabla f(\bar{x})T' = T\nabla f(\bar{x})$ follows $\nabla f(T\bar{x}) = T\nabla f(\bar{x})$. Thus, we have

$$\mu(\nabla f(T\bar{x})) = \mu(T\nabla f(\bar{x})) = \mu(\nabla f(\bar{x}))$$

showing that the $\mathcal{T}$-gradient is well-defined.

The $\mathcal{T}$-gradient induces the $\mathcal{T}$-function

$$\nabla F(\overline{X}) : \mathcal{X}_{\mathcal{T}} \to \mathbb{R}, \quad X \mapsto \langle \nabla F(\overline{X}), X\rangle^*.$$

We use this function for showing that the geometrical properties of a gradient also hold for a $\mathcal{T}$-gradient. For this, we define the *directional $\mathcal{T}$-derivative* of F at $\overline{X}$ along the direction V with $\|V\|_* = 1$ by

$$D_V F(\overline{X}) = \langle V, \nabla F(\overline{X})\rangle^*.$$

The first geometrical result shows that the $\mathcal{T}$-gradient is a structure pointing to the direction of steepest ascent.

Proposition 16 *Let $f : \mathcal{X} \to \mathbb{R}$ be a continuously differentiable representation function of a $\mathcal{T}$-function $F : \mathcal{X}_\mathcal{T} \to \mathbb{R}$. Then for all $\bar{X} \in \mathcal{X}_\mathcal{T} \setminus \{0_\mathcal{T}\}$ and all $V \in \mathcal{U}_\mathcal{T} = \{X \in \mathcal{X} : \|X\|_* = 1\}$, we have*

$$\frac{\nabla F(\bar{X})}{\|\nabla F(X)\|_*} = \arg\max_{V \in \mathcal{U}_\mathcal{T}} D_V F(\bar{X}).$$

Proof From the definition of the directional derivative and the implications of Theorem 11 follows

$$D_V F(\bar{X}) = \langle V, \nabla F(\bar{X}) \rangle^* = \|V\|_* \|\nabla F(\bar{X})\|_* \cos \alpha,$$

where α is the angle between V and $\nabla F(\bar{X})$. Hence, the directional derivative $D_V F(\bar{X})$ becomes maximal if V points to the same direction as $\nabla F(\bar{X})$. ■

Next, we show that the necessary condition for optimality can be transferred to $\mathcal{T}$-differentiable $\mathcal{T}$-functions.

Proposition 17 *Let $F : \mathcal{X}_\mathcal{T} \to \mathbb{R}$ be a $\mathcal{T}$-function with a partial differentiable representation function. If $X \in \mathcal{X}_\mathcal{T}$ is a local optimum of F, then we have*

$$\nabla F(X) = 0_\mathcal{T}.$$

Proof Let $f : \mathcal{X} \to \mathbb{R}$ be the representation function of F, and let $x \in X$ be a vector representation of X. Since f is partial differentiable, the gradient of f at x exists. By definition of the $\mathcal{T}$-gradient, we have

$$\mu(\nabla f(x)) = \nabla F(X) = 0_\mathcal{T}.$$

From Prop. 2 follows that 0 is the unique vector representation of $\nabla F(X)$. Thus, any vector representation x of X is a local optimum of the representation function f. Without loss of generality, we assume that $x \in X$ is a local minimum. Then there is an open neighborhood $\mathcal{U}$ of x with $f(x) \leq f(x')$ for all $x' \in \mathcal{U}$. Since μ is an open mapping by Prop. 13, the set $\mathcal{U}_\mathcal{T} = \mu(\mathcal{U})$ is an open neighborhood of X. From the $\mathcal{T}$-invariance of f follows that $F(X) = f(x) \leq f(x') = F(X')$ for all $X' \in \mathcal{U}_\mathcal{T}$. This shows that F is a local minimum. ■

An immediate consequence of the proof is that if x is a local minimum (maximum) of the representation function f then all $x' \in [x]$ are local minima (maxima).

B.3 Pointwise Maximizers

This section introduces and studies differential properties of pointwise maximizers and applies the results to structural similarity and distance functions.

B.3.1 POINTWISE MAXIMIZERS

The *pointwise maximizer* of functions $f_1, \ldots, f_m : \mathcal{U} \to \mathbb{R}$ defined on an open subset $\mathcal{U} \subseteq \mathbb{R}^n$ is the function $f : \mathcal{U} \to \mathbb{R}$ with

$$f(x) = \max_{1 \leq i \leq m} f_i(x).$$

We call the set $\operatorname{supp}(f) = \{f_i : 1 \leq i \leq m\}$ the *support* of f, and its elements *support functions*.

Theorem 18 *Let $f : \mathcal{U} \to \mathbb{R}$ be a pointwise maximizer with finite support* $\operatorname{supp}(f)$*. We have:*

- *If all $f_i \in \operatorname{supp}(f)$ are locally Lipschitz at x, then f is locally Lipschitz at x and*

$$\partial f(x) \subseteq con\{\partial f_i(x) : f_i \in \operatorname{supp}(f) \wedge f_i(x) = f(x)\}. \tag{3}$$

- *If all $f_i \in \operatorname{supp}(f)$ are regular at x, then f is regular at x and equality in (3) holds.*

- *If all $f_i \in \operatorname{supp}(f)$ are smooth at x, then f is regular at x and*

$$\partial f(x) = con\{\nabla f_i(x) : f_i \in \operatorname{supp}(f) \wedge f_i(x) = f(x)\}.$$

Proof Mäkelä and Neittaanmäki (1992), Theorem 3.2.12 and Corollary 3.2.14. ■

If all support functions of f^* are locally Lipschitz, then f^* is also locally Lipschitz and admits a generalized gradient at any point from $\mathcal{U}$. In addition, f^* is differentiable almost everywhere on $\mathcal{U}$ by Rademacher's Theorem (see Appendix, Theorem 23).

Similarly, we can define in the obvious way the *pointwise minimizer* of a finite set of functions. According to Theorem 19, all statements made on pointwise maximizers carry over to pointwise minimizers.

Theorem 19 *If f be locally Lipschitz at x, then*

$$\partial f(\lambda x) = \lambda \partial f(x)$$

for all $\lambda \in \mathbb{R}$.

Proof Mäkelä and Neittaanmäki (1992), Theorem 3.2.4. ■

In the remainder of this section, we consider similarity and distance functions as examples of pointwise maximizers and minimizers, respectively.

B.3.2 SIMILARITY FUNCTIONS: THE GENERAL CASE

We consider similarity functions of the form

$$S^* : \mathcal{X}_{\mathcal{T}} \times \mathcal{X}_{\mathcal{T}} \to \mathbb{R}, \quad (X,Y) \mapsto \max_{x \in X, y \in Y} s(x,y)$$

that are maximizers of similarity functions $s : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. For a given $Y \in \mathcal{X}_{\mathcal{T}}$, we define the function

$$s_Y : \mathcal{X} \to \mathbb{R}, \quad x \mapsto S^*(\mu(x), Y).$$

The function s_Y represents $S^*(\cdot, Y)$ and is a pointwise maximizer with support

$$\operatorname{supp}(s_Y) = \{s_y : s_y(\cdot) = s(\cdot, y), y \in Y\}.$$

If the support functions of s_Y are locally Lipschitz, regular, or smooth, we can apply Theorem 18 to show that s_Y is locally Lipschitz, admits a generalized gradient at each point of its domain, and is differentiable almost everywhere.

B.3.3 SIMILARITY FUNCTIONS: THE INNER $\mathcal{T}$-PRODUCT

As a specific example of a similarity function as a pointwise maximizer, we consider the inner $\mathcal{T}$-product. Suppose that $S^*(\cdot,\cdot) = \langle\cdot,\cdot\rangle^*$. For a fixed structure $Y \in \mathcal{X}_\mathcal{T}$, the support of the pointwise maximizer s_Y is of the form

$$\text{supp}(s_Y) = \{s_y : s_y(\cdot) = \langle\cdot, y\rangle, y \in Y\},$$

As linear functions, these support functions s_y are continuously differentiable. From Theorem 18 follows

- s_Y is locally Lipschitz and regular,
- the generalized gradient $\partial s_Y(x)$ is the convex set
$$\partial s_Y(x) = \text{con}\,\{y \in Y : (x,y) \in \text{supp}(s_Y|x)\}.$$

The next statement follows directly from Prop. 9.

Proposition 20 *The function $s_Y : \mathcal{X} \to \mathbb{R}$ is convex.*

Proof From Prop. 9 follows that s_y is positively homogeneous and sublinear. Hence, s_Y is convex. ■

B.3.4 DISTANCE FUNCTIONS: THE GENERAL CASE

Suppose that we are given an arbitrary distance function of the form

$$D_* : \mathcal{X}_\mathcal{T} \times \mathcal{X}_\mathcal{T} \to \mathbb{R}, \quad (X,Y) \mapsto \min_{x \in X, y \in Y} d(x,y).$$

To apply the theorems on pointwise maximizers, we consider the function

$$\widehat{D}_*(X,Y) = -D_*(X,Y) = \max_{x \in X, y \in Y} -d(x,y).$$

For a given $Y \in \mathcal{X}_\mathcal{T}$, we define the function

$$\widehat{d}_Y : \mathcal{X} \to \mathbb{R}, \quad x \mapsto \widehat{D}_*(\mu(x), Y).$$

The function $\widehat{d}_Y$ represents $\widehat{D}_*(\cdot, Y)$ and is a pointwise maximizer with support

$$\text{supp}\left(\widehat{d}_Y\right) = \left\{\widehat{d}_y : \widehat{d}_y(\cdot) = -d(\cdot, y), y \in Y\right\}.$$

B.3.5 DISTANCE FUNCTIONS: THE METRIC D_*

Let D_* be the metric on $\mathcal{X}_\mathcal{T}$ induced by a metric d on $\mathcal{X}$ of the form $d(x,y) = \|x-y\|$. For a given $Y \in \mathcal{X}_\mathcal{T}$, we define the function

$$\widehat{d}_Y : \mathcal{X} \to \mathbb{R}, \quad x \mapsto \widehat{D}_*(\mu(x), Y).$$

The function $\widehat{d}_Y$ represents $\widehat{D}_*(\cdot, Y)$ and is a pointwise maximizer with support

$$\text{supp}\left(\widehat{d}_Y\right) = \{\widehat{d}_y : y \in Y\},$$

where $\widehat{d}_y(x) = -\|x-y\|$ for all $x \in \mathcal{X}$. The support functions of $\widehat{d}_Y$ are locally Lipschitz and regular on $\mathcal{X}$, and smooth on $\mathcal{X} \setminus \{y\}$. From Theorem 18 follows

- $\widehat{d_Y}$ is locally Lipschitz and regular,
- the generalized gradient $\partial\widehat{d_Y}(x)$ is the convex set

$$\partial\widehat{d_Y}(x) = \begin{cases} \operatorname{con}\left\{-\frac{(x-y)}{\|x-y\|} : y \in Y \wedge (y,x) \in \operatorname{supp}(\widehat{d_Y}|x)\right\} & : \quad x \neq y \\ \{y \in \mathcal{X}_T : \|y\| \leq 1\} & : \quad x = y. \end{cases}$$

As in the Euclidean space, the squared structural Euclidean metric D_*^2 turns out to be more convenient than D_*. As opposed to $\widehat{d_Y}$, the support functions of the squared function $\widehat{d_Y^2}$ are continuously differentiable at all points from $\mathcal{X}$. In particular, we have

- $\widehat{d_Y^2}$ is locally Lipschitz and regular,
- the generalized gradient $\partial\widehat{d_Y^2}(x)$ is the convex set

$$\partial\widehat{d_Y^2}(x) = \operatorname{con}\left\{-2(x-y) : y \in Y \wedge (y,x) \in \operatorname{supp}\left(\widehat{d_Y^2}|x\right)\right\}.$$

Appendix C. Locally Lipschitz Functions

We review some basic properties of locally Lipschitz functions and their generalized gradients. Unless otherwise stated proofs can be found in Clarke (1990), Section 2.3. For a detailed treatment to the first-order generalized derivative we refer to Clarke (1990); Mäkelä and Neittaanmäki (1992).

Let $(\mathcal{X}, d_X)$ and $(\mathcal{Y}, d_Y)$ be metric spaces, and let $\mathcal{U} \subseteq \mathcal{X}$ be an open set. A map $f : \mathcal{X} \to \mathcal{Y}$ is *Lipschitz* on $\mathcal{U}$ if there is a scalar $L \geq 0$ with

$$d_Y(f(u), f(v)) \leq L \cdot d_X(u,v)$$

for all $u, v \in \mathcal{U}$. We say that f is *locally Lipschitz* at $u \in \mathcal{U}$ if f is Lipschitz on some ε-neighborhood $\mathcal{N}(u,\varepsilon) \subseteq \mathcal{U}$ of u.

Proposition 21 *Let $f, g : \mathcal{U} \subseteq \mathcal{X} \to \mathbb{R}$ be locally Lipschitz at u, and let $\lambda \in \mathbb{R}$ be a scalar. Then λf, $f+g$, and $f \cdot g$ are locally Lipschitz at u. If $g(u) \neq 0$, then f/g is locally Lipschitz at u.*

Proposition 22 *Let $f : \mathcal{X} \to \mathcal{Y}$ be locally Lipschitz at x, and let $g : \mathcal{X} \to \mathcal{Z}$ be locally Lipschitz at $y = f(x)$. Then $h = g \circ f$ is locally Lipschitz at x.*

Proof Let $N(x,\varepsilon_x) \subseteq \mathcal{X}$, $N(y,\varepsilon_y) \subseteq \mathcal{Y}$ be neighborhoods of x and y satisfying the following properties: (i) $f(N(x,\varepsilon_x)) \subseteq N(y,\varepsilon_y)$, (ii) there are $L_x, L_y \geq 0$ such that $d_Y(f(u), f(v)) \leq L_x \cdot d_X(u,v)$ for all $u, v \in N(x,\varepsilon_x)$ and $d_Z(g(p), g(q)) \leq L_y \cdot d_Y(p,q)$ for all $p, q \in N(y,\varepsilon_y)$. For any $u, v \in \mathcal{X}$, we have

$$d_Z(g \circ f(u), g \circ f(v)) \leq L_y d_Y(f(u), f(v)) \leq L_y L_x d_X(u,v).$$

■

Theorem 23 (Rademacher) *Let $\mathcal{U} \subseteq \mathbb{R}^n$ be a nonempty open set, and let $f : \mathcal{U} \to \mathbb{R}$ be locally Lipschitz. Then the set of points at which f is not differentiable has Lebesgue measure zero.*

The function f is *directionally differentiable* at $x \in \mathcal{U}$ if the limit

$$f'(x,d) = \lim_{t \downarrow 0} \frac{f(x+td) - f(x)}{t}$$

exists for all directions $d \in \mathbb{R}^n$. In this case, the value $f'(x,d)$ is the *directional derivative* of f at x in the direction d. We call the function f *directionally differentiable* if f is directionally differentiable at all points $x \in \mathcal{U}$.

The *generalized directional derivative* of f at $x \in \mathcal{U}$ in the direction $d \in \mathcal{X}$ is defined by

$$f^\circ(x,d) = \lim \sup_{t \downarrow 0, y \to x} \frac{f(y+td) - f(y)}{t},$$

where $t \downarrow 0$ and $y \to x$ are sequences such that $y + td$ is always in $\mathcal{U}$.

We say f is *regular* at $x \in \mathcal{U}$ if the following conditions are satisfied

1. f is directionally differentiable at x.
2. $f^\circ(\overline{x},d) = f'(x,d)$ for all $d \in \mathbb{R}^n$.

A function f is said to be *smooth* at x if f is continuously differentiable at x. We have the following implications.

Proposition 24 *Let $f : \mathcal{U} \to \mathbb{R}$ be a function. The following implications hold:*

1. *f is smooth at x, then f is locally Lipschitz, regular, continuous, and differentiable at x.*
2. *f is locally Lipschitz or differentiable at x, then f is continuous at x.*

Proof

1. Smoothness implies differentiability and continuity are well-known results from analysis. The locally Lipschitz property follows from Mäkelä and Neittaanmäki (1992), Lemma 3.1.6 and regularity from Mäkelä and Neittaanmäki (1992), Theorem 3.2.2.
2. Both assertions are again well-known results from analysis.

■

The *generalized gradient* $\partial f(x)$ of f at x is the set

$$\partial f(x) = \{y \in \mathcal{X} : f^\circ(x,d) \geq \langle y,d \rangle \text{ for all } d \in \mathcal{X}\}.$$

The elements of the set $\partial f(x)$ are called *subgradients* of f at x.

Theorem 25 *Let $f : \mathcal{U} \to \mathbb{R}$ be a function on the open subset $\mathcal{U}$. We have:*

- *If f is locally Lipschitz and differentiable at x, then*

$$\nabla f(x) \in \partial f(x).$$

- *If f is locally Lipschitz, regular, and differentiable at x, then*

$$\partial f(x) = \{\nabla f(x)\}.$$

Proof Mäkelä and Neittaanmäki (1992), Theorem 3.1.5, Theorem 3.1.7, and Theorem 3.2.4. ■

Appendix D. The Graduated Assignment Algorithm

We use the graduated assignment algorithm to approximate the NP-hard squared distance $D_*(X,Y)^2$ between two weighted graphs and to determine a subgradient from the generalized gradient ∂d_Y^2 (see Section B.3.5). According to Prop. 10, the squared distance $D_*(X,Y)^2$ can be expressed by the inner $\mathcal{T}$-product. Here, we determine $D_*(X,Y)^2$ via $\langle X,Y\rangle^*$.

Let X and Y be weighted graphs with weighted adjacency matrices $X = (x_{ij})$ and $Y = (y_{ij})$. Suppose that X and Y are of order n and m, respectively. Without loss of generality, we assume that $n < m$. By $\Pi = \Pi^{n\times m}$ we denote the set of $(n \times m)$-match matrices $M = (m_{ij})$ with elements from $[0,1]$ such that each row sums to 1 and each column sums to n/m. Then we have

$$\langle X,Y\rangle^* = \max_{M\in\Pi} \left\langle M'XM, Y\right\rangle,$$

where M' denotes the transpose of M. To compute the inner $\mathcal{T}$-product, graduated assignment minimizes

$$F(M) = -\frac{1}{2}\left\langle M'XM, Y\right\rangle.$$

subject to $M \in \Pi$. Suppose that M_0 is an optimal solution. Then $2\left(M_0'XM_0 - Y\right)$ is a subgradient from the generalized gradient $\partial d_Y^2(X)$.

The core of the algorithm implements a deterministic annealing process with annealing parameter T by the following iteration scheme

$$m_{ij}^{(t+1)} = a_i b_j \exp\left(-\frac{1}{T}\sum_{r=1}^{n}\sum_{s=1}^{m} m_{rs}^{(t)}\left\langle x_{ir}, y_{js}\right\rangle\right),$$

where t denotes the time step. The scaling factors a_i, b_i computed by Sinkhorn's algorithm (Sinkhorn, 1964) enforce the constraints of the match matrix. The algorithm in detail is described in Gold, Rangarajan, and Mjolsness (1996).

References

B. Bonev, F. Escolano, M.A. Lozano, P. Suau, M.A. Cazorla, and W. Aguilar. Constellations and the unsupervised learning of graph. *Lecture Notes In Computer Science*, 4538:340, 2007.

A. Caprara and G. Lancia. Structural alignment of large–size proteins via lagrangian relaxation. In *Proceedings of the 6th Annual International Conference on Research in Computational Molecular Biology (RECOMB 2002)*, pages 100–108, 2002.

A. Caprara, R. Carr, S. Istrail, G. Lancia, and B. Walenz. 1001 optimal pdb structure alignments: Integer programming methods for finding the maximum contact map overlap. *Journal of Computational Biology*, 11(1):27–52, 2004.

F. H. Clarke. *Optimization and Nonsmooth Analysis*. SIAM, 1990.

T. F. Cox and M. A. A. Cox. *Multidimensional Scaling*. CRC Press, 2000.

L. Dehaspe, H. Toivonen, and R.D. King. Finding frequent substructures in chemical compounds. In *In Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 30–36, 1998.

P. Dosch and E. Valveny. Report on the second symbol recognition contest. In *GREC 2005*, pages 381–397, 2006.

M.A. Eshera and K.S. Fu. An image understanding system using attributed symbolic representation and inexact graph-matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:604–618, 1986.

T. Gärtner. A survey of kernels for structured data. *ACM SIGKDD Explorations Newsletter*, 5(1): 49–58, 2003.

T. Gärtner. *Kernels for Structured Data*. PhD thesis, University of Bonn, Germany, 2005.

S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(4):377–388, 1996.

S. Gold, A. Rangarajan, and E. Mjolsness. Learning with pre-knowledge: Clustering with point and graph-matching distance measures. *Neural Computation*, 8:787–804, 1996.

L. Goldfarb. A new approach to pattern recognition. *Progress in Pattern Recognition*, 2:241–402, 1985.

D. Goldman, S. Istrail, and C.H. Papadimitriou. Algorithmic aspects of protein structure similarity. In *40th Annual Symposium on Foundations of Computer Science*, pages 512–521, 1999.

T. Graepel and K. Obermayer. A self-organizing map for proximity data. *Neural Computation*, 11: 139–155, 1999.

T. Graepel, R. Herbrich, P. Bollmann-Sdorra, and K. Obermayer. Classification on pairwise proximity data. In *Advances in Neural Information Processing*, volume 11, pages 438–444,, 1999.

S. Günter and H. Bunke. Self-organizing map for clustering in the graph domain. *Pattern Recognition Letters*, 23:401–417, 2002.

D. Gusfield. *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press, 1997.

J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *In Proceedings of the ACM SIGMOD International Conference on Management of Database*, pages 1–12, 2000.

M. Hein, O. Bousquet, and B. Schölkopf. Maximal margin classification for metric spaces. *Journal of Computer and System Sciences*, 71(3):333–359, 2005.

R. Herbrich, T. Graepel, P. Bollmann-Sdorra, and K. Obermayer. Learning a preference relation for information retrieval. In *Workshop Text Categorization and Machine Learning, International Conference on Machine Learning 1998*, page 8084, 1998.

D. Hochbaum and D. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 10(2):180–184, 1985.

S. Hochreiter and K. Obermayer. *Kernel Methods in Computational Biology*, chapter Gene Selection for Microarray Data, pages 319–356. MIT Press, Cambridge, Massachusetts, 2004.

S. Hochreiter and K. Obermayer. Support vector machines for dyadic data. *Neural Computation*, 18:1472–1510, 2006.

T. Hofmann and J. M. Buhmann. Pairwise data clustering by deterministic annealing. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 19(1):1–14, 1997.

L. Holm and C. Sander. Protein structure comparison by alignment of distance matrices. *Journal of Molecular Biology*, 233(1):123–38, 1993.

A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In *In Proceedings of the 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'00)*, pages 13–23, 2000.

B.J. Jain and M. Lappe. Joining softassign and dynamic programming for the contact map overlap problem. In *Proceedings of the 1st International Conference on Bioinformatics Research and Development (BIRD 2007)*, 2007.

B.J. Jain and K. Obermayer. On the sample mean of graphs. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2008)*, 2008.

B.J. Jain and K. Obermayer. Multiple alignment of contact maps. In *International Joint Conference on Neural Networks (IJCNN 2009)*, 2009.

B.J. Jain and F. Wysotzki. Central clustering of attributed graphs. *Machine Learning*, 56(1):169–207, 2004.

X. Jiang, A. Münger, and H. Bunke. On median graphs: Properties, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(10):1144–1151, 2001.

H Kashima, K Tsuda, and A Inokuchi. Marginalized kernels between labeled graphs. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*, 2003.

J. Kazius, R. McGuire, and R. Bursi. Derivation and validation of toxicophores for mutagenicity prediction. *Journal of Medicinal Chemistry*, 48(1):312–320, 2005.

M. Kuramochi and G. Karypis. Frequent subgraph discovery. In *In Proceedings of IEEE International Conference on Data Mining ICDM'01*, pages 313–320, 2001.

V. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics-Doklady*, 10:707–710, 1966.

Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444, 2002.

M. A. Lozano and F. Escolano. Em algorithm for clustering an ensemble of graphs with comb matching. In A. Rangarajan, M. Figueiredo, and J. Zerubia, editors, *Energy Minimization Methods in Computer Vision and Pattern Recognition, EMMCVPR 2003*, LNCS 2683, pages 52–67. Springer, 2003.

B. Luo, R. C. Wilson, and E. R. Hancock. Spectral embedding of graphs. *Pattern Recognition*, 36 (10):2213–2230, 2003.

M.M. Mäkelä and P. Neittaanmäki. *Nonsmooth Optimization: Analysis and Algorithms with Applications to Optimal Control*. World Scientific, 1992.

H. Quang Minh and T. Hofmann. Learning over compact metric spaces. In *Proceedings of the 17th Annual Conference on Learning Theory (COLT 2004)*, 2004.

R. Myers, R.C. Wilson, and E.R. Hancock. Bayesian graph edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(6):628–635, 2000.

M. Neuhaus and H. Bunke. A graph matching based approach to fingerprint classification using directional variance. In *AVBPA 2005*, pages 191–200, 2005.

E. Pekalska, P. Paclik, and R.P.W. Duin. A generalized kernel approach to dissimilarity-based classification. *Journal of Machine Learning Research*, 2:175–211, 2001.

N. Ratliff, J. A. Bagnell, and M. Zinkevich. Subgradient methods for maximum margin structured learning. In *In Proceddings of the 23rd International Conference on Machine Learning (ICML'06), Workshop on Learning in Structured Output Spaces*, 2006.

J.W. Raymond and P. Willett. Maximum common subgraph isomorphism algorithms for the matching of chemical structure. *Journal of Computer-Aided Molecular Design*, 16:521–533, 2002.

K. Riesen and H. Bunke. IAM graph database repository for graph based pattern recognition and machine learning. In *SSPR 2008*, pages 287–297, 2008.

A. Robles-Kelly and E.R. Hancock. Graph edit distance from spectral seriation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):365– 378, 2005.

A. Sanfeliu and K.S. Fu. A distance measure between attributed relational graphs for pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 13:353–362, 1983.

I. J. Schoenberg. On certain metric spaces arising from euclidean spaces by a change of metric and their imbedding in hilbert space. *Annals of Mathematics*, 38(4):787–793, 1937.

L. G. Shapiro and R.M. Haralick. A metric for comparing relational descriptions. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 7:90–94, 1985.

D. Shasha and K. Zhang. Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal of Computing*, 18:245–262, 1989.

N.Z. Shor. *Minimization Methods for non-differentiable Functions*. Springer, 1985.

R. Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *Annals of Mathematical Statistics*, 35(2):876 – 879, 1964.

B. Taskar. *Learning Structured Prediction Models: A Large Margin Approach*. PhD thesis, Stanford University, 2004.

S. Theodoridis and K. Koutroumbas. *Pattern Recognition*. Elsevier, 4 edition, 2009.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *In Proceedings of the 21st International Conference on Machine Learning (ICML'04)*, 2004.

U. von Luxburg and O. Bousquet. Distance-based classification with Lipschitz functions. *Journal of Machine Learning Research*, 5:669–665, 2004.

C. Watson and C. Wilson. NIST Special Database 4, Fingerprint Database. National Institute of Standards and Technology, 1992.

W. Xie and N.V. Sahinidis. A reduction-based exact algorithm for the contact map overlap problem. *Journal of Computational Biology*, 14(5):637–654, 2007.

X. Yan and J. Han. gSpan: Graph-based substructure pattern mining. In *Second IEEE International Conference on Data Mining (ICDM'02)*, pages 721–724, 2002.

K. Zhang. A constrained edit distance between unordered labeled trees. *Algorithmica*, 15:205–222, 1996.